

NCL REFERENCE DOCUMENT

Version 10.1

Revision Date: April 30, 2014



NUMERICAL
CONTROL
COMPUTER
SCIENCES

2600 Michelson Drive, Suite 1700
Irvine, CA 92612
(949) 852-3664

Warning and Disclaimer

Every effort has been made to make this document complete and as accurate as possible.
However, no warranty or fitness is implied.

The information is provided on an "as is" basis. Numerical Control Computer Sciences shall have neither liability nor responsibility to any person or entity with respect to any loss or damages in connection with or rising from the information contained in this document.

Copyright 1983-2014 by Numerical Control Computer Sciences
Irvine, California. Printed in the United States of America.
All rights reserved. The contents of this publication may not
be reproduced in any form or by any means, electronic
or mechanical, including photocopying, recording, or
information storage and retrieval systems, for any
purpose other than the licensee's personal use,
without prior written consent from
Numerical Control Computer Sciences.

TABLE OF CONTENTS

TABLE OF CONTENTS

1	OVERVIEW	1-1
1.1	Convention Used	1-2
1.1.1	Word Conventions	1-2
1.1.2	Modifiers	1-3
1.1.2.1	Vertical Lists	1-3
1.1.2.2	Optional Modifiers	1-3
1.2	NCL System Features	1-3
1.2.1	NCL Product Suite	1-4
1.2.2	NCL Execution Modes	1-4
1.2.2.1	Interactive Execution Mode	1-4
1.2.2.2	Batch Execution Mode	1-5
1.2.3	Input/Output File Support	1-5
1.3	Hardware And OS Requirements	1-7
2	NCL LANGUAGE	2-1
2.1	NCL Programming Language Structure And Syntax	2-1
2.2	Statement Structure	2-1
2.2.1	Statement Size And Format	2-1
2.2.2	Statement Continuation	2-2
2.2.3	Statement Comments	2-2
2.2.3.1	The Double Dollar Sign \$\$	2-2
2.2.3.2	The Percent Sign %	2-3
2.2.3.3	REMARK/ON	2-4
	OFF	
2.3	Statement Syntax	2-4
2.3.1	Vocabulary Words	2-4
2.3.1.1	Major Words	2-5
2.3.1.2	Minor Words	2-5
2.3.1.3	Processor And Postprocessor Words	2-5
2.3.1.4	Modifier Words	2-6
2.3.1.4.1	Direction Modifiers	2-6
2.3.1.4.2	Orientation Modifiers	2-6
2.3.1.4.3	Rotational Modifiers	2-7
2.3.1.4.4	Positional Modifiers	2-7
2.3.1.4.5	Size Modifiers	2-7
2.3.1.4.6	Range Modifiers	2-8

TABLE OF CONTENTS

2.3.1.5	Inclusive Subscripts May Also Be Used With The Following Syntax:	2-9
2.3.2	Data Fields	2-9
2.3.2.1	Explicit Data Fields:	2-9
2.3.2.2	Implicit Data Fields:	2-10
2.3.2.3	Optional Data Fields:	2-10
2.3.3	Identifiers	2-11
2.3.3.1	Default Identifier Naming Convention For Multi-Entities Geometry Creation Commands	2-13
2.3.3.2	Local Identifiers	2-14
2.3.4	Labels	2-15
2.3.5	Punctuation	2-16
2.3.5.1	The Comma “,”	2-16
2.3.5.2	The Slash “/”	2-17
2.3.5.3	The Colon “:”	2-17
2.3.5.4	The Asterisk “*”	2-17
2.3.5.5	The Double Asterisk “**”	2-17
2.3.5.6	The Plus Sign “+”	2-18
2.3.5.7	The Minus Sign “-”	2-18
2.3.5.8	The Period “.”	2-18
2.3.5.9	The Equal Sign “=”	2-18
2.3.5.10	The Left And Right Parentheses “()”	2-19
2.3.5.11	The Left And Right Square Brackets “[]”	2-20
2.3.5.12	The @ (At Sign)	2-20
2.3.5.13	The & (Ampersand Sign)	2-20
2.3.6	Numbers And Computing	2-21
2.3.6.1	Integer	2-21
2.3.6.2	Real Numbers	2-21
2.3.6.3	Arithmetic Operators And The Order Of Operation	2-21
2.3.6.4	Functions	2-22
2.3.6.4.1	Trigonometric Functions	2-22
2.3.6.4.2	Angle Functions	2-24
2.3.6.4.3	Maximum And Minimum Functions	2-25
2.3.6.4.4	Sign Function	2-26
2.3.6.4.5	Distance Function	2-26
2.3.6.4.6	TDISTF Function	2-28
2.3.6.4.7	Absolute Value Function	2-29
2.3.6.4.8	Square Root Function	2-29
2.3.6.4.9	Cube Root Function	2-30
2.3.6.4.10	Natural Logarithm Function	2-30
2.3.6.4.11	Common Logarithm Function	2-30
2.3.6.4.12	Exponent Function	2-31
2.3.6.4.13	Length Of A Geometric Entity Function	2-31

TABLE OF CONTENTS

2.3.6.4.14	Length Of A Three Dimensional Curve Function	2-32
2.3.6.4.15	Integer Function	2-33
2.3.6.4.16	Dot Product Function	2-33
2.3.6.4.17	Canonical Form Function	2-34
2.3.6.4.18	Num Function	2-39
2.3.6.4.19	Type Function	2-40
2.3.6.4.20	XTYPE Function	2-41
2.3.6.4.21	Vocabulary Function	2-42
2.3.7	Text Control	2-43
2.3.7.1	Text String	2-43
2.3.7.2	Text Variable	2-43
2.3.7.3	Text Element	2-44
2.3.7.4	Sub-String Clause	2-45
2.3.7.5	Format Function	2-45
2.3.7.6	Text Functions	2-47
2.3.7.6.1	The LNTH Function	2-47
2.3.7.6.2	The FINDEX Function	2-48
2.3.7.6.3	The RINDEX Function	2-48
2.3.7.6.4	The STRCMP Function	2-49
2.3.7.7	The TEXTF Function	2-50
2.3.7.8	Text Entity Removal	2-50
2.3.7.9	Text String/Variable Evaluation	2-51
2.3.8	Expressions	2-52
2.3.8.1	Scalar Expressions	2-53
2.3.8.2	Geometric Expressions	2-53
2.3.8.3	Nested Expressions	2-54
2.3.9	Assignment Statements	2-55
2.3.9.1	Scalar Assignment Statements	2-55
2.3.9.2	Geometric Assignment Statements	2-56
2.3.9.3	Macro Assignment Statements	2-59
2.3.9.4	Matrix Assignment Statements	2-59
2.3.9.5	Text String Assignment Statements	2-59
2.4	File Specifications	2-60
3	GEOMETRY EXPRESSIONS	3-1
CIRCLES		3-4
3.1	CIRCLE Expressions	3-7
3.1.1	A Circle By Center Coordinates And Radius	3-9
3.1.2	A Circle By Three Points On Its Circumference	3-11
3.1.3	A Circle By A Combination Of Points, Vectors And Point-Vectors	3-12

TABLE OF CONTENTS

3.1.4	A Circle Tangent To A Line, A Point On Its Circumference And A Radius	3-15
3.1.5	A Circle Tangent To Two Intersecting Lines And A Radius	3-17
3.1.6	A Circle Tangent To Two Circles And A Radius	3-19
3.1.7	A Circle Tangent To A Circle And A Line With A Specified Radius	3-21
3.1.8	A Circle Tangent To Three Lines	3-23
3.1.9	A Circle Tangent To A Line And A Curve/B-Spline With A Specified Radius And A Near Point/Point-Vector	3-25
3.1.10	A Circle Tangent To A Circle And A Curve/Spline, With A Specified Radius And A Near Point/Point-Vector	3-27
3.1.11	A Circle By A Point On Its Circumference, A Tangent Circle And A Specified Radius	3-29
3.1.12	A Circle By A Center At A Point/Point-Vector And Tangent To A Line	3-31
3.1.13	A Circle With Center At A Point/Point-Vector With A Specified Radius	3-33
3.1.14	A Circle With Center At A Point/Point-Vector And A Point/Point-Vector On Its Circumference	3-34
3.1.15	A Circle With Center At A Point/Point-Vector And Tangent To A Circle	3-36
3.1.16	A Circle By Two Points/Point-Vectors Or A Point And A Point-Vector On Its Circumference With A Specified Radius	3-38
3.1.17	A Circle Delta X, Y, Z From Another Circle	3-41
3.1.18	A Circle Offset From Another Circle	3-42
3.1.19	A Circle By Canonical Form	3-43
3.1.20	Multiple Fillet	3-45
	CURVES	3-47
3.2	CURVE Expressions	3-51
3.2.1	A Spline/Curve [FIT] Through A Series Of Combination Of Point-Vectors, And/Or Points With Optional Slope Control Vectors, With Optional Input Entities Skipped By A Constant Step	3-52
3.2.2	A Curve That Approximates A Conic Section Using Five Points	3-58
3.2.3	A Curve That Approximates A Conic Section With Three Points And One Point Slope At Either End	3-59
3.2.4	A Curve That Approximates A Conic Section Using Two Point Slopes At Both Ends And One Additional Point/Point-Vector In Between	3-61

TABLE OF CONTENTS

3.2.5	A Curve As A Composite Collection Of Lines, Circles And/Or Curves	3-63
3.2.6	A Spline/Curve Translates From A Wire Entity At A Delta Distance	3-66
3.2.7	A Spline/Curve Offset From A Wire Entity In A Direction At A Distance	3-68
3.2.8	A Composite Curve Offset From A Composite Curve In A Direction At A Distance	3-71
3.2.9	A Spline/Curve Defined As The Intersection Of Two Surfaces With An Optional Near Point Or A Point-Vector	3-73
3.2.10	A Spline/Curve Defined As The Intersection Of A Surface And A Plane With An Optional Near Point Or Point/Vector	3-76
3.2.11	A Spline/Curve As The Edge Of A Surface Or Any U Or V Offset On The Surface	3-79
3.2.12	A Spline As A 2-Dimensional Outline Of A Group Of Surfaces	3-82
3.2.13	A Spline Or S-Spline Extracted From A Revolved Surface	3-86
3.2.14	A Spline Or S-Spline Extracted From A Trimmed Surface	3-88
3.2.15	A Spline Or S-spline Obtained By Projecting A Curve On A Surface At An Angle	3-91
3.2.16	A Spline Or S-Spline Obtained By Projecting A Curve On A Surface	3-93
3.2.17	Isoparametric Surface-Spline On A Surface	3-100
3.2.18	Surface-Spline On A Surface Intersecting The Surface With Another Surface Or Plane With An Optional Near Point/Point-Vector	3-102
3.2.19	A Surface-Spline As A Composite Collections Of Surface-Splines	3-104
3.2.20	A Surface-Spline As An Offset Of A Surface-Spline	3-105
 LINES		3-106
3.3	LINE Expressions	3-109
3.3.1	A Line As Its Coordinates	3-110
3.3.2	A Line Between A Point/Point-Vector And A Point/Point-Vector	3-111
3.3.3	A Line From The Canonical Form Of A Point-Vector	3-112
3.3.4	A Line From A Point/Point-Vector And Parallel To A Line/Point-Vector	3-113
3.3.5	A Line From A Point/Point-Vector And Perpendicular To A Line/Point-Vector	3-115
3.3.6	A Line Parallel To A Line At A Distance	3-117
3.3.7	A Line Defined By The Forward Sense Of The Current Tool Position	3-118
3.3.8	A Line Through A Point/Point-Vector And Tangent To A Circle	3-120
3.3.9	A Line Tangent To Two Circles	3-122

TABLE OF CONTENTS

3.3.10	A Line Parallel To the X Or Y Axis At A Distance	3-124
3.3.11	A Line Through A Point/Point-Vector At An Angle To The X-Axis Or the Y-Axis Or A Line Or A Point-Vector	3-126
3.3.12	A Line From A Point/Point-Vector And Perpendicular Or T angent To A Curve Near A Point/Point-Vector	3-129
3.3.13	A Line As The Intersection Of Two Planes	3-132
3.3.14	A Line Tangent To A Circle At An Angle To The X-Axis Or The Y-Axis Or A Line Or A Point-Vector	3-133
3.3.15	A Line As The X, Y and Z Delta Distance From A Line Or A Point-Vector	3-135
3.3.16	A Line At The Intersection Of A Plane And The XY-Plane	3-136
PATERN	3-137
3.4	PATERN Expressions	3-140
3.4.1	A Linear Pattern Between Two Points Or Two Point-Vectors	3-141
3.4.2	A Linear Pattern Along A Vector At A Start Point/Point-Vector With Distance Between Individual Elements Equal To The Magnitude Of The Vector	3-143
3.4.3	A Linear Pattern Along A Vector With A Start Point/Point-Vector And Specific INCR Clauses	3-145
3.4.4	A Pattern Of Points Equally Spaced Around A Circle Between Two Specified Angles	3-149
3.4.5	A Pattern Of Points Around A Circle With A Start Angle And Specific INCR Clauses	3-153
3.4.6	A Pattern Parallel To A Pattern In The Direction Of A Vector/Point-Vector	3-158
3.4.7	A Pattern That Is Parallel To A Pattern In the Direction Of A Vector/Point-Vector With Specified INCR Clauses	3-160
3.4.8	A Pattern Obtained By Projecting A Pattern On A Surface	3-164
3.4.9	A Pattern That Randomly Combines Either Points And Point-Patterns, Or Point-vectors And Point-Vector-Patterns	3-170
PLANES	3-172
3.5	PLANE Expressions	3-174
3.5.1	A Plane By The Normal Vector Components And An Offset Value With An Optional Display Point/Point-Vector	3-175
3.5.2	A Plane Through Three Geometric Entities In Any Combination Of Points Or Point-Vectors With An Optional Display Point/Point-Vector	3-177

TABLE OF CONTENTS

3.5.3	A Plane Through A Point/Point-Vector And Parallel To Two Geometric Entities In Any Combination Of Points Or Point-Vectors With An Optional Display Point/Point-Vector	3-179
3.5.4	A Plane Through A Point/Point-Vector And Parallel To A Plane With An Optional Display Point/Point-Vector	3-181
3.5.5	A Plane Parallel To A Plane/Planar-Surface At A Distance With An Optional Display Point/Point-Vector	3-182
3.5.6	A Plane Perpendicular To A Point Vector And Through Its Origin With An Optional Display Point/Point-Vector	3-184
3.5.7	A Plane Through A Point/Point-Vector And Perpendicular To A Vector/Point-Vector With An Optional Display Point/Point-Vector	3-185
3.5.8	A Plane Perpendicular To A Plane And Through Two Entities Which Can Be Points And/Or Point-Vectors With An Optional Display Point/Point-Vector	3-187
3.5.9	A Plane Through A Point Or Point-Vector And Perpendicular To Two Intersecting Planes With An Optional Display Point/Point-Vector	3-188
3.5.10	A Plane Which Contains A Line And Perpendicular To A Plane With An Optional Display Point/Point-Vector	3-189
3.5.11	A Plane By Cloning Itself With An Optional Display Point/Point-Vector	3-191
3.5.12	A Plane From A Planar Surface With Optional Display Point/Point-Vector	3-192
3.5.13	An Optimal Plane From A Curve or A Surface	3-193
POINTS	3-195	
3.6	POINT Expressions	3-198
3.6.1	A Point By Its Rectangular Coordinates	3-199
3.6.2	A Point At The Origin Of A Point-Vector In Three-Dimensional Space	3-201
3.6.3	A Point At The Current Cutter Location Or Tool End	3-202
3.6.4	A Point At the Center Of A Circle	3-203
3.6.5	A Point At the Center Of Gravity Of A Curve Or Surface	3-204
3.6.6	A Point By Its X, Y and Z Delta Displacements Relative To A Point Or A Point-Vector	3-206
3.6.7	A Point By a Vectorial Displacement	3-208
3.6.8	A Point At the Intersection Of Two Lines, Two Point-Vectors, Or One Line And One Point-Vector	3-209
3.6.9	A Point At the Intersection Of A Line And A Plane	3-212
3.6.10	A Point At The Intersection Of A Point-Vector And A Plane	3-213

TABLE OF CONTENTS

3.6.11	A Point At The Intersection Of Three Planes	3-214
3.6.12	A Point At The Intersection Of A Circle/Plane And A Curve, Nearest To A Referenced Point Or The Origin Of A Referenced Point-Vector	3-215
3.6.13	All Points At The Intersection Of A Curve And, A Curve Or A Line Or A Point-Vector Or A Circle Or A Plane, Or An Intersection Point Nearest To A Referenced Point Or The Origin Of A Referenced Point-Vector	3-218
3.6.14	A Point At The Intersection Of Two 3-D Wire Entities In 3-Dimensional Space	3-221
3.6.15	A Point At The End Of A Line Or A Circle Or A Point-Vector	3-223
3.6.16	A Point At The Intersection Of A Circle/Line/Point-Vector And A Circle	3-225
3.6.17	A Point On A Circle At An Angle With The X-axis	3-227
3.6.18	A Point On A Circle At An Angle With A Line Passes Through A Point Or The Origin Of A Point-Vector And The Circle	3-229
3.6.19	A Point On A Circle/Curve/Line At A Distance	3-231
3.6.20	A Point On A Wire Frame Entity At A Specified U Or Percent Value	3-234
3.6.21	A Point On A Surface At A Specified UV Or Percent Values	3-236
3.6.22	A Point In A Pattern	3-238
3.6.23	A Point Obtained By Projecting A Point/Point-Vector Onto A Surface/Solid Along An Optional Vector/Point-Vector And An Optional Near Point/Point-Vector	3-240
3.6.24	PODDEF And PODPTS	3-243
3.6.24.1	PODDEF Expression	3-243
3.6.24.2	PODPTS	3-245

POINT-VECTORS 3-248

3.7	POINT-VECTOR Expressions	3-251
3.7.1	A Point-Vector By Its Rectangular Coordinates And Its Vector Components	3-252
3.7.2	A Point-Vector From One Point To Another	3-254
3.7.3	A Point-Vector From A Point And A Vector	3-255
3.7.4	A Point-Vector In The Direction From The Current Tool End Up The Current Tool Axis	3-256
3.7.5	A Point-Vector In A Direction From The Current Tool End Toward The Current Forward Motion	3-257
3.7.6	A Point-Vector Along A Line	3-258
3.7.7	A Point-Vector From The Center Of A Circle	3-259
3.7.8	A Point-Vector At The Center Of Gravity Of A Curve Or A Surface Perpendicular To The Optimal Plane	3-260

TABLE OF CONTENTS

3.7.9	A Point-Vector From A Point On A Curve Along A Specified Distance And Tangent To The Curve	3-262
3.7.10	A Point-Vector From An Existing Point And In The Direction Of The Slope Of A Curve	3-263
3.7.11	A Point-Vector As The Directed Distance From A Point Normal To A Surface	3-264
3.7.12	A Point-Vector Perpendicular To A Plane	3-265
3.7.13	A Point-Vector Through A Point Or Point-Vector And Perpendicular To A Line, Circle, Curve, Plane Or Surface	3-266
3.7.14	A Point-Vector Through A Point Or Point-Vector And Tangent To A Line, Circle, Curve	3-270
3.7.15	A Point-Vector As The Intersection Of Three Planes	3-272
3.7.16	A Point-Vector As The Sum Or Difference Of A Point-Vector And A Vector	3-274
3.7.17	A Point-Vector As The Cross Product (Normal) Of A Point-Vector And A Vector	3-276
3.7.18	A Point-Vector Multiplied By A Scalar	3-278
3.7.19	A Point-Vector Offset By A Scalar	3-279
3.7.20	A Point-Vector Which Is the Result Of Unitizing Another Point-Vector	3-280
3.7.21	A Point-Vector As A Point-Vector In A Pattern Of Point-Vectors	3-281
3.7.22	A Point-Vector Normal To A Surface Obtained By Projecting A Point/Point-Vector Onto The Surface Along An Optional Vector/Point-Vector And An Optional Near Point/Point-Vector	3-282
3.7.23	A Point-Vector From A Point On A Wire Frame Entity At A Specified UV or Percent Values And Tangent To The Wire Frame Entity	3-284
3.7.24	A Point-Vector Normal To A Surface Obtained By Specifying The UV Or Percent Values	3-286
3.7.25	A Point-Vector Along The Axis of A Surface Of Revolution And At The Center Of The Base	3-288
SURFACES		3-289
3.8	SURFace Expressions	3-292
3.8.1	A Surface Constructed From Two Geometric Entities Using System-Generated Slope Constraints	3-296
3.8.2	A Surface Constructed Through A Series Of Geometries With Slope Control	3-300
3.8.3	A Fillet Surface Tangent To Two Intersecting Planes Using Point Or Point-Vector Boundaries	3-305
3.8.4	A Fillet Surface Between Two Surfaces Or A Surface And A Plane	3-307

TABLE OF CONTENTS

3.8.5	A Surface Defined By A Series Of Geometric Curves, Lines, Or Circles	3-314
3.8.6	A Surface Offset From A Surface In A Direction At A Distance	3-316
3.8.7	A Surface Obtained By Untrimming A Trimmed Surface	3-318
3.8.8	A Surface As A Collection Of A Number Of Other Surfaces	3-319
3.8.9	A Trimmed Surface Created By Specifying The Outer Boundary With Optional Inner Boundary(ies)	3-321
3.8.10	A Surface Of Revolution Constructed From A Single Wireframe Entity With Optional Starting And Ending Angle	3-325
3.8.11	A Surface Constructed From Four Edge Entities	3-328
VECTORS		3-330
3.9	VECTOR Expressions	3-333
3.9.1	A Vector By Its Components Along The X, Y And Z Axes	3-334
3.9.2	A Vector As The Direction From One Point/Point-Vector To Another Point/Point-Vector	3-335
3.9.3	A Vector From A Point-Vector	3-337
3.9.4	A Vector In The Direction Of The Current Forward Sense Of The Tool	3-338
3.9.5	A Vector In The Direction Up From The Current Tool Axis	3-339
3.9.6	A Vector Which Is The Result Of Unitizing A Vector/Point-Vector ..	3-340
3.9.7	A Vector From A Vector/Point-Vector Multiplied By A Scalar	3-341
3.9.8	A Vector As The Cross Product (Normal) Of Any Combination Of Two Vectors/Point-Vectors	3-342
3.9.9	A Vector As The Sum or Difference Of Any Combinations Of Two Vectors Or Point-Vectors	3-345
3.9.10	A Vector Perpendicular To A Plane	3-347
3.9.11	A Vector As The Intersection Of Two Planes	3-348
3.9.12	A Vector As The Directed Distance From A Point/Point-Vector Normal To A Surface	3-349
3.9.13	A Vector Tangent To A Curve With A Near Point/Point-Vector Specified	3-350
3.9.14	A Vector In The XY-Plane At An Angle With A Line	3-352
MATRIX		3-354
3.10	MATRIX Expressions	3-356
3.10.1	A Matrix In Terms Of Its Canonical Form	3-357
3.10.2	A Matrix As A Translation In X, Y And Z	3-359
3.10.3	A Matrix As A Rotation In A Specified Coordinate Plane	3-360
3.10.4	A Matrix As Scale Factors	3-362
3.10.5	A Matrix As The Product Of Two Other Matrices	3-366
3.10.6	A Matrix As The Inverse Of A Matrix	3-368

TABLE OF CONTENTS

3.10.7	A Matrix As A Mirror Image Relative To A Specified Plane	3-369
3.10.8	A Matrix In Terms Of A Point And A Combination Of Two Entities Which Can Be Vector Or Point-Vector	3-370
3.10.9	A Matrix In Terms Of A Point-Vector And Another Point-Vector Or Vector	3-372
3.10.10	A Matrix Converts One Coordinate System To Another Coordinate System	3-373
MULTIPLE ENTITIES TYPE GEOMETRY STATEMENTS		3-374
3.11.1	Creating Boundary Contours Or Their Components At The Intersection Of Groups Of Surfaces And A Plane, A Surface Or A Z-value.	3-374
3.11.2	Extracting Component(s) From A Composite Curve	3-377
3.12	[label] = SOLID/OUT,solid,ALL [, NUM,ncomp] id-list	3-378
3.13	[label] = SURF/OUT,geo,ALL [, NUM,ncomp] id-list	3-379
SOLID		3-380
3.14	Solids Expressions	3-383
3.14.1	A Solid Box By Two Opposite Corner Points	3-386
3.14.2	A Solid Box By A Center Point, Length, Width And Height	3-389
3.14.3	A Solid Box As An XYZ Bounding Box Around A List Of Geometry Or Generated Motion	3-391
3.14.4	A Composite Solid Created By A List Of Solids/Surfaces	3-393
3.14.5	A Solid Cone By Two Points And Two Radius	3-395
3.14.6	A Solid Cone By A Point-Vector, Height And Two Radius	3-396
3.14.7	A Solid Cylinder By A Circle And Height	3-398
3.14.8	A Solid Cylinder By Two End Points And Radius	3-399
3.14.9	A Solid Cylinder By A Point-Vector, Height And Radius	3-400
3.14.10	A Solid Extrusion By Lifting A Closed Planar Curve With Distance Along A Vector	3-402
3.14.11	A Solid Extrusion By Following the Contour Of The Part	3-404
3.14.12	A Solid By Revolving A Planar Curve Around A Point-Vector	3-406
3.14.13	A Solid Sphere By A Circle	3-408
3.14.14	A Solid Sphere By A Center Point And Radius	3-409
3.14.15	A Solid By Importing An External STL File	3-410
3.14.16	A Solid Torus By Two Circles	3-412
3.14.17	A Solid Torus By A Circle And A Radius	3-414
3.14.18	A Solid Torus By A Point-Vector And Two Radius	3-415
3.14.19	A Solid By Loading An External Stock File	3-417
3.14.20	Save Defined Solids As An External Stock File	3-418

TABLE OF CONTENTS

3.14.21	Save Defined Solids As An External STL File	3-419
ANNOTATIONS		3-420
3.15	ANOTE	3-420
3.16	Annotation Attributes	3-427
3.17	Miscellaneous.	3-433
3.17.1	Fonts Support By Annotation	3-433
3.17.2	Obtain Annotation Parameters	3-454
4 GEOMETRY CONTROL STATEMENTS		4-1
4.1	ANALYZ/ sf1 [[, THRU], sfn] [. . .] [, sfm [[, THRU], sfmn]]	4-1
4.2	CANON	4-1
4.3	Geometry Filter Control	4-2
4.3.1	CLIPF Function	4-2
4.3.2	COLF Function	4-3
4.3.3	FILTER Function	4-3
4.3.4	LAYF Function	4-4
4.3.5	MARKF Function	4-4
4.4	CLONE	4-4
4.5	Creating New Entities From Existing Entities	4-5
4.6	d1 = DATA / element [delimiter element [. . .]]	4-5
4.7	DATA Statement That Reads From A Comma Or Tab Delimited File ..	4-7
4.8	label = DECOMP/ALL [, nent [, RESET]]	4-10
	instance1 [. . .] PLUS	
4.9	DEFNAM/ geo_typ1, name1 [, INDEX] [, geo_typ2, name2, . . .]	4-11
4.10	GENPTS	4-12
4.10.1	GENPTS/ [POINT,] [TE ,] \$	
	ARC	
	DS	
	PS	
	num-points-scalar, reserved-point-array \$	
	[, reser-vector-1-array [, reser-vector-2-array]] \$	
	[, NOW]	4-12
	NEXT	
4.10.2	GENPTS/ PNTVEC [, type] [, TE], n, pv-array [, vec-array] \$	
	ARC	
	DS	
	[, NOW]	4-14
	NEXT	

TABLE OF CONTENTS

4.10.3	GENPTS/ NOMORE	4-16
4.11	CHKPTS/ tolerance, maxdp, maxang [, MACRO, m1 [, PS]]	4-18
	DS	
	CHKPTS/ NOMORE	4-18
	CHKPTS/ REMOVE	4-18
4.12	MODSYS/ matrix-id	4-19
	MODSYS/ NOMORE	4-19
4.13	MOVE	4-20
4.14	OBTAIN	4-20
4.14.1	OBTAIN/ ATTRIB, geometry-id	4-21
4.14.2	OBTAIN/ geometry-id	4-21
4.14.3	OBTAIN/ data-id	4-21
4.14.4	OBTAIN/ "Modal Setting"	4-22
4.14.5	OBTAIN/ "Tool End Point And Tool Axis Vector"	4-24
4.15	label = PLACE/ [symlib,] sym, AT, pt [, SCALE, s]	\$
	[, ROTATE, r]	4-24
4.16	PRINT	4-25
4.16.1	PRINT/ ALL	4-25
4.16.2	PRINT/ geometry	4-25
4.16.3	PRINT/ SCALAR	4-26
4.16.4	PRINT/ 0	4-26
4.16.5	PRINT/ OFF, IN	4-26
4.16.6	PRINT/ ON, IN	4-26
4.17	REDEF	4-26
4.17.1	REDEF/ circle, line-1, line-2, modifier	4-26
4.17.2	REDEF/ curve, CLOSE	4-26
	OPEN	
4.17.3	REDEF/ curve1 [,near-pt1], EDGE, curve2 [,near-pt 2]	4-27
	modifier	
	modifier	
4.17.4	REDEF/ curve1 [, cv-name] [,near-pt1], EDGE, curve2	\$
	modifier	
	[, near-pt2], EDGE, curve3 [,near-pt3]	4-29
	modifier	
	modifier	
4.17.5	REDEF/ surface	4-31
4.17.5.1	REDEF/ surface, CLOSE,0	4-31
	1	
4.17.5.2	REDEF/ surface, OPEN, 0	4-31

TABLE OF CONTENTS

4.17.5.3	REDEF/ surface, [PARAMS, [SWAP,] [, REVERS, 0 1 NORMAL NORMAL [[,] PARLEM, 0][[,] BASE] 1 FACE -1	\$ \$
		4-32
4.17.5.4	[sfn =] REDEF/ surface	4-35
4.17.5.5	[sfn =] REDEF/ sf1 [, OUT, cv1] [, modifier] [, IN, cv2, . . .] sf2 pl1	4-38
4.17.5.6	[sfn =] REDEF/ [pl1,] OUT, cv1 [, IN, cv2, . . .] sf1	4-40
4.17.5.7	[sfn =] REDEF/ sf1, REMOVE, i1 [[, THRU], i2, . . .] [0,] ALL	4-42
4.17.6	REDEF/ geo	4-44
4.18	REFSYS	4-44
4.18.1	REFSYS/ matrix	4-45
4.18.2	REFSYS/ NOMORE	4-45
4.19	RENAME	4-45
4.20	REMOVE	4-45
4.21	RESERV	4-47
4.21.1	Inclusive Subscripted Variables	4-48
4.22	REVERS	4-49
4.22.1	REVERS/ data-statement	4-49
4.22.2	REVERS/ geometry	4-49
4.23	sym1 = SYMBOL/ [AT, point,] geometry-list	4-51
4.1	ZSURF	4-52
4.1.1	ZSURF/ scalar	4-52
4.1.2	ZSURF/ plane	4-53
4.1.3	ZSURF/ NOMORE	4-53
5	UNIBASES	5-1
5.1	UBFN	5-1
5.1.1	UBFN/ Unibase file name	5-1
5.1.2	UBFN/ CLOSE	5-1
5.2	ASCII Unibase Files	5-2
5.3	GET	5-2
5.3.1	GET/ alphanumeric-character-string*	5-3
5.3.2	GET/ entity-id, AS, new-id	5-3
5.3.3	GET/ entity-id [, THRU, entity-id]	5-3
5.3.4	GET/ type	5-4

TABLE OF CONTENTS

5.3.5	GET/[RENAME,] LAYER, n1 [[, THRU], nn] [. . .] [,m [, THRU], mn] [, OFFSET, k]	\$ 5-5
5.3.6	GET/ ALL [, OFFSET, n]	5-5
5.3.7	GET/ RENAME,	5-6
5.4	LOADU	5-7
5.5	PUT	5-8
5.5.1	PUT/ alphanumeric-character-string*	5-9
5.5.2	PUT/ entity-id, AS, new-id	5-9
5.5.3	PUT/ entity-id [, THRU, entity-id]	5-9
5.5.4	PUT/ type	5-10
5.5.5	PUT/ LAYER, n1 [[, THRU], nn] [. . .] [, m [, THRU], mn]	5-11
5.5.6	PUT/ ALL	5-12
5.6	SAVEU	5-13

6 MOTION CONTROL STATEMENTS 6-1

CUTTER AND TOOL AXIS STATEMENTS 6-2

6.1	CUTTER	6-2
6.1.1	Mill Style Cutter	6-2
6.1.1.1	CUTTER/ diameter [, corner-radius [, height [, side-angle]]]	6-2
6.1.1.2	CUTTER/ diameter, corner-radius, height, side-radius, Z-height [, flat-angle]	\$ 6-4
6.1.1.3	CUTTER/ d, r, e, f, a, b, h	6-6
6.1.1.4	CUTTER/ PSEUDO, dia, corner-radius, height, side-angle	6-7
	CUTTER/ PSEUDO, dia, corner-radius, height, side-radius, Z-height [, flat-angle]	\$ 6-7
6.1.1.5	CUTTER/ DISPLAY, dia, corner-radius, height, side-angle	6-7
6.1.1.6	CUTTER/ DISPLAY, dia, corner-radius, hgt, side-radius, Z-height [, flat-angle]	\$ 6-8
6.1.1.7	CUTTER/ DISPLAY, surface cv [, pv] pt-list solid [symlib,] symbol	6-8
6.1.2	Blade Style Cutter	6-9
6.1.2.1	CUTTER/ BLADE, width, chisel, height, angle	6-9
6.1.2.2	CUTTER/ DISPLAY, curve pt-list solid [symlib,] symbol	6-9
6.1.3	Lathe Style Cutter	6-10

TABLE OF CONTENTS

6.1.3.1	CUTTER/ LATHE, radius, diameter, height [, angle [, mount-angle]]	\$ 6-10
6.1.3.1.1	Square Insert	6-11
6.1.3.1.2	Diamond Insert	6-11
6.1.3.1.3	Triangle Insert	6-12
6.1.3.1.4	Round Insert	6-12
6.1.3.2	CUTTER/ LATHE, radius, width, height, 0, length	6-12
6.1.4	CUTTER/ DISPLAY, SHANK,	6-13
6.1.4.1	CUTTER/ DISPLAY, SHANK, diameter, height [, side-angle] [,ofs] [, CUTTER]	\$ 6-14
	HOLDER	
6.1.4.2	CUTTER/ DISPLAY, SHANK,surface cv [, pv] pt-list solid [symlib,] symbol [, CUTTER]	[, ofs] \$ 6-15
	HOLDER	
6.1.4.3	CUTTER/ DISPLAY, SHANK, width, length, depth, y-offset [, CUTTER]	\$ 6-16
	HOLDER	
6.1.4.4	CUTTER/ DISPLAY, SHANK,curve pt-list solid [symlib,] symbol [, OFFSET, zatt, zdep] [, CUTTER]	[, x, y] \$ 6-17
	HOLDER	
6.1.5	CUTTER/ DISPLAY, HOLDER,	6-18
6.1.5.1	CUTTER/ DISPLAY, HOLDER, diameter, height [, side-angle] [,ofs]	\$ 6-18
6.1.5.2	CUTTER/ DISPLAY, HOLDER,surface cv [, pv] pt-list solid [symlib,] symbol	[, ofs] \$ 6-19
6.1.5.3	CUTTER/ DISPLAY, HOLDER, width, length, depth, y-offset	6-20
6.1.5.4	CUTTER/ DISPLAY, HOLDER,curve pt-list solid [symlib,] symbol [, OFFSET, zatt, zdep]	[, x, y] \$ 6-21
6.1.6	CUTTER/ TOOL, [lib,] tool [, params]	6-22
6.1.7	CUTTER/ READ, [lib,] tool [, params]	6-22

TABLE OF CONTENTS

6.1.8	CUTTER/PROFIL,"lib-name"	6-23
6.1.9	CUTTER/ DISPLAY,PART	6-23
	ALL	
6.1.10	CUTTER/ DISPLAY, MOVE,ON	6-23
	OFF	
6.1.11	CUTTER/ DISPLAY, SHADE, ON [, CUTTER]	6-24
	OFF SHANK	
	HOLDER	
6.1.12	Tool Profile Description File	6-24
6.2	TLAXIS	6-28
6.2.1	TLAXIS/ I, J, K [, NORMAL] [, modify-clause]	6-30
6.2.2	TLAXIS/ vector [, NORMAL] [, modify-clause]	6-30
	point-vector	
6.2.3	TLAXIS/ 1	6-30
	SAME [, NORMAL] [, modify-clause]	
6.2.4	TLAXIS/ NORMAL, PS [, surface]	\$
	NORMPS plane	
	[, PERPTO, [LAST,] vector][, modify-clause]	6-30
	point-vector	
6.2.5	TLAXIS/ ATANGL, angle, PS [,surface] [, CLDIST, dist]	\$
	plane	6-31
	[, CONTCT] [, PERPTO,[LAST,]vector]	\$
	point-vector	
	[, modify-clause]	6-31
6.2.6	TLAXIS/ TANTO, DS, height [, PS,surface]	\$
	plane	
	[, PERPTO,[LAST,]vector][, modify-clause]	6-32
	point-vector	
6.2.7	TLAXIS/ TANTO, DS, height, FAN [, CENTER,OFF]	\$
	ON	
	AUTO	
	[, SMOOTH [, d, r]][, modify-clause]	6-32
6.2.8	TLAXIS/ TANTO, DS, height, PARELM [, modify-clause]	6-33
6.2.9	TLAXIS/ COMBIN, height, [CENTER,OFF ,]	\$
	ON	
	AUTO	
	[PS, surface,] leave-dist [, approach-dist]	\$
	plane	
	[, SMOOTH [, d, r]][, modify-clause]	6-34
6.2.10	TLAXIS/ COMBIN, height, PARELM, [CENTER,OFF ,]	\$
	ON	
	AUTO	
	leave-dist [, approach-dist][, modify-clause]	6-36

TABLE OF CONTENTS

6.2.11	TLAXIS/ [. . . ,] [RIGHT, right-angle] [[,] FWD, fwd-angle]	6-36
6.2.12	TLAXIS/ THRU, point	6-37
6.2.13	TLAXIS/ THRU, curve [, dist]	6-38
6.2.14	TLAXIS/ [. . . ,] GUIDE, curve, [, CONTCT] [, TLLFT] \$ OFFSET TLRGT TLON	
	[, thick]	6-39
6.2.15	TLAXIS/ [. . . ,] GOUGCK, ON	6-42
	OFF	
6.2.16	TLAXIS/ INTERP,vector [, SMOOTH [, d, r]]	6-43
	point-vector	
	i, j, k	
6.2.17	TLAXIS/ [. . . ,], LOCK,OFF	6-44
	END	
	ds1 [, LINEAR] [, ds2 [,INTERP]] \$ RADIUS FAN	
	[, OMIT]	6-44
	RETAIN	

POINT TO POINT MOTION STATEMENTS 6-45

6.3	FROM	6-45
6.3.1	FROM/ x, y [, z] [, vector] [, feedrate]	6-45
6.3.2	FROM/ point-vector [, feedrate] point [, vector]	6-45
6.4	GO	6-45
6.4.1	One Surface GO	6-47
6.4.2	Two Surface GO	6-48
6.4.3	Three Surface GO	6-49
6.5	The SRFVCT Statement	6-51
6.6	GODLTA	6-55
6.6.1	GODLTA/ distance [, feedrate]	6-56
6.6.2	GODLTA/ delta-x, delta-y, delta-z [, feedrate]	6-56
6.6.3	GODLTA/ plane [, feedrate]	6-56
6.6.4	GODLTA/ vector [, feedrate]	6-56
6.6.5	GODLTA/ point-vector [, feedrate]	6-56
6.6.6	GODLTA/ surface [, feedrate]	6-56
6.7	GOTO	6-57
6.7.1	GOTO/ x, y [, z [, i, j, k] [, feedrate]]	6-57
6.7.2	GOTO/ point [, vector] [, feedrate]	6-57
6.7.3	GOTO/ point-vector [, feedrate]	6-57
6.7.4	GOTO/ patern-id [, INVERS] [, CONST] [, AVOID-clause] \$ [, RETAIN-clause] [, OMIT-clause]	6-58

TABLE OF CONTENTS

6.8	NOPS	6-59
CONTINUOUS MOTION STATEMENTS		6-60
6.9	PART SURFACE	6-60
6.9.1	AUTOPS	6-60
6.9.2	PSIS	6-60
6.10	DRIVE SURFACE	6-61
6.11	TOOL CONDITIONS	6-62
6.12	Combined Drive And Part Surface	6-62
6.13	CHECK SURFACE	6-63
6.13.1	Multiple Check Surfaces	6-66
6.13.2	Multiple Intersection Motion Statements	6-69
6.13.3	Point As A Check Surface	6-70
6.13.4	Check Surface At A Near Point	6-71
6.13.5	Check Surface Tangent To A Surface	6-71
6.14	User Specified U And V Values For Curves And Surfaces	6-73
6.14.1	Automatic Use Of Previous U And V Values During Continuous Path Motion	6-74
6.14.2	Driving A Composite Curve With Components Not Tangent To Each Other	6-75
6.15	GOBACK	6-76
6.16	GODOWN	6-76
6.17	GOFWD	6-77
6.18	GOLFT	6-78
6.19	GORGT	6-78
6.20	GOUP	6-79
6.21	INDIRP	6-80
6.21.1	INDIRP/ point	6-80
6.21.2	INDIRP/ x, y, z	6-80
6.22	INDIRV	6-80
6.22.1	INDIRV/ vector	6-81
6.22.2	INDIRV/ x, y, z	6-81
6.22.3	INDIRV/ Point-Vector	6-81
6.23	TLLFT	6-81
6.24	TLON	6-81
6.25	TLONPS	6-82
6.26	TLOFPS	6-82
6.27	TLRGRT	6-83
AUTOMATIC MOTION ROUTINE STATEMENTS		6-84
6.26	FMILL	6-84
6.27	GOFWDA	6-105

TABLE OF CONTENTS

6.27.1	GOFWDA For Driving A Single Composite Curve	6-105
6.27.2	GOFWDA For Driving Multiple Entities	6-106
6.28	POCKETing	6-111
6.28.1	POCKET	6-111
6.28.2	ADVANCED POCKET	6-113
6.28.2.1	POKMOD	6-114
6.28.2.1.1	Helix Output as CYCLE in POKMOD Command	6-148
6.28.2.1.2	Helix Output as Text String in POKMOD Command	6-149
6.28.2.2	POCKET	6-151
6.28.2.3	Save Pocket Routine	6-167
6.29	PROFIL	6-168
6.29.1	PROFIL - 2D/3D Curve Cutting	6-168
6.29.2	PROFIL - Annotation Engraving	6-195
6.30	RMILL	6-203
6.31	SMILL	6-220
6.32	Waterline Roughing	6-230
6.32.1	Waterline Roughing Examples	6-255
6.32.1.1	Example Using The FIT (Calculated Box) Stock Specification	6-255
6.32.1.2	Example Using The PART (Calculated Contour) Stock Specification	6-256
6.32.1.3	Example Using The OMIT (Included) Stock Specification	6-257
6.32.1.4	Example Using The IN (Included, Ignore Outer Stock Specification	6-258
6.32.1.5	Example Machining Overhanging Surfaces	6-259
6.33	VoluMill Pocketing Motion	6-260
6.33.1	VMPMOD	6-260
6.33.2	VMPOCK	6-274

MISCELLANEOUS MOTION STATEMENTS **6-283**

6.32	ARCSLP/ FILLET	6-283
6.32.1	ARCSLP/ FILLET, rad,	6-283
6.32.2	ARCSLP/ FILLET, SAVE	6-286
6.32.3	ARCSLP/ FILLET, RESTOR	6-286
6.33	CHKPTS	6-286
6.34	CONTCT/ON	6-288
	OFF	
6.35	COPY	6-290
6.35.1	COPY/ index [, SAME [, copies]]	6-290
6.35.2	COPY/ index, MODIFY, matrix [, copies]	6-290
6.35.3	COPY/ index, TRANSL, X-trans, Y-trans, Z-trans [, copies]	6-290
6.35.4	COPY/ index, axis-rotation, angle [, copies]	6-291
6.36	CUT	6-292

TABLE OF CONTENTS

6.37	DNTCUT	6-292
6.37.1	DNTCUT/ NOMORE	6-292
6.38	FEDRAT	6-293
6.38.1	FEDRAT/ feedrate [,IPM]	6-293
	IPR	
6.38.2	FEDRAT/ [AT , dist-1 [, SCALE] , fs1 [, nfed] [, ONCE] [[,] OUT [[, dist-2] [[, SCALE], fs2] [, nfed]] [, ONCE]] [, LENGTH , dis]	\$ \$ 6-293
6.39	GOUGCK	6-296
6.39.1	GOUGCK/ PS , 0, DS, 0, CS, 0	6-296
	1 1 1	
	2 2 2	
	3 3 3	
	4 4	
6.39.2	GOUGCK/ ON [, 1]	6-298
	2	
	3	
6.39.3	GOUGCK/ OFF	6-298
6.40	INDEX	6-298
6.40.1	INDEX/ id number	6-298
6.40.2	INDEX/ id number, NOMORE	6-299
6.41	MAXANG	6-299
6.42	MAXDP	6-300
6.43	MULTAX	6-301
6.43.1	MULTAX [/ ON]	6-302
6.43.2	MULTAX/ OFF	6-302
6.44	NUMPTS	6-302
6.45	PRINT	6-302
6.45.1	PRINT/ SMALL	6-303
6.45.2	PRINT/ LARGE	6-303
6.46	RAPID [/ Optional Parameters]	6-303
6.47	REVERS/ON	6-303
	OFF	
6.48	SEQUNC	6-303
6.49	SET	6-304
6.49.1	SET/ MODE, CIRCUL	6-304
6.49.2	SET/ MODE, LINEAR	6-305
6.50	THICK	6-305
6.50.1	THICK/ distance-PS [, distance-DS [, distance-CS]]	6-305
6.50.2	THICK/ ps, ds, cs1 [, cs2 [, cs3 [, cs4 [, cs5]]]]	6-305
6.50.3	THICK/ OFF	6-306
6.51	TOLER/ chor-tol [, pos-tol]	6-306
6.52	TRACUT	6-308

TABLE OF CONTENTS

6.52.1	TRACUT/ matrix	6-308
6.52.2	TRACUT/LAST,matrix	6-309
	NOMORE	
6.52.3	TRACUT/ NOMORE	6-309
6.53	TRALST/ ON	6-310
	OFF	
6.54	UNITS/ MM	6-310
	INCHES	
7	PROGRAM CONTROL STATEMENTS	7-1
7.1	CALL	7-1
7.1.1	CALL/ macro-name [, parameter list]	7-1
7.2	CONTIN	7-1
7.3	DO	7-2
7.4	FINI	7-4
7.5	FORMAT	7-4
7.5.1	FORMAT/SHORT	7-4
7.5.2	FORMAT/LONG	7-5
7.6	IF Statements	7-5
7.6.1	IF (arithmetic)	7-5
7.6.2	IF (logical)	7-5
7.6.3	IF-Then-Else Structures	7-8
7.7	IFTOL	7-10
7.8	INCLUD	7-10
7.9	INSERT	7-11
7.10	JUMPTO	7-12
7.11	Logical-expression	7-12
7.12	LOOPND	7-14
7.13	LOOPST	7-14
7.14	MACRO	7-14
7.14.1	Local Identifiers	7-16
7.15	ON/ERROR,THEN,CONTIN[, WARN]	7-17
	STOP NOWARN	
	label	
7.16	PPRINT	7-18
7.17	PROMPT	7-18
7.17.1	PROMPT To Allow Class And Description Text Attached To A Scalar Variable	7-19
7.17.2	PROMPT In *RUN Mode	7-19
7.17.3	PROMPT To Customize A Dynamic Macro Form	7-21
7.17.4	PROMPT To Allow A Description Of A Macro	7-23
7.18	Dynamic Macro Calling	7-24

TABLE OF CONTENTS

7.19	READ	7-30
7.20	REMARK	7-31
7.20.1	REMARK/ON OFF	7-31
7.21	SYN	7-32
7.22	TERMAC	7-32
7.23	TITLES	7-33
7.24	UNDO	7-33
7.25	Restoring The Current Program In The Event Of An Abnormal Termination Of NCL	7-34
8	GRAPHIC DISPLAY CONTROL STATEMENTS	8-1
8.1	Color	8-1
8.2	Draft Commands	8-2
8.2.1	Set Geometry Default Color	8-2
8.2.2	Set Axes Default Color	8-4
8.2.3	To Control Attributes	8-4
8.2.4	Modify Viewing Parameters	8-8
8.2.5	Selectively Modify Viewports	8-9
8.2.6	Modify Or Create Views	8-9
8.2.7	Perform An EXTREME ZOOM	8-11
8.2.8	Repaint The Screen Or A View	8-12
8.2.9	Reset Specified Views	8-12
8.2.10	Control Tool Motion Display	8-12
8.2.11	Label Display Properties	8-15
8.2.12	Label Display Control During Geometry Creation	8-17
8.2.13	Labels ON And OFF For Previously Defined Geometries	8-18
8.2.14	Location Of Geometry Labels	8-19
8.3	DISPLAY Statement:	8-20
8.3.1	DISPLAY/curve-name [,number-of-points]	8-20
	DISPLAY/surface-name [, number-of-points, number-of-u-lns, \$ number-of-points, number-of-v-lns]	8-20
	DISPLAY/surface-name [, number-of-u-lns, number-of-v-lns]	8-20
8.3.2	DISPLAY/ plane, AT, point	8-21
8.3.3	DISPLAY/ geometry-list	8-21
8.3.4	DISPLAY/ALL	8-22
8.3.5	DISPLAY/ CUTTER	8-22
8.3.6	DISPLAY/ matrix-name [, ax, boxx [, boxy, boxz]]	8-22
8.3.7	DISPLAY/ LAYER, n1, n2, ... [, THRU, nn] [, nm]	8-23
8.3.8	DISPLAY/MAXIS WAXIS	8-23
8.4	ERASE Statement	8-23

TABLE OF CONTENTS

8.4.1	ERASE/ ALL	8-24
8.4.2	ERASE/ geometry-list	8-24
8.4.3	ERASE/ LAYER, n1, n2, ... [, THRU, nn] [, nm]	8-24
8.4.4	ERASE/MAXIS	8-24
	WAXIS	
8.4.5	ERASE/ MOTION	8-24
8.5	INVIS Statement	8-24
8.6	VISIBL Statement	8-25

9 NCL CONTROL COMMANDS 9-1

9.1	*CONSOL	9-1
9.2	*DELETE	9-1
9.3	*EDIT	9-1
9.3.1	*EDIT [/ positive-scalar]	9-2
9.4	*EDT	9-2
9.5	*FIND And *FINDTK	9-3
9.6	*INPUT	9-3
9.7	*INSERT	9-3
9.8	*LOADAPP	9-4
9.9	*PAUSE	9-4
9.10	*RESET	9-4
9.10.1	*RESET/ ADISPL	9-5
9.10.2	*RESET/ APTSRC, CIRCUL	9-5
9.10.3	*RESET/ APTSRC, DATA	9-5
9.10.4	*RESET/ APTSRC, IPV	9-5
9.10.5	*RESET/ APTSRC, IPVCOM	9-5
9.10.6	*RESET/ APTSRC, VERIFY	9-6
9.10.7	*RESET/ APTSRC, REMARK	9-6
9.10.8	*RESET/ APTSRC, REMARK, ACTIVE	9-6
9.10.9	*RESET/ AUTOST [, tol, angtol] [,RETAIN]	9-7
	OMIT	
9.10.10	*RESET/ AUTOUV	9-7
9.10.11	*RESET/ CANON	9-7
9.10.12	*RESET/ CALL [, n]	9-8
9.10.13	*RESET/ CASE	9-8
9.10.14	*RESET/ DISPLAY [, geometry-type-list]	9-8
9.10.15	*RESET/ EXPCL	9-8
9.10.16	*RESET/ INDENT	9-9
9.10.17	*RESET/ MOTION	9-9
9.10.18	*RESET/ NOWARN	9-9
9.10.19	*RESET/ PAUSE	9-9
9.10.20	*RESET/RUNCMD	9-9

TABLE OF CONTENTS

9.10.21	*RESET/ SCHECK	9-10
9.10.22	*RESET/ STATLN	9-10
9.10.23	*RESET/ STOP	9-10
9.10.24	*RESET/STPCMD	9-10
9.11	*RUN	9-10
9.11.1	*RUN/ scalar	9-11
9.11.2	*RUN/ MACRO	9-11
9.11.3	*RUN/ TERMAC	9-11
9.11.4	*RUN/ LOOPST	9-11
9.11.5	*RUN/ LOOPND	9-11
9.12	*SAVEPP	9-11
9.13	*SET	9-12
9.13.1	*SET/ ADISPL	9-12
9.13.2	*SET/ ADISPL, TOLER, tol	9-13
9.13.3	*SET/ ADISPL, [number-of-points,] number-of-u-lns, number-of-v-lns	\$ 9-13
9.13.4	*SET/ ADISPL, [number-of-points,] number-of-points, number-of-u-lns, number-of-points, number-of-v-lns	\$ 9-13
9.13.5	*SET/ APTSRC, CIRCUL [,IPV]	9-13
	VERIFY	
9.13.6	*SET/ APTSRC, CUTTER,APT NCL PPRINT scalar	9-14
9.13.7	*SET/ APTSRC, DATA	9-15
9.13.8	*SET/ APTSRC, IPV	9-15
9.13.9	*SET/ APTSRC, IPVCOM	9-15
9.13.10	*SET/ APTSRC, LOW	9-15
	HIGH	
9.13.11	*SET/ APTSRC, REAL	9-16
9.13.12	*SET/ APTSRC, REMARK	9-16
9.13.13	*SET/ APTSRC, REMARK, ACTIVE	9-16
9.13.14	*SET/ APTSRC, TRACUT [, matrix]	9-17
9.13.15	*SET/ APTSRC, VERIFY	9-17
9.13.16	*SET/ AUTOST [, tol, angtol [,RETAIN]]	9-17
	OMIT	
9.13.17	*SET/ AUTOUV	9-18
9.13.18	*SET/ CANON	9-18
9.13.19	*SET/ CASE	9-18
9.13.20	*SET/CMDLEN, n	9-19
9.13.21	*SET/CMDCOM, n	9-19
9.13.22	*SET/ DISPLAY [, geometry-type-list]	9-19
9.13.23	*SET/ ELIMIT, scalar	9-19

TABLE OF CONTENTS

9.13.24	*SET/ EXPCL	9-20
9.13.25	*SET/ geometry-type, scalar	9-20
9.13.26	*SET/ INDENT, ALL, scalar SEP, scalar OFF	9-20
9.13.27	*SET/ MOTION	9-21
9.13.28	*SET/ NOWARN	9-21
9.13.29	*SET/ PAUSE	9-22
9.13.30	*SET/RUNCMD	9-22
9.13.31	*SET/ SCHECK	9-22
9.13.32	*SET/ STATLN	9-22
9.13.33	*SET/ STOP	9-22
9.13.34	*SET/STPCMD	9-22
9.13.35	*SET/ TRIMMED,FACE BASE DEFALT	9-23
9.13.36	*SET/ VER, vflag	9-23
9.13.37	*SET/ WLIMIT, scalar	9-24
9.14	*SHOW	9-24
9.14.1	*SHOW/ identifier	9-25
9.14.2	*SHOW/ ADISPL	9-25
9.14.3	*SHOW/ CUTTER	9-25
9.14.4	*SHOW/ FEDRAT	9-25
9.14.5	*SHOW/ FILES	9-25
9.14.6	*SHOW/ MAXANG	9-26
9.14.7	*SHOW/ MAXDP	9-26
9.14.8	*SHOW/ MODALS	9-26
9.14.9	*SHOW/ MODSYS	9-26
9.14.10	*SHOW/ NUMPTS	9-26
9.14.11	*SHOW/ REFSYS	9-26
9.14.12	*SHOW/ SOURCE [, scalar]	9-26
9.14.13	*SHOW/ SOURCE, S	9-26
9.14.14	*SHOW/ STATLN	9-27
9.14.15	*SHOW/ THICK	9-27
9.14.16	*SHOW/ TOLER	9-27
9.14.17	*SHOW/ TOOL	9-27
9.14.18	*SHOW/ TLAXIS	9-28
9.14.19	*SHOW/ TRACUT	9-28
9.14.20	*SHOW/ UNITS	9-28
9.15	*SHOW/ vocab-word	9-28
9.15.1	*SKIP	9-29
9.15.2	*SKIP/ scalar	9-29

TABLE OF CONTENTS

9.15.3	*SKIP/ TO, scalar	9-29
9.15.4	*SKIP/ TO, END	9-29
9.16	*STOP	9-29
9.17	*SYSTEM [/ system-command [, OFF]]	9-30
	ON	
9.18	*TIME	9-30
9.19	*VER	9-30
9.20	*WINDOW [/ OPEN] And *WINDOW/ CLOSE	9-30
9.21	** command	9-31
9.22	NON-CONTROL Command * Statements:	9-31
9.23	NON-CONTROL Command * Statements In LOOPS And MACROS ..	9-32
10 NCL/IGES	10-1
10.1	Startup	10-1
10.2	Importing IGES Files	10-1
10.2.1	IGES IN	10-2
10.2.2	Input IGES Data File Name:	10-2
10.2.3	Output Unibase File Name:	10-2
10.2.4	File Summary	10-3
10.2.5	Options	10-3
10.2.6	RUN	10-14
10.2.7	EXIT	10-15
10.3	Viewing The Translated File	10-15
10.4	Translating The IGES View Entity (Type 410)	10-15
10.4.1	Removing View Dependency	10-16
10.5	Translating The IGES Drawing Entity (Type 404)	10-17
10.6	Subfigures And Subfigure Instances	10-17
10.7	Translating The IGES Solid Entity (Type 186)	10-18
10.8	User Definable Maximum Parametric Record Size	10-18
10.9	Exporting IGES Files	10-18
10.9.1	IGES OUT	10-19
10.9.2	Input Part File Name:	10-19
10.9.3	Output IGES Data File Name:	10-19
10.9.4	Output Drawings Only	10-20
10.9.5	Output Units	10-20
10.9.6	RUN	10-20
10.9.7	EXIT	10-20
10.10	Imported Entity Mapping	10-20
10.11	Exported Entity Mapping	10-22
10.12	Miscellaneous	10-23

TABLE OF CONTENTS

11 NCL/STEP	11-1
11.1 Startup	11-1
11.2 Importing STEP Files	11-1
11.2.1 STEP IN	11-2
11.2.2 Input STEP Data File:	11-2
11.2.3 Output Unibase File:	11-2
11.2.4 File Summary	11-3
11.2.5 Options	11-3
11.2.6 RUN	11-14
11.2.7 EXIT	11-14
11.3 Viewing The Translated File	11-15
11.4 User Definable Maximum Parametric Record Size	11-15
11.5 Exporting STEP Files	11-15
12 LATHE MODULE	12-1
12.1 Defining A Shape	12-1
12.2 Rough Cutting The Shape	12-3
12.3 Finish Cutting The Shape	12-6
13 NCL/TOOLIB	13-1
13.1 Startup	13-1
13.2 Using NCL/TOOLIB	13-2
13.3 Accessing Tools From The Library	13-32
13.3.1 Accessing The Library Using The CUTTER/TOOL Statement	13-32
13.3.2 Accessing The Library Using The CUTTER/READ Statement	13-33
13.3.3 Accessing Tools Using The Interactive Interface	13-34
13.4 Working Units Of NCL/TOOLIB	13-38
14 NCL/IPV COMMANDS	14-1
14.1 PPRINT IPV Command	14-1
14.2 Creating/Loading Stock Or Fixture	14-2
14.2.1 PPRINT IPV FIXTUR BOX id x1,y1,z1,x2,y2,z2 STOCK	14-2
14.2.2 PPRINT IPV FIXTUR CONE id x,y,z,i,j,k,r1,r2,h STOCK	14-2
14.2.3 PPRINT IPV FIXTUR CYLNDR id x,y,z,i,j,k,r,h STOCK	14-3
14.2.4 PPRINT IPV FIXTUR SPHERE id x,y,z,r STOCK	14-4

TABLE OF CONTENTS

14.2.5	PPRINT IPV FIXTUR TORUS id x,y,z,i,j,k,r1,r2	14-4
	STOCK	
14.2.6	PPRINT IPV FIXTUR LOAD id “file_name.stk”	14-5
	STOCK	
14.2.7	PPRINT IPV FIXTUR STL id INCHES “file_name.stl”	14-5
	STOCK MM	
14.2.8	PPRINT IPV FIXTUR CLONE id idn,ncopies [, ~]	
	STOCK	
	PPRINT AT, x1,y1,z1,d1, x2,y2,z2,d2, x3,y3,z3,d3]	14-5
	TRANSL, x,y,z	
	XYROT, ang	
	YZROT	
	ZXROT	
14.2.9	PPRINT IPV FIXTUR COMPOS id id-list	14-6
	STOCK	
14.2.10	PPRINT IPV FIXTUR DECOMP id id-list	14-6
	STOCK	
14.3	Stock/Fixture Control Commands	14-7
14.3.1	PPRINT IPV FIXTUR MOVE 0 x1,y1,z1,d1, ~	
	STOCK 1	
	INCR	
	PPRINT x2,y2,z2,d2,x3,y3,z3,d3,id-list.	14-7
14.3.2	PPRINT IPV FIXTUR TRANSL 0 x,y,z,id-list	14-7
	STOCK 1	
	INCR	
14.3.3	PPRINT IPV FIXTUR XYROT 0 angle,id-list	14-8
	STOCK YZROT 1	
	ZXROT INCR	
14.3.4	PPRINT IPV FIXTUR REMOVE id-list	14-8
	STOCK	
14.3.5	PPRINT IPV STOCK SAVE id [“file”]	14-8
	FIXTUR	
14.3.6	PPRINT IPV FIXTUR MODIFY 0 color,visible,lucency,active,~	14-9
	STOCK	
	PPRINT toler,id-list	14-9
14.3.7	PPRINT IPV STOCK REMOVE_CHIPS 0 x1,y1,z1,i1,j1,k1,...,~	
	PPRINT xn,yn,zn,in,jn,kn	14-9
14.4	CUTTER/SHANK/HOLDER Commands	14-10
14.4.1	PPRINT IPV CUTTER parameters	14-10
14.4.2	PPRINT IPV CUTTER BLADE parameters	14-10
14.4.3	PPRINT IPV CUTTER LATHE parameters	14-10
14.4.4	PPRINT IPV CUTTER DISPLAY “pt-list”	14-10
14.4.5	PPRINT IPV SHANK parameters	14-10

TABLE OF CONTENTS

14.4.6	PPRINT IPV SHANK DISPLAY “pt-list” parameters	14-10
14.4.7	PPRINT IPV HOLDER parameters	14-10
14.4.8	PPRINT IPV HOLDER DISPLAY “pt-list” parameters	14-11
14.5	NCL/IPV Model Commands	14-11
14.5.1	PPRINT IPV MODALS AUTO_HIDE [mode] [TRANS tval] ~ PPRINT [EDGES emode]	14-11
14.5.2	PPRINT IPV MODALS COLORS [CUT ccol] [CUTTER ctcol] ~ PPRINT [SHANK scol] [HOLDER hcol] [FIXTUR_CUT fcol] ~ PPRINT [HOLDER_CUT hccol] [RAPID_CUT rcol] ~ PPRINT [AUTO_COLOR acol] [USE_STOCK smod] ~ PPRINT [USE_FIXTUR fmod]	14-11
14.5.3	PPRINT IPV MODALS MACHINE type	14-12
14.5.4	PPRINT IPV MODALS STACK [mode] [FIXTUR fstate] [size]	14-12
14.5.5	PPRINT IPV MODALS STOCK [COLOR col] ~ FIXTUR PPRINT [VISIBLE vmod] [TRANS tval] [TOLER tol] ~ PPRINT [IMPORTANT imod] [EDGES ecol] [STL smod] ~ PPRINT [STL_STOP spmod] [STL_DEACT sdmod] ~ PPRINT [STL_SKIP_ERROR skmod]	14-12
14.5.6	PPRINT IPV MODALS TOOL [TOLER tol] [MAXANG ang] ~ PPRINT [TRANS tval] [EDGES ecol] [MIN_HEIGHT nhgt] ~ PPRINT [MAX_HEIGHT xhgt] [MIN_DIAMETER ndia] ~ PPRINT [FROM_NEXT fmod] [SHANK smod] [RAPID rap]	14-13
14.6	Miscellaneous PPRINT IPV Commands	14-13
14.6.1	PPRINT IPV DNTCUT	14-13
14.6.2	PPRINT IPV OFFSET “file.ofs”	14-14
14.6.3	PPRINT IPV OFFSET label ofs [label2 ofs2 [...] labeln ofsn]	14-15
14.6.4	PPRINT IPV POSITN label pos [label2 pos2 [...] labeln posn]	14-15
14.6.5	PPRINT IPV PRINT_SCREEN type size [“file”]	14-15
14.6.6	PPRINT IPV SESSION EXPORT [“file”]	14-16
14.6.7	PPRINT IPV SESSION IMPORT “file”	14-16
14.6.8	PPRINT IPV SPINDLE n [, n1, n2, ...]	14-16
14.6.9	PRINT IPV STOCK RESET_CUTCOLOR [id-list]	14-16
14.6.10	PPRINT IPV TOOL [CUT_COLOR ccol] ~ PPRINT [CUTTER_COLOR ctcol] [CUTTER_EDGES cecol] ~ PPRINT [CUTTER_TRANS ctval] [HOLDER_COLOR hcol] ~ PPRINT [[HOLDER_EDGES hecol] [HOLDER_TRANS htval] ~ PPRINT [MAXANG ang] [RAPID rap] [SHANK smod] ~ PPRINT [SHANK_COLOR scol] [SHANK_EDGES secol] ~ PPRINT [SHANK_TRANS stval] [TOLER tol]	14-16
14.6.11	PPRINT IPV TOOLPN [label] [x,y,z,i,j,k,u,v,w]	14-17
14.6.12	PPRINT IPV VIEW FIT	14-18

TABLE OF CONTENTS

APPENDIX A: ERROR MESSAGES **A-1**

APPENDIX B: VOCABULARY **B-1**

B.1	Vocabulary List	B-1
B.2	Postprocessor Statements	B-25
B.2.1	Cycle Statements	B-26
B.3	Define Synonyms For Existing Words	B-27
B.3.1	Define Synonyms For Existing Words Inside A Part Program	B-27
B.3.2	Define Synonyms For Existing Words By Using The “nclvoc.syn” File..	B-28
B.4	Add New Word To The Vocabulary List.	B-29

APPENDIX C: PREVIOUS VERSION COMPATIBILITY **C-1**

C.1	Previous Version Unibase Files	C-1
C.2	MESH SURFACES	C-2
C.3	Loading A Mesh Surface File	C-2
C.3.1	Converting The Mesh Surface File	C-2
C.3.2	Building The Mesh Surface	C-2
C.3.3	Displaying The Mesh Surface	C-5
C.3.4	Using The Mesh Surface	C-5
C.3.5	Example Mesh Surface File	C-5
C.4	QUILT SURFACES	C-6
C.4.1	Loading The Quilt Surface File	C-6
C.4.2	Converting The Quilt Surface File	C-6
C.4.3	Building The Quilt Surface	C-7
C.4.4	Displaying The Quilt Surface	C-7
C.4.5	Using The Quilt Surface	C-7
C.4.6	Example Quilt Surface File	C-7
C.4.7	Obsolete Data Base File Name (DBFN)	C-9
C.5	Obsolete Commands	C-9
C.5.1	*DBSHOW	C-10
C.5.2	*QUIT	C-10
C.5.3	REDEF/line , point-1, point-2	C-10
	circle	
C.5.4	REDEF/ line, geometry [, near-pt] [, modifier-1 [, modifier-2]]	C-10
C.5.5	REDEF/ circle, geometry [, modifier-1 [, modifier-2]]	C-11
C.5.6	*RESET/ APTCOM	C-11
C.5.7	*RESET/ CIRAPT	C-11
C.5.8	DISPDB Statement	C-11
C.5.9	*RESET/ ECHO	C-11
C.5.10	*RESET/ LABEL, geometry-type-list	C-12

TABLE OF CONTENTS

C.5.11	*RESET/ PLOT, terminal-type GRAPH	C-12
C.5.12	SCRUB	C-12
C.5.13	*SET/ APTCOM	C-13
C.5.14	*SET/ AUTOL1	C-13
C.5.15	*SET/ CIRAPT	C-13
C.5.16	*SET/ ECHO	C-13
C.5.17	*SET/ LABEL	C-14
C.5.18	SHAPE Command	C-14
C.6	IGES Out Label Options	C-14
C.7	NCL Recover Utility	C-15
C.8	Previous Versions Colors Numbering Scheme	C-16
C.9	Obsolete NCL/IPV Command	C-18
C.9.1	FIXTUR/ BOX, id, 1, point , point [, minz, maxz] STOCK x1,y1,z1 x2,y2,z2	C-18
C.9.2	FIXTUR/ BOX, id, 2, point, width, length, height STOCK x,y,z	C-18
C.9.3	FIXTUR/ CONE, id, 2,point_1 , point_2, radius_1, radius_2 STOCK x1,y1,z1 x2,y2,z2	C-19
C.9.4	FIXTUR/ CONE, id, 3,point , vector, height, radius_1, radius_2 STOCK x,y,z i,j,k	C-19
C.9.5	FIXTUR/ CYLNDR, id, 1, circle, length STOCK	C-20
C.9.6	FIXTUR/ CYLNDR, id, 2,point , point , radius STOCK x1,y1,z1 x2,y2,z2	C-21
C.9.7	FIXTUR/ CYLNDR, id, 3,point , vector, height, radius STOCK x,y,z i,j,k	C-21
C.9.8	FIXTUR/ SOLID, id, solid STOCK	C-22
C.9.9	FIXTUR/ SPHERE, id, 1, circle STOCK	C-22
C.9.10	FIXTUR/ SPHERE, id, 2,point, radius STOCK x,y,z	C-23
C.9.11	FIXTUR/ TORUS, id, 1, circle_1, circle_2 STOCK	C-23
C.9.12	FIXTUR/ TORUS, id, 2, circle, radius STOCK	C-24
C.9.13	FIXTUR/ TORUS, id, 3,point, vector, radius_1, radius_2 STOCK x,y,z i,j,k	C-24
C.9.14	FIXTUR/ CLONE, id1, id2 [, m] STOCK	C-25
C.10	Loading A Stock Or Fixture File	C-25

TABLE OF CONTENTS

C.10.1	FIXTUR/ LOAD, id, [“]file_name[“][, n]	C-26
	STOCK	
C.10.2	FIXTUR/ STL, id, INCHES, [“]file_name[“]	C-26
	STOCK MM	
C.10.3	FIXTUR/ MODIFY, id1 [[[. . .] [, idn, THRU, idm]] [. . .]] \$	
	STOCK	
	[, COLOR=color] [, VISIBL=ON]	\$
	OFF	
	[, TOLER=tol] [, TRANS=tra] [, ACTIVE=ON]	C-27
	OFF	
C.10.4	FIXTUR/ MOVE, id1 [[[. . .] [, idn, THRU, idm]]	\$
	STOCK	
	[. . .], AT, matrix	C-27
C.10.5	FIXTUR/ REMOVE, id1 [[[. . .] [, idn, THRU, idm]] [. . .]]	C-28
	STOCK	
C.10.6	STOCK/REMOVE,CHIPS,pv1[...]	C-28
C.10.7	TOOLPN/ x, y, z, i, j, k, u, v, w	C-28
C.10.8	*SET/RAPID,r	C-28

APPENDIX D: Summary of *NCL* Commands **D-1**

APPENDIX E: Programmable Functions **E-1**

APPENDIX F: Programmable Keys **F-1**

F.1	Hot Keys Definition File	F-1
F.2	Key Names For Keyboard Hot Keys Definition	F-2
F.3	Key Names For SpaceMouse's Dials And Buttons Definition	F-7

APPENDIX G: Miscellaneous **G-1**

G.1	Save And Load Session	G-1
G.2	Autosave Part Program or Unibase	G-1
G.3	Define Scalar Variables With Interface Form	G-4
G.4	Select A Scalar Variable With Interface Form	G-6
G.5	Select A Data Statement Element With Interface Form	G-7
G.6	NCL Visual Calculator	G-9
G.7	Color Form	G-18
G.8	Geometry Entity Data Query And Edit Tool	G-19
G.9	Geometry Entity Attributes Query And Edit Tool	G-23
G.10	Geometry Entity Measuring Tool	G-26
G.11	Motion Query And Edit Tool	G-31
G.12	Customized Screen Layout With Customized View	G-34

TABLE OF CONTENTS

G.13	Customizing Macro Form	G-36
G.13.1	Customizing Macro Form With External File	G-36
G.13.2	Customizing Macro Form Interactively	G-48
G.13.3	Adding Picture To A Macro Form	G-56
G.14	Batch Processing	G-58
G.15	Mouse Buttons	G-65
G.16	NCL Initialization Files	G-66
G.16.1	drwsize.init	G-66
G.16.2	nccs.init	G-66
G.16.3	ncl.init	G-66
G.16.4	ncliges.init	G-66
G.16.5	nclstep.init	G-66
G.16.6	nclplot.ini	G-66
G.16.7	ncq.ini	G-67
G.16.8	user.init	G-68
G.17	NCL Modal Files	G-69
G.17.1	ncl.mod	G-69
G.17.2	ncl_autosave.mod	G-75
G.17.3	ncl_background.mod	G-76
G.17.4	ncl_chain.mod	G-79
G.17.5	ncl_cmdline.mod	G-80
G.17.6	ncl_color.mod	G-82
G.17.7	ncl_graphic.mod	G-82
G.17.8	ncl_interface.mod	G-83
G.17.9	ncl_labels.mod	G-87
G.17.10	ncl_motion.mod	G-89
G.17.11	ncl_pick.mod	G-92
G.17.12	ncl_playfeed.mod	G-94
G.17.13	ncl_playinterp.mod	G-94
G.17.14	ncl_source.mod	G-96
G.17.15	ncl_srfatt.mod	G-98
G.17.16	ncl_unibase.mod	G-99
G.17.17	ncl_view.mod	G-100
G.17.18	ncliges.mod	G-103
G.17.19	ncliges_color.mod	G-105
G.17.20	nclstep.mod	G-106
G.17.21	nclstep_color.mod	G-108
G.17.22	nclipv_*.mod	G-108

INDEX

1 OVERVIEW

This reference manual describes the **NCL** Multi-axis Machining Software product. **NCL** (Numerical Control Language) is a Computer-aided manufacturing software package used to create programs to run CNC machine tools. **NCL** has been specifically developed for multi-axis machining applications and is ideally suited for environments where design changes are frequent and machine time is critical.

NCL is capable of creating, importing and modifying 3D geometric data which is used to represent the component(s) to be machined.

NCL offers a variety of strategies for producing efficient, user controlled toolpaths including simultaneous 4 and 5-axis motion.

NCL receives its instructions from the user in the form of a part program. A **NCL** part program consists of statements and commands which make up the Numerical Control Language. **NCL**'s syntax is based on the ANSI X3.37-1977 standard for the APT programming language. **NCL** implements a subset of that standard as well as extensions to that standard. It is important to note that **NCL** is not APT. **NCL** relies on the APT syntax because of its industry familiarity, however, the internal processor is unique to **NCL** and incorporates the most modern CAD/CAM strategies for solving geometric and NC toolpath problems.

The **NCL** user can produce a part program using **NCL**'s interactive, graphical user interface. The interface produces **NCL** part program statements, provides dynamic viewing of the geometric model and toolpath data, produces a variety of output files, and allowed previously created part programs to be processed and modified.

The **NCL** user may also write or modify a part program using a standard text editor. The program can then be processed interactively to verify the syntax and function of the program.

NCL users tend to use a combination of interactive program creation/modification and text editing.

NCL programs can also be processed in a non-interactive batch mode which processes the program, reports errors, and produces the output files. Processing in batch mode makes it easy to make changes to the program, (i.e. modifying a variable, changing a feed rate, etc.) and quickly generate new results.

1.1 Convention Used

The convention of **NCL** language syntax construction used in this document is intended to show every possible combination of allowable syntax for a given statement. This is done through symbolic representation of the various tokens that make up an **NCL** statement. For example, the symbolic representation of a line expression is:

LINE/point, point

In this example, the word “point” in lower case letters is used to show that any valid syntax (including nested expressions) which defines a point may appear at that location. All punctuation shown must appear exactly as in the example. In this case, the word “LINE” must appear and be followed by a “/”, and the comma must appear between the first point and the second point. Also notice that nothing may appear after the second point. If more text is allowed to follow, the symbol “...” is used. For example, in the SURFACE expression:

SURF/curve, [THRU,] curve[. . .], curve

the symbol [...] signifies that more curves may appear at that location in the SURF expression. The use of brackets signifies an optional item. The optional [THRU,] may or may not appear.

1.1.1 Word Conventions

The following conventions are used to distinguish the different types of words used throughout this manual when showing a general format (these conventions do not apply to actual part programs or sample part program statements in this manual).

- All **NCL** vocabulary words are printed in capital letters throughout this manual. For example: COOLNT/OFF
- In command syntax examples, a word that specifies a scalar identifier, a geometric identifier, a matrix identifier, a macro identifier or an actual numerical value is shown in lowercase letters only.
- Items that are optional are enclosed in square brackets [] except where otherwise indicated.
- An item that may be repeated is indicated as follows: [...]

1 OVERVIEW

1.1.2 Modifiers

1.1.2.1 Vertical Lists

When modifiers are written in vertical lists, the user is expected to choose the most appropriate modifier.

Example:

```
XLARGE  
XSMALL  
YLARGE  
PLANE/PARREL,plane,YSMALL,distance  
ZLARGE  
ZSMALL
```

1.1.2.2 Optional Modifiers

Optional informational fields in the Command Syntax representation are enclosed in brackets ([]). Optional information may be specified by the **NCL** user, but it is not required by **NCL**. In some cases, the optional information is established by a default value, that is, **NCL** uses a system-defined value unless the **NCL** user specifies a replacement value.

Example:

```
CIRCLE/x-coordinate, y-coordinate,      $  
[z-coordinate,] radius
```

1.2 NCL System Features

NCL is an interactive, workstation based part programming system capable of geometry definition and 5-axis motion generation over a variety of complex shapes. Also provided are features such as automatic name generation for geometry, interactive inquiry of geometric and toolpath data, on-line debugging, support for a variety of plotters and printers, and the output of ANSI format CLfiles and APT source files. Key features include:

- The ability to create associative geometry, including sculptured surfaces.
- Toolpaths and geometry are associated so that a change to the geometry results in a change to the toolpath.

- Parametric programming using variables, macros, and looping logic.
- Powerful tool axis control modes for generating 3, 4, and 5-axis simultaneous motion.
- Interactive macro calling with user defined prompts.
- Dynamic display of generated toolpaths.
- 3D dynamic rotation of geometry and toolpaths.
- Importation of data from other CAD/CAM systems.
- Full integration with other CAD/CAM related products.
- Off-line, batch processing.

1.2.1 NCL Product Suite

A complete **NCL** system can be made from a number of individual software modules. The following is a list of optional modules along with a brief description of each one:

- PostWorks** - 2-10 axis universal postprocessor.
- NCL/IPV** - Integrated, solids-based NC verification.
- NCL/CADD** - Standards compliant dimensioning, symbol definition, generation of documentation containing text and graphics.
- NCL/IGES** - Reads and writes IGES files.

1.2.2 NCL Execution Modes

NCL part programs can be run either interactively or in an unattended batch mode. The dual processing modes of **NCL** provide the user with the “best of both worlds.” There are many valid and justifiable reasons why the **NCL** user will want to use both execution modes. The interactive mode is a powerful program development tool while the batch mode is the appropriate choice for final output generation and program maintenance.

1.2.2.1 Interactive Execution Mode

When running **NCL** in the interactive mode, **NCL** “captures” the user’s input and creates an **NCL** part program file. The user can view the geometric model and all created toolpaths, modify the part program, change variables, inquire the data base and many other functions. As a statement is received by the system, it is checked

1 OVERVIEW

for valid syntax and then processed. Editing tools are available for correcting erroneous input, including access to the user's text editor of choice.

1.2.2.2 Batch Execution Mode

The batch execution mode allows one or several part program files to be processed in a background mode, unattended by the user. Output files, including error reports, are generated for later access. This allows easy program maintenance when simple changes are required. It also frees up the workstation for interactive processing when large programs are being processed.

1.2.3 Input/Output File Support

NCL reads and writes a variety of different file types. The following is a list of file types supported by **NCL** (shown with their default file extension) along with a brief description of each one:

Part program file (.pp)

Text file containing **NCL** source statements. This is the primary input file to **NCL** and is how the system receives its instructions for creating geometric and toolpath data. This file can be created in a text editor or by **NCL**'s interactive interface.

Unibase file (.u)

Binary file containing geometric data and associated attributes. This file is created by **NCL** at the user's discretion and can be accessed via other **NCL** part program files. This allows, for example, a master geometry part program to create a single unibase file that can be accessed by any number of part programs for the purpose of creating tool motion. A unibase is the only way to store geometry and dimensions created using the **NCL/CADD** module.

Text unibase file (.ud)

A text version of the unibase file. Storing a unibase file in the text format allows it to be used across different computing platforms.

CLFILE (.cl)

Binary file containing postprocessor words, cutter locations and tool axis vectors. This file is used as input to a postprocessor which will generate the machine specific output file.

APT source file (.as)

Text file containing postprocessor words, cutter locations, and tool axis vectors. This file can be used as input to a postprocessor, read by **NCL** or other APT-based systems, or used as input to NC verification packages such as **NCL/IPV**.

Drawing file (.dw)

Binary file containing 2D geometric data and text. This file is created by **NCL/CADD** and is used primarily for creating printed documents.

Symbol file (.sy)

Binary file containing an **NCL/CADD** symbol. A symbol is a collection of geometric and text data which are treated as a single entity. A symbol created with **NCL/CADD** can be accessed by other files. The **NCL** command: **SYMBOL**, generates a local symbol and can only be accessed locally. It will not store in the symbol file and cannot access by other files.

IGES file (.igs)

Text file in the format of the Initial Graphics Exchange Specification. IGES files can be created by virtually any CAD/CAM system including the **NCL/IGES** translator. The **IGES** translator can also read IGES files which creates a unibase (.u) file containing the converted data.

NCL print file (.pr)

Text file created by **NCL** when run in a batch mode. The **NCL** print file contains a listing of the input part program, generated cutter locations and tool axis vectors shown with the associated motion command, and error messages, if any.

Postscript file (.ps)

An output file generated by **NCL**'s "Plot Screen" or "Plot" functions. This file can then be output by a printer which supports the Postscript format.

1 OVERVIEW

1.3 Hardware And OS Requirements

NCL runs on a variety of popular PC Windows based workstations. The following is a list of currently supported hardware platforms and operating systems.

Platform & Operating System	Graphics	Memory (GB) Min/Rec.	Approx. Disk Space (MB)	Virtual Memory (GB) Min/Rec
Windows XP (Service Pack 2 or Higher), Vista, Windows 7, Windows 8	Open GL	32 bits OS 1/2 64 bits OS 2/4	100	32 bits OS 1/2 64 bits OS 4/8

2 NCL LANGUAGE

2.1 NCL Programming Language Structure And Syntax

NCL is a user-oriented, high-level computer programming language, similar to the APT language.

The user conveys the instructions to **NCL** in the form of a Part Program, which is a collection of English-like statements that may either be written out line by line or automatically generated during an interactive session.

Each statement must contain the exact syntax (grammar, arithmetic, punctuation and order) to convey the correct instruction.

2.2 Statement Structure

2.2.1 Statement Size And Format

NCL will accept either UPPER or lower case.

Each statement may contain a maximum of 1536 characters (excluding blanks) in as many as 50 lines.

A STATEMENT LINE may be up to 1016 characters or columns long.

NCL statements may be written in any position within the value of “one minus number of columns” specified by the [/COMMENT/](#) parameter of the “[ncl_cmdline.mod](#)” file or the “*SET/CMDCOM,n” command, i.e. “n-1”. Blank columns are ignored by **NCL** so blanks may be used to make it easier for the reader to follow the logic used in the part program.

Statements beginning with Fixed Field Words are exceptions to the above rule. These special words MUST be positioned in specific columns. Each word is six (6) characters in length and must be positioned in columns 1 through 6. Columns 7 through 72 are used to define an alphanumeric string of characters. All characters after 72 columns will be truncated and not output to the cl or APT file. Fixed field words are:

PARTNO, [PPRINT](#), [INSERT](#) and LETTER.

2 NCL LANGUAGE

Example:

```
COLUMNS:123456.....72
PARTNO ANB7182-1
PPRINT LOAD TOOL N0.1
INSERT N8T01M06
```

2.2.2 Statement Continuation

Most **NCL** statements are input on a single line but it is possible for a single statement to be continued on one or more successive lines. A single statement may have a maximum of (49) continuation lines, for a total of (50) lines per statement.

The Single Dollar Sign (\$) at the end of a statement indicates that the statement is continued on the next line. This \$ sign must be positioned prior to column specified by the [/COMMENT/](#) parameter in the “[ncl_cmdline.mod](#)” file or the “Command Column” item in the “Command Line Modals” interface form. A statement may be continued on several lines. Any information to the right of the dollar sign is ignored by the processor, so comments may be placed there if desired.

Exceptions to this rule are the [REMARK](#), [TITLES](#) statements and statements with Fixed Field Words of which the \$ sign will be interpreted as part of the body of the command. The [LOADU](#), [UBFN](#), [SAVEU](#) and [INCLUD](#) are other statements that will not allow a \$ sign. Only one line is allowed for these type of statements.

Example: The following statement continues for three lines.

```
PL1=PLANE/ (POINT/1,2,3),$ POCKET BOTTOM
(POINT/4,5,6),$ these are comments
(POINT/7,8,9)
```

2.2.3 Statement Comments

2.2.3.1 The Double Dollar Sign \$\$

Comments may be included along with statement information as shown above or by inserting a \$\$ (double dollar sign). The comments are placed to the right of the double dollar sign. The double dollar sign indicates the logical end of the current statement. This sign must be positioned prior to column specified by the [/COMMENT/](#) parameter in the “[ncl_cmdline.mod](#)” file or the “Command Column” item in the “Command Line Modals” interface form. If no statement

appears to the left of the double dollar sign, the line serves solely as a comment line. A comment or comment line has no effect on the **NCL** Processor; it is ignored.

Exceptions:

- **REMARK, TITLES** statements and statements with Fixed Field Words of which the \$\$ sign will be interpreted as part of the body of the command.
- The **LOADU, UBFN**, and **SAVEU** are other statements that will not allow any \$ sign.

Example: Statement with comment following

```
FEDRAT/15    $$ SET ROUGHING FEEDRATE
```

Example: Comment Only Line

```
$$ INDEX TURRET TO 1.000 INCH END MILL
```

The phrases "SET ROUGHING FEEDRATE" and "INDEX TURRET TO 1.000 INCH END MILL" are comments.

2.2.3.2

The Percent Sign %

The percent sign is an alternate specification for the double dollar sign and may be used in the same way.

Exceptions:

- The Fixed Field Words of which the % sign and whatever follows will be interpreted as part of the body of the command.
- The **LOADU, SAVEU, UBFN** and **INCLUD** are other statements that will not allow any % sign.
- The format string of the text **FORMAT** function in which the % sign is part of the required syntax and does not have the meaning of comment.

Example:

```
PZERO=POINT/0,0,0      % PART ORIGIN X=0, Y=0, Z=0
```

The phrase "PART ORIGIN X=0, Y=0, Z=0" is the comment.

2 NCL LANGUAGE

2.2.3.3 REMARK/ ON OFF

This set of statements is used to denote a section of the part program file not to be processed. For example:

```
REMARK/On
....
Various statements (can be comments or NCL statements)
....
REMARK/OFF
```

All statements between the REMARK/ON and REMARK/OFF statements will be treated as comments.

2.3 Statement Syntax

Part program statements must begin with an **NCL** MAJOR VOCABULARY word, a COMMENT or a LABEL usually followed by MINOR VOCABULARY words and descriptive words, and punctuated in a manner that will convey a specific instruction to **NCL**.

Each **NCL** statement is made up of one or more of the following elements:

- Vocabulary words
- Labels
- Text Control
- Data Fields
- Punctuation
- Expressions
- Identifiers
- Numbers

2.3.1 Vocabulary Words

An **NCL** Vocabulary Word is a word that has a built-in meaning to **NCL**. **NCL** default vocabulary words are limited to six (6) or fewer characters. The six character restriction causes most vocabulary words to be compressed into a unique “short hand”. Some vocabulary words are based on more than one word but are considered single words by **NCL**. For example, the **NCL** word **GORT** is based on the English words go and right.

Vocabulary words are divided into two categories - Major words and Minor words (See Appendix B for a full listing of **NCL** vocabulary words).

Note:

- User defined vocabulary word for **NCL** can be up to twenty-four (24) characters.
- User defined vocabulary word for **PostWorks** can be up to eight (8) characters.

2.3.1.1 Major Words

A Major word establishes the basic meaning of the **NCL** statement. Some major words express a complete meaning by themselves and can stand alone.

Example:

TERMAC and CUT.

However, most major words cannot stand alone and require additional information to express the complete meaning of the statement. In these cases, the major word is followed by a slash (/) which in turn is followed by the supporting information necessary to clarify the meaning of the statement. Only one major word may appear in any statement.

Example Statement Syntax:

FEDRAT/30, I PM

2.3.1.2 Minor Words

Minor words cannot stand alone. Minor words are modifiers which provide supporting information to clarify the meaning of the major word specified in an **NCL** statement. Minor words are used to specify such things as direction, orientation, rotation, position, size and range. Minor words are also used to select between several possible choices.

2.3.1.3 Processor And Postprocessor Words

NCL vocabulary words are further sub-divided into two categories which are evaluated at different stages in the processing of an **NCL** part program. Postprocessor words are not used during the processing phase but are passed directly to the postprocessor for evaluation. Some **NCL** vocabulary words (most often modifiers) are valid for both categories (See [Appendix B](#)).

2 NCL LANGUAGE

Examples, use of the modifier ATANGL:

```
LINE/PT1,ATANGL,45,LN1 $$ a processor statement  
ROTHED/ATANGL,45,CLW $$ a postprocessor statement
```

2.3.1.4 Modifier Words

Modifier Menu selections are written in vertical lists in all command syntax examples. The **NCL** user is expected to choose the most appropriate modifier from the list of allowable options.

2.3.1.4.1 Direction Modifiers

Direction modifiers are Minor words which specify direction relative to the referenced geometric entity as it “sits” within the rectangular coordinate system.

Example:

```
XLARGE  
XSMALL  
PLANE/PARREL,plane,YLARGE,distance  
YSMALL  
ZLARGE  
ZSMALL
```

Here, the appropriate selection from the modifier menu list designates which of six possible directions describes the desired position of the plane being defined relative to the referenced plane as it “sits” within the rectangular coordinate system.

2.3.1.4.2 Orientation Modifiers

Orientation modifiers are Minor words which specify orientation relative to the rectangular coordinate system.

Example:

```
POSX  
POSY  
VECTOR/PERPTO,plane,POSZ  
NEGX  
NEGY  
NEGZ
```

Here, the appropriate selection from the modifier menu list specifies which of six possible choices is the desired orientation of the vector being defined relative to the rectangular coordinate system.

2.3.1.4.3 Rotational Modifiers

Rotational modifiers are Minor words which specify rotation relative to the rectangular coordinate system.

Example:

```
XYROT  
MATRIX/YZROT, angle  
ZXROT
```

2.3.1.4.4 Positional Modifiers

Positional modifiers are Minor words which designate the desired selection based on relative position.

Examples:

```
CIRCLE/YLARGE, IN ,circle, IN ,circle, RADIUS, radius  
                  OUT                OUT  
  
LINE/RIGHT, TANTO, circle, RIGHT, TANTO, circle  
                  LEFT               LEFT  
  
GO/TO ,drive-surface, part-surface, TO , $  
      ON                                 ON  
      PAST                            PAST  
                                     check-surface
```

2.3.1.4.5 Size Modifiers

Size modifiers are Minor words which designate the desired selection based on relative size.

Examples:

```
CIRCLE/CENTER, point, LARGE, TANTO, circle  
                                          SMALL
```

2 NCL LANGUAGE

VECTOR/vector , PLUS , vector
MINUS

2.3.1.4.6 Range Modifiers

THRU (through) is a range modifier that may be used to specify the inclusive (from - through) range of data to be used in an **NCL** statement. Some **NCL** statements allow large amounts of data to define a single geometric entity. If these data elements have been defined earlier in the part program by identifiers or **NCL**'s automatic naming feature, then in place of writing a string of data, such as:

```
CURVE/PT1, PT2, PT3, PT4, PT5, PT6, PT7, PT8, PT9,      $  
      PT10, PT11, PT12, PT13, PT14, PT15, PT16,      $  
      PT17, PT18, PT19, PT20
```

The same curve can be defined as simply:

```
CURVE/PT1, THRU, PT20 or CURVE/1, THRU, 20
```

which implies PT1, THRU, PT20.

A curve defined by a subscripted array of points can be written as:

```
CURVE/P(1), THRU, P(20)
```

INCR (increment) is a range modifier that will allow you to skip items within a range, for example:

```
CURVE/PT1, THRU, PT7, INCR, 2
```

is the same as

```
CURVE/PT1, PT3, PT5, PT7
```

Note: The THRU modifier is not implemented for use with all types of geometric entities in geometry creation. It is only valid for specifying points in a curve or random pattern definition or curves in a surface definition. The value specified as the increment must be a positive number.

The THRU modifier may also be used to specify the list of points in a **POCKET** statement and THRU may be used in the **GET**, **PUT**, **DISPLAY** and **ERASE** commands.

2.3.1.5 Inclusive Subscripts May Also Be Used With The Following Syntax:

```
var( [a, ]THRU,b  [, INCR,c] )  
      ALL    DECR  
  
var(ALL)
```

Where var is a subscripted variable and a, b, and c are scalar values. This is equivalent to a list of subscripted variables starting at var(a), ending at var(b), at intervals of "c". If "a" is omitted, 1 is assumed. If ALL is used in place of "b", the highest subscript defined is used. If the INCR or DECR phrase is omitted, INCR, 1 is assumed.

Examples:

```
CURVE/cvpts(2,THRU,20)  
CURVE/cvpt(ALL)  
SPLINE/cvpv(THRU,ALL,INCR,2)  
SURF/cvdef(NUM(cvdef),THRU,1,DECR,1)
```

2.3.2 Data Fields

Data fields are used for data entry. Data fields appear in lower case lettering in their syntax examples throughout this manual. There are several types of data fields.

2.3.2.1 Explicit Data Fields:

Example statement syntax:

```
CIRCLE/CENTER,point,RADIUS, radius
```

Example statement syntax with data entered:

```
CIRCLE/CENTER, PT1, RADIUS, .5
```

The data field indicated by "point" specifies that the predefined point PT1 is the CENTER of the CIRCLE being defined and "radius" specifies that .5 is the RADIUS of the CIRCLE being defined. Similar examples of statement syntax:

```
POINT/CENTER,circle  
LINE/XAXIS,distance  
YAXIS
```

2 NCL LANGUAGE

2.3.2.2 Implicit Data Fields:

Statement Syntax:

POINT/X-coordinate, Y-coordinate, Z-coordinate

Example statement syntax with data entered:

POINT/1.000,2.000,3.000

The data field is implied by the position of the data within the **NCL** statement; that is, the data $x = 1.000$ is implied by the position of the data “1.000” in the field occupied by “X- coordinate” in the statement syntax.

Explicit and implicit data fields are often used in the same **NCL** statement, for example:

Statement syntax:

LINE/point, ATANGL, angle, line

Statement syntax with data:

LINE/P1,ATANGL,30,L1

The explicit data field is “angle”. The implicit data fields are “point” and “line”.

2.3.2.3 Optional Data Fields:

Optional Data Fields (shown enclosed in brackets “[]” in this manual) indicate that optional information may be specified but is not required by **NCL**.

In some cases, the optional information is established by a default value; **NCL** uses a system-defined value unless the **NCL** user specifies a replacement value.

Statement syntax:

CIRCLE/X-coordinate, Y-coordinate, \$
[Z-coordinate,]radius

If the processor reads only three data fields, the optional Z-coordinate is assumed to be zero or the “DEFAULT value” at that time. For example:

X Y radius
CIRCLE/2.00,3.00,.75 % z=0

If four data fields are entered, the optional Z-coordinate value will be that which is entered.

```
X      Y      Z      radius
CIRCLE/2.00,3.00,1.25,.75           % z=1.25
```

NCL statements can be a combination of all of the above, for example:

Statement syntax:

```
XLARGE
XSMALL
LINE/YLARGE,circle,ATANGL,angle[,line]
YSMALL
```

Statement syntax with data:

```
LINE/XSMALL,C1,ATANGL,30,L1
```

“XSMALL” is the selection from the modifier list, “C1” is an implicit data field, “30” is an explicit data field and “L1” is an optional implicit data field.

2.3.3 Identifiers

Identifiers are names given to items in a **NCL** program and are the only way by which those items may be referenced within the program. Valid identifiers may be from one (1) to sixty-three (63) alphanumeric characters in length and must contain at least one alpha letter. However, automatic naming with prefix style only allows a maximum of 20 alpha characters for the prefix section and subscript style allows a maximum of 20 alphanumeric characters with at least one alpha character. Identifiers may not be vocabulary words. Subscripted identifiers are allowed. The subscript must be a positive number and may not exceed 1,000,000. If a RESERV statement is utilized for the specified geometry type, it cannot exceed the upper value given for the variable name in the RESERV statement for this particular geometry type. (See the **RESERV** statement under Geometry Control Statements) If a real number or an expression yielding a real result is used as a subscript, the fractional part will be truncated.

Identifiers are assigned by the user or, if omitted, automatically by **NCL**. If the user allows **NCL** to assign the identifiers (Prefix style), you may specify, using the “***SET/geometry-type**, number” **NCL** Control Command, the starting identifier number (Default is “1”) for each geometry type.

2 NCL LANGUAGE

The maximum number of automatically named items (Prefix Style) for each geometry type is 1,000,000.

The maximum number of automatically named items (Subscript Style) for each geometry type is 1,000,000.

The Automatic Name Generation feature of **NCL** assigns unique names to each geometric entity by appending the next unused sequential number to the **NCL** synonym for each geometry type:

CIRCLEs	are assigned names CI1 through CI1000000
CURVEs	are assigned names CV1 through CV1000000
LINEs	are assigned names LN1 through LN1000000
MATRIXs	are assigned names MX1 through MX1000000
PATTERNs	are assigned names PN1 through PN1000000
POINTs	are assigned names PT1 through PT1000000
PNTVECs	are assigned names PV1 through PV1000000
PLANEs	are assigned names PL1 through PL1000000
SURFaces	are assigned names SF1 through SF1000000
VECTORs	are assigned names VE1 through VE1000000
NSHAPEs	are assigned names SH1 through SH1000000
SOLIDs	are assigned names SO1 through SO1000000
SYMBOLs	are assigned names SY1 through SY1000000
ANOTEs	are assigned names AN1 through AN1000000

Entity type CURVE includes SPLINE and SSPLIN.

Examples:

POINT/0, 0, 0 would be assigned the name PT1
CIRCLE/CENTER, PT1 would be assigned the name CI1

The following rules apply to the use of identifiers:

- Valid identifiers may be from one (1) to sixty-three (63) alphanumeric characters in length and must contain at least one letter.
- Identifiers may not be the same as **NCL** Vocabulary words or their synonyms. (See [Appendix B](#))
- Once defined, an identifier may not be redefined as a different geometry type.
- Subscripted identifiers can be RESERVEd (see the **RESERV** statement); this is an option. If a subscripted identifier is RESERVEd, it must be a positive integer number and may not exceed the upper value specified in

the RESERV statement. If a real number or an expression yielding a real number is used as a subscript, the fractional part of the number will be truncated. If a subscripted identifier is used without having been RESERVED it will automatically be RESERVED for a default value of 1,000,000.

- Identifiers within a subscripted array may be used to define more than one type of geometry.
- A subscripted identifier may not be a [MACRO](#) name.

Examples:

<u>Valid</u>	<u>Invalid</u>	
A	1	(Must contain at least one letter)
1SURF	SURF	(SURF is a vocabulary word)
XC(3)	XC(-1)	(Subscript cannot be negative)
ICORD(X-Y)		(1 <= X-Y <= 1,000,000)

The two character prefix used in the automatic naming of geometry can be modified by using the DEFNAM/...INDEX... command (See [DEFNAM](#) statement under Geometry Control Statements for detail).

The generated name can also be a fixed base name with an automatically increasing subscript.

2.3.3.1 Default Identifier Naming Convention For Multi-Entities Geometry Creation Commands

The following naming convention is used when multiple entities are created by one command (CURVE or SPLINE command is used as an example):

- 1 If no label is given, the curves are labeled using the current naming default, such as cv3, cv4, etc.
2. If a label is given, the first curve to be generated must be specified with the command.

e.g. CV3 = CURVE/INTOF, ALL, ...

3. If the command is preceded by a subscripted label, e.g.

CCC(2) = CURVE/INTOF, ALL, ...

2 NCL LANGUAGE

the first curve gets the specified subscripted name, i.e. CCC(2), and the subsequent curves get the same name with following:

CCC(3), CCC(4), etc.).

4. If the command is preceded by a non-subscripted label, ending with a letter, e.g.,

ABC = CURVE/INTOF, ALL, ...

the first curve gets the label-specified subscripted name, i.e. ABC(1), and the subsequent curves get the following subscripts

ABC(2), ABC(3), etc.

5. If the command is preceded by a non-subscripted label, ending with a number, e.g.,

Q23 = CURVE/INTOF, ALL, ...

the first curve gets the specified label, i.e. Q23, and the subsequent curves get the following numbers

Q24, Q25, etc.

Note: If lines or circles are created with the command, they are named as curves.

2.3.3.2 Local Identifiers

Local identifiers are names given to items in a **NCL** program within a Macro or a loop. This type of identifier is different from the regular type of identifier. The local identifiers can only be referenced within the defined Macro or the loop and will be automatically deleted when the Macro call or the loop is terminated. Local identifiers are defined by using the # character as the first letter of the identifier.

Examples:

```
#A      =1
#B(1)=2
#PT1   =POINT/#A,#B(1),1
CI1    =CIRCLE/CENTER,#PT1,RADIUS,2
```

Note: #PT1 is different from PT1.

When processing these identifiers inside a macro, **NCL** will internally convert them to macro#label, where macro is the name of the Macro currently being executed and #label is the user provided identifier. If these identifiers are defined inside a “LOOPST/LOOPND” region, **NCL** will internally convert them to LOOPST#label, where label is the user provided identifier. If these identifiers are defined inside a “DO loop” and not inside a macro definition or “LOOPST/LOOPND” region, **NCL** will internally convert them to id#label, where “id” is the label of the “DO loop” being executed and #label is the user provided identifier. If a Macro, “LOOPST/LOOPND” or “DO loop” is not being executed, then an error message will be displayed if a local identifier is defined.

The *SHOW and Scalar form will display local identifier as #label, while most other functions (such as tool tip text when in Pick mode) will display the local identifiers as macro#label, LOOPST#label or loop_id#label.

The maximum number of total characters for local identifiers (includes the # characters) and macro/loop identifier together is 63.

Examples: If the macro/loop identifier has 10 characters, the maximum number of characters allowed for local identifiers (includes the # characters) is 53.

2.3.4 Labels

A statement may have a label or “ID” so that other part program statements can reference it. The label may be from one (1) to sixty-three (63) alphanumeric characters (only 9 digits will be allowed if all numeric) and must be unique within the loop or macro in which it is defined. It must be the first item in a statement and be terminated by a Colon [:] or []) when being defined. When a label is referenced by a statement, the Colon [:] or []) is not specified. Labels may only appear within LOOPS, MACROS and DO loops.

Great care should be taken when deleting or inserting lines that contain labels.

Examples:

<u>Valid</u>	<u>Invalid</u>
L1:	L1 (missing colon)
100:	1.1: (must be integer)
	1234567890: (more than 9 digits)
123456789A:	
1234567890A:	
L(3):	

2 NCL LANGUAGE

LAB2)
JUMPTO/L1 JUMPTO/L1: (colon not allowed)

Example of a LOOP using statement labels. The statement labels are ID1 & ID2

```
LOOPST  
N=0  
ID1) N=N+1  
GOTO/PTS(N)      $$ where PTS(N) is a predefined  
                  $$ array of points  
IF (N>200) ID1, ID2, ID2  
ID2:  
LOOPND
```

2.3.5 Punctuation

Punctuation is used to separate individual words and elements of the **NCL** statements used in the **NCL** language and to indicate arithmetic operations.

Punctuation for the **NCL** language includes the following symbols:

,	comma
/	Slash
:	colon
*	asterisk
**	double asterisk
+	plus sign
-	minus sign
=	equal sign
()	left and right parentheses
.	period
[]	left and right brackets
@	at sign
&	ampersand sign

2.3.5.1 The Comma “,”

The comma is used to separate the collection of minor elements used in an **NCL** statement.

Examples:

```
V1=VECTOR/-SIN(20+5/60),0,COS(20+5/60)  
IF(X>1)10,15,15  
TLRGHT,GOLFT/L1,TANTO,C1
```

2.3.5.2 The Slash “/”

The slash is used to designate and separate the major element of an **NCL** statement from the minor elements or as the arithmetic operator for division; it replaces the division sign “÷”.

Example: The Slash as a separator.

PT4 = POINT/0.500, 1.675

Example: The Slash as the arithmetic operator to indicate Division is to take place.

A = B/D \$\$ A equals B divided by D

2.3.5.3 The Colon “:”

The colon is used to separate a statement Label from the statement.

Example:

10 : N=N+1

The label “10:” refers to the statement N=N+1

2.3.5.4 The Asterisk “*”

The asterisk is used as the arithmetic operator for Multiplication; it replaces the times sign “×”.

Example:

A = B*C \$\$ A equals B multiplied by C

2.3.5.5 The Double Asterisk “**”

The double asterisk is used as the arithmetic operator to indicate Exponentiation.

Example:

A = B**4 \$\$ A equals B * B * B * B

2 NCL LANGUAGE

2.3.5.6 The Plus Sign “+”

The plus sign is used as the arithmetic operator for Addition or to specify a positive number. In specifying a positive number, plus is assumed if no sign is specified.

Examples:

```
A =B+C           $$ A equals B plus C all  
P1=POINT/+1,2,3    $$ numbers are positive
```

2.3.5.7 The Minus Sign “-”

The minus sign is used as the arithmetic operator for Subtraction or to specify a negative number.

Examples:

```
A =B-C           $$ A equals B minus C  
P1=POINT/1,-1,1.25    $$ "y" coordinate is "-1"
```

2.3.5.8 The Period “.”

The period is used as a Decimal Point to specify the fractional part of a number. If a number has no fractional part, a decimal point is not required but it is permissible.

Example:

```
A=1.000+1.00+1.0+1.+1      $$ A equals 5
```

2.3.5.9 The Equal Sign “=”

The equal sign is used as follows:

- To assign a name (Identifier) to a scalar variable, a geometric entity, a **MATRIX** or a **MACRO**.

Examples:

```
A = .125           $$ Scalar  
PT1=POINT/X,Y,Z        $$ Entity  
MX1=MATRIX/PT1,VE1,VE2    $$ Matrix
```

- To assign a value to a macro variable.

Examples:

```
MAC1=MACRO/A=1.25,C  
CALL/MAC1,A=4.375,C=LN3
```

2.3.5.10 The Left And Right Parentheses “()”

The left and right parentheses are used as follows:

- To enclose the Arguments of a function.

Examples:

```
A      =COS(B)  
V1     =VECTOR/-SIN(20+5/60),0,COS(20+5/60)  
LNTHV=SQRT(I**2+J**2+K**2)
```

- To enclose elements in a scalar expression (this determines the [Evaluation Priority](#) in a mathematical equation).

Example:

```
A=(B+C)*D
```

- To enclose subscripts in a subscripted variable name.

Examples:

```
B(2)   =A*3  
A(J)   =B(2)+G  
CVA(1)=CURVE/P(1),THRU,P(20)  
SURF   /CVA(1),THRU,CVA(10)
```

- To enclose a Nested definition.

Example:

```
GO/(L1=LINE/X1,Y1,X2,Y2),PL1,L3
```

- To enclose the arguments in an IF statement.

Example:

```
IF(A .LT. 0) A=0
```

2 NCL LANGUAGE

2.3.5.11 The Left And Right Square Brackets “[]”

Special case of the use of square brackets within an **NCL** statement. The square brackets are used to specify either the U (for curves) or the UV values (for surfaces). These values identify where **NCL** will begin its iteration process. See [Section 6.5](#) in Motion Control Statements for further explanation of U and V values as related to motion.

Example:

```
d1=distf(pt1,sf1[.5,.5])
```

Note: This is a special case where square brackets are used within an **NCL** statement rather than to indicate an option.

2.3.5.12 The @ (At Sign)

Used to pass a scalar or a text variable to the postprocessor statements: PARTNO, [PPRINT](#), [INSERT](#).

2.3.5.13 The & (Ampersand Sign)

Used to concatenate [text elements](#) or text element with [Text Format](#) statement to form a new text element.

Example:

```
TXT1="String 1"  
TXT2="String 2"  
TXT3=TXT1 & TXT2 & "String 3"  
  
K      =2  
A1     =3  
ANGTXT="Angle"  
TXTFMT="%f, Cos %s = %-10.7f"  
TXT4   ="Value of K = "  
TXT5   =TXT4 & FORMAT(TXTFMT,K,ANGTXT,COS(A1))
```

2.3.6 Numbers And Computing

2.3.6.1 Integer

An integer is a string of digits which represents a whole number. It may be signed (+ or -) or unsigned. An integer may not contain a decimal point. Integers are normally used in an **NCL** Part Program as a loop counter or as a location in a subscripted array.

Examples:

<u>Valid</u>	<u>Invalid</u>
0	1.0 (Integer may not have decimal point)
-123	
+123456	

2.3.6.2 Real Numbers

A real number is a number that contains a Decimal Point.

A real number represents a number which contains a decimal fraction and may be signed (+ or -) or unsigned. Real numbers are normally used in an **NCL** Part Program to specify the size or location of a geometric entity or cutter. Numbers are limited to 9 digits to the left of the decimal point and 20 characters in total.

Examples:

<u>Valid</u>	<u>Invalid</u>
+123.456	10.1. (Only one decimal point allowed)
.0123	
2.0	

2.3.6.3 Arithmetic Operators And The Order Of Operation

<u>Operator</u>	<u>Computing Priority</u>
() parentheses	Priority 1 (Highest)
	functions
**	Priority 2
*	exponentiation
/	Priority 3
+	multiplication
-	Priority 4
	division
	Priority 5 (Lowest)
	addition
	subtraction

2 NCL LANGUAGE

Calculation within a statement is done in order of Priority; the highest priority functions are performed first. When computations are of equal priority, they will be performed from left to right.

- Parentheses - Priority 1 (Highest)

Everything enclosed in parentheses is calculated first in the order of priority as listed below. When more than one set of parentheses exists, each set is calculated from left to right and from innermost to outermost.

- Functions - Priority 2

All the valid Mathematical Functions above are calculated from left to right.

- Exponentiation - Priority 3

All exponent numbers are calculated from left to right.

- Multiplication and Division - Priority 4

All multiplication and division is calculated from left to right.

- Addition and Subtraction - Priority 5 (Lowest)

All addition and subtraction is calculated last from left to right.

2.3.6.4 Functions

NCL provides a group of miscellaneous functions that may be used in numerical expressions.

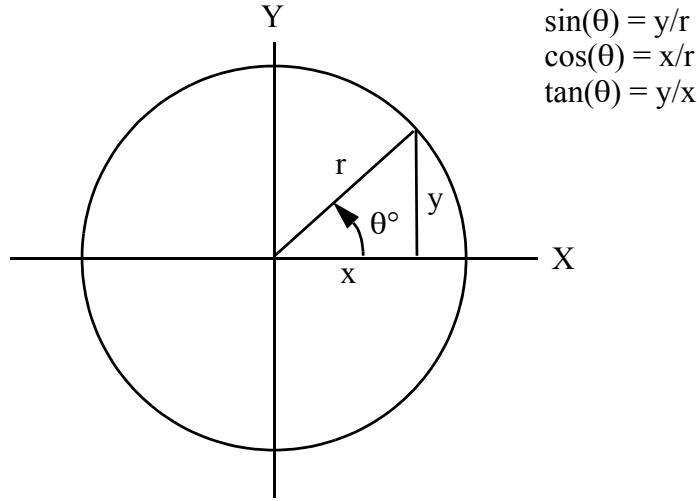
2.3.6.4.1 Trigonometric Functions

Function Syntax:

```
SIN (angle)
SINF
COS
COSF
TAN
TANF
```

Where: angle is in unit of degrees

The ratio of sides of a triangle in a rectangular coordinate system as shown below:



Examples:

- SIN (A1) – Returns the SINE of the angle (A1)
- COS (A2) – Returns the COSINE of the angle (A2)
- TAN (A3) – Returns the TANGENT of the angle (A3)

Function Syntax:

```

ASIN (scalar)
ASINF
ACOS
ACOSF
ATAN
ATANF
    
```

Examples:

- ASIN(n1) – Returns the angle (degrees) whose SINE is the scalar value of n1.
- ACOS(n2) – Returns the angle (degrees) whose COSINE is the scalar value of n2.
- ATAN(n3) – Returns the angle (degrees) whose TANGENT is the scalar value of n3.

Function Syntax:

```

ATAN2 (n1,n2)
ATAN2F
    
```

2 NCL LANGUAGE

Example:

`ATAN2 (n1 , n2) -` Returns the angle (degrees) whose TANGENT is n1 divided by n2.

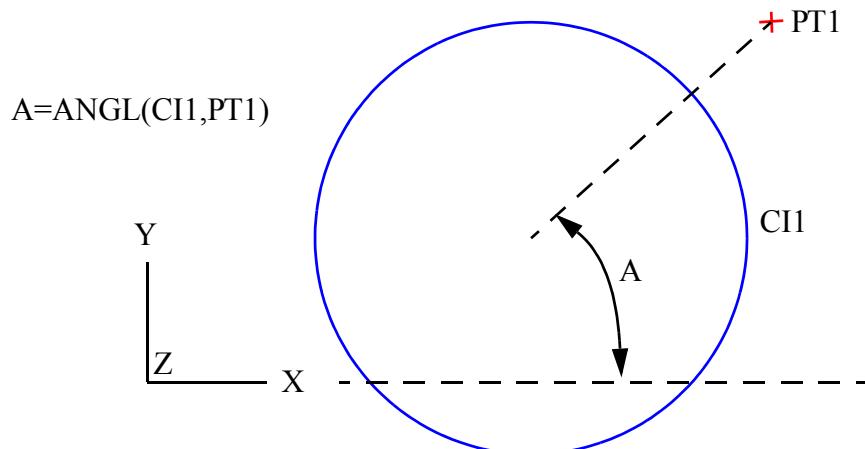
2.3.6.4.2 Angle Functions

Angles between any two lines, two planes, two vectors, two point-vectors, a vector and a point-vector, or the angle between the positive x-axis and a vector from the circle center to a specified point. The angle will be in the range from 0 and up to (not including) 360 degrees.

Function Syntax:

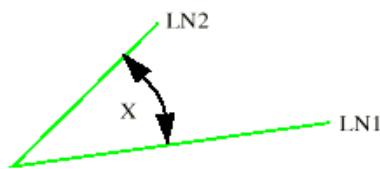
```
ANGL (line1,line2      )
ANGLF plane1,plane2
vector1,vector2
pntvec1,pntvec2
vector,pntvec
pntvec,vector
circle,point
```

Examples:

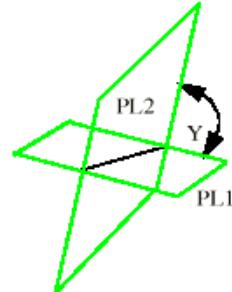


Examples:

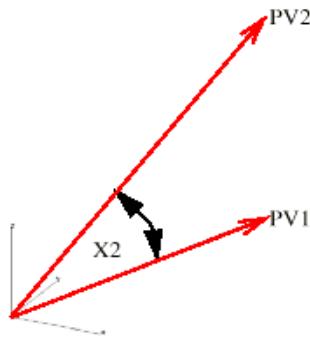
$$X = \text{ANGLF} (\text{LN1}, \text{LN2})$$



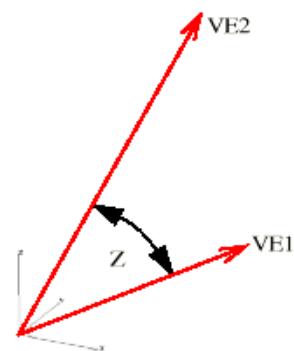
$$Y = \text{ANGLF} (\text{PL1}, \text{PL2})$$



$$X2 = \text{ANGLF} (\text{PV1}, \text{PV2})$$



$$Z = \text{ANGLF} (\text{VE1}, \text{VE2})$$



2.3.6.4.3 Maximum And Minimum Functions

Return the maximum or minimum value of a list of scalar quantities (or variables that represent scalar quantities).

Function Syntax:

```
MAX1F(scalar1, scalar2, ... scalarn)
MIN1F
```

Examples:

```
N1=MAX1F(2.62,3.916,1.5,3.125)
then N1=3.916
```

```
N2=MIN1F(4.125,3.6945,8.654)
then N2=3.6945
```

2 NCL LANGUAGE

2.3.6.4.4 Sign Function

Returns the value of the first scalar quantity (or variable that represents a scalar quantity) with the sign of the second.

Function Syntax:

```
SIGNF(scalar1,scalar2)
```

Example:

```
N1=SIGNF(3.625,-4.793)
      then   N1=-3.625
```

2.3.6.4.5 Distance Function

The distance function obtains the distance between two geometry, e.g. two points, two point-vectors, two parallel lines, a point and a plane, a point and a line, two parallel planes or a point and a surface, etc. The optional [U, V] values (with point or point-vector to surface) may be used to start the iteration process at a specific UV location on the surface. See [User Specified U and V Values for Curves and Surfaces](#) in Continuous Motion Statements.

Function Syntax:

```
DIST (entity_1,entity_2)
DISTF
or
DIST /entity_2,entity_1
DISTF
```

Examples:

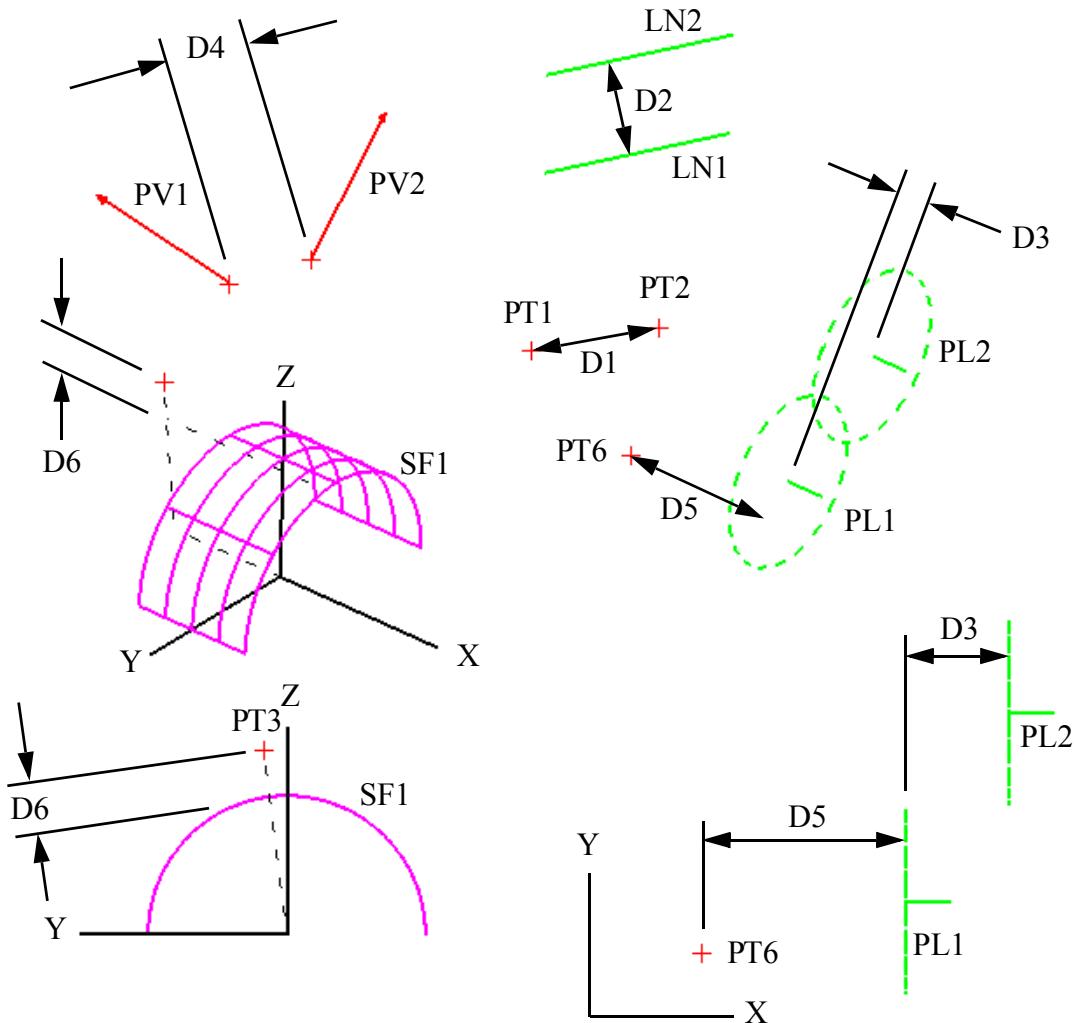
D1=DIST(PT1,PT2)	or	D1=DIST/PT1,PT2
D1=DISTF(PT1,PT2)	or	D1=DISTF/PT1,PT2
D2=DIST(LN1,LN2)	or	D2=DIST/LN1,LN2
D2=DISTF(LN1,LN2)	or	D2=DISTF/LN1,LN2
D3=DIST(PL1,PL2)	or	D3=DIST/PL1,PL2
D3=DISTF(PL1,PL2)	or	D3=DISTF/PL1,PL2
D4=DIST(PV1,PV2)	or	D4=DIST/PV1,PV2
D4=DISTF(PV1,PV2)	or	D4=DISTF/PV1,PV2

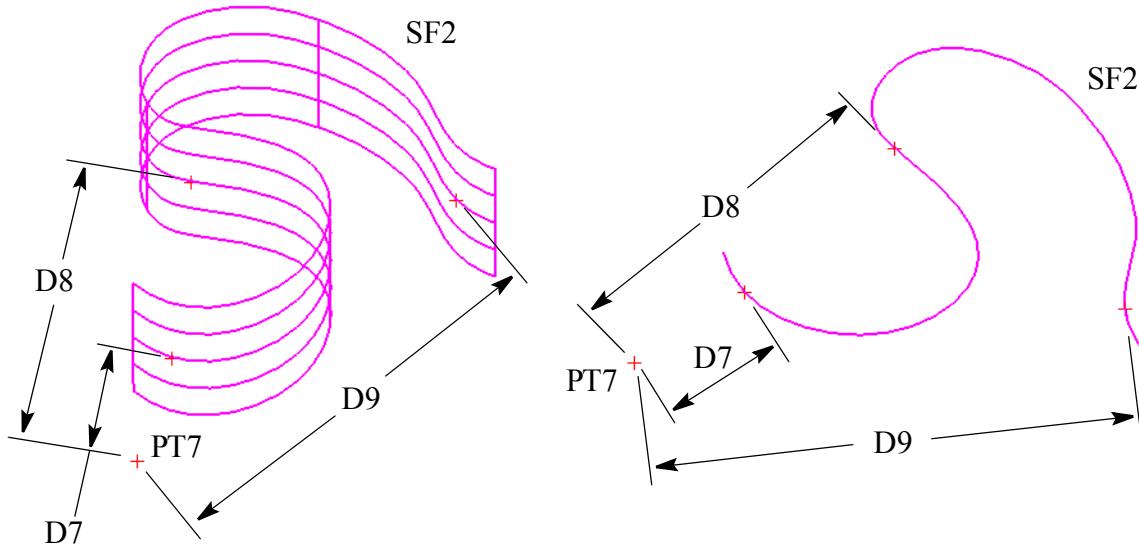
D5=DIST(PT6, PL1)	or D5=DIST/PT6, PL1
D5=DISTF(PT6, PL1)	or D5=DISTF/PT6, PL1
D6=DIST(PT3, SF1, [u,v])	or D6=DIST/PT3, SF1, [u,v]
D6=DIST(FPT3, SF1, [u,v])	or D6=DISTF/PT3, SF1, [u,v]

Note that “D6” is a special case of the use of square brackets within an **NCL** statement.

NOTE: Only the DIST(a,b) or DISTF(a.b) syntax can be used within a nested statement or inside an **NCL** function.

Examples:





$D7 = \text{DIST}/\text{PT7}, \text{SF2}[0,1]$
 $D8 = \text{DIST}/\text{PT7}, \text{SF2}[0.5,1] \text{ or } \text{DIST}/\text{PT7}, \text{SF2}$
 $D9 = \text{DIST}/\text{PT7}, \text{SF2}[1,1]$

2.3.6.4.6 TDISTF Function

This function returns the distance between the origin of a point-vector to a geometry entity along its vector direction.

Function Syntax:

```

TDIST (point-vector,geo-entity,1) )
TDISTF point,vector 2
      x,y,z,i,j,k -1
      -2
      near-point
    
```

Where:

“geo-entity” can be a line, circle, curve, surface, or a solid.

“1” specifies the closest distance in the point-vector direction.

“2” specifies the furthest distance in the point-vector direction.

“-1” specifies the closest distance in the opposite direction of the point-vector.

“-2” specifies the furthest distance in the opposite direction of the point-vector.

“near-point” specifies the distance to the near-point intersection between the point-vector and the geo-entity.

Note: The TDIST function will always look in the specified direction first. If no intersection is found then the direction is reversed and TDIST will look for an intersection in the new direction. If the intersection that is returned is in the reversed direction the distance will be returned as a negative value.

Distance calculations with wireframe geometry (lines, circles, curves) will be done using a 2D intersection method, while the distance calculations with surfaces, solids, and planes will use a 3D intersection method.

2.3.6.4.7 Absolute Value Function

The value of a scalar without regard to its sign.

Function Syntax:

ABS (scalar)

ABSF

Example:

```
S1=ABS( (22-46)*365/400-SQRT(689) )
      then S1=48.1488095
```

2.3.6.4.8 Square Root Function

A mathematical function returning the square root of a scalar value.

Function Syntax:

SQRT(scalar)

Example:

```
F1=SQRT(16)
      then F1=4
```

2.3.6.4.9 Cube Root Function

A mathematical function returning the cubic root of a scalar value.

Function Syntax:

```
CBRTF(scalar)
```

Example:

```
F1=CBRTF(27)
      then F1=3
```

2.3.6.4.10 Natural Logarithm Function

A scalar value based on the logarithm to the base of an irrational number denoted by e, whose approximate value is 2.71828...

Function Syntax:

```
LOG (arithmetic function)
LOGF
```

Example:

```
X=LOG(6*24+25)
      then X=5.1298987
```

2.3.6.4.11 Common Logarithm Function

A scalar value based on the logarithm to the base 10 ($\log_{10} X$).

Function Syntax:

```
LOG10F(arithmetic function)
```

Example:

```
Y=LOG10F(6*24+25)
      then Y=2.2278867
```

2.3.6.4.12 Exponent Function

A scalar value that represents the value of e (2.71828183...) raised to the power of a scalar x, i.e. e^x . Where x can be any real number.

Function Syntax:

```
EXP (scalar)
EXPF
```

Example:

```
N=EXPF(6)
      then N=403.4287935
```

2.3.6.4.13 Length Of A Geometric Entity Function

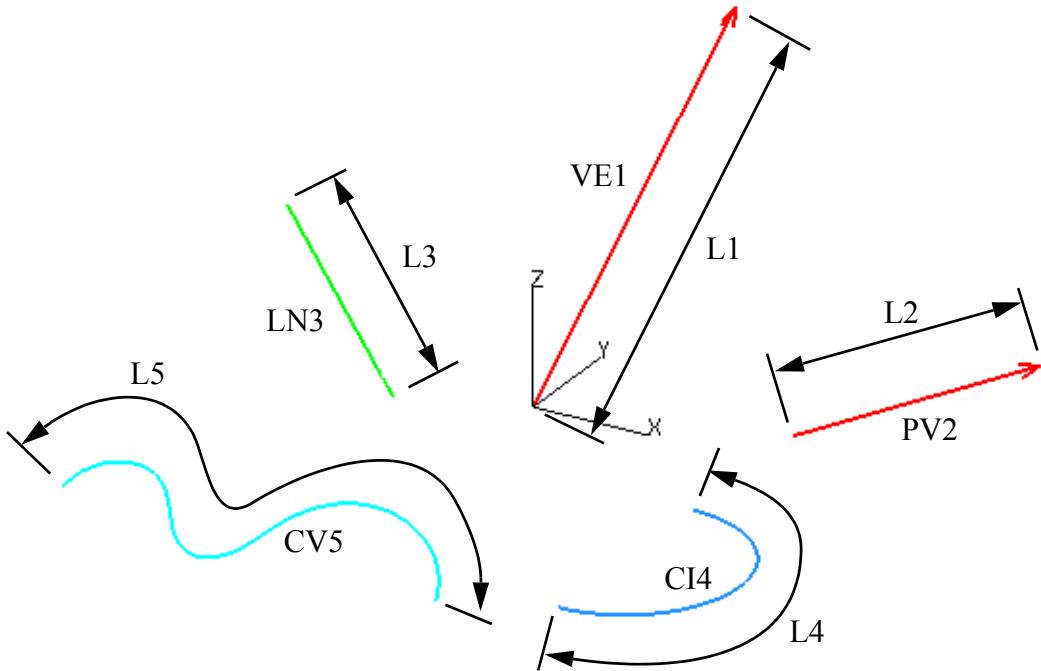
Returns the length of a vector, point-vector, line, circle, or curve.

Function Syntax:

```
LNTH (vector)
LNTHF pntvec
      line
      circle
      curve
```

Examples:

```
L1=LNTHF(VE1)
L2=LNTHF(PV2)
L3=LNTHF(LN3)
L4=LNTHF(CI4)
L5=LNTHF(CV5)
```



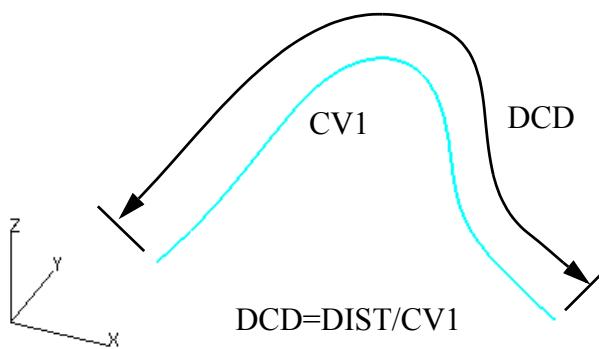
2.3.6.4.14 Length Of A Three Dimensional Curve Function

The actual length along a specified 3-D space curve is assigned to a scalar variable.

Function Syntax:

DIST /curve	or	DIST (curve)
DISTF		DISTF

Example:



NOTE: Only the DIST(curve) or DISTF(curve) syntax can be used within a nested statement or inside an **NCL** function.

2.3.6.4.15 Integer Function

Returns the integer portion of a real scalar value or the nearest integer of a scalar value.

Function Syntax:

```
INT  (scalar)
INTF
NINT
NINTF
```

Examples:

```
N=INT(6.6)
      then N=6.000000 (the Integer portion of a scalar)

N=NINT(6.6)
      then N=7.000000 (the Nearest Integer of a scalar)
```

2.3.6.4.16 Dot Product Function

Returns a value equal to the cosine of the true angle between two vectors, two point-vectors, a point and a point-vector, or two planes.

Function Syntax:

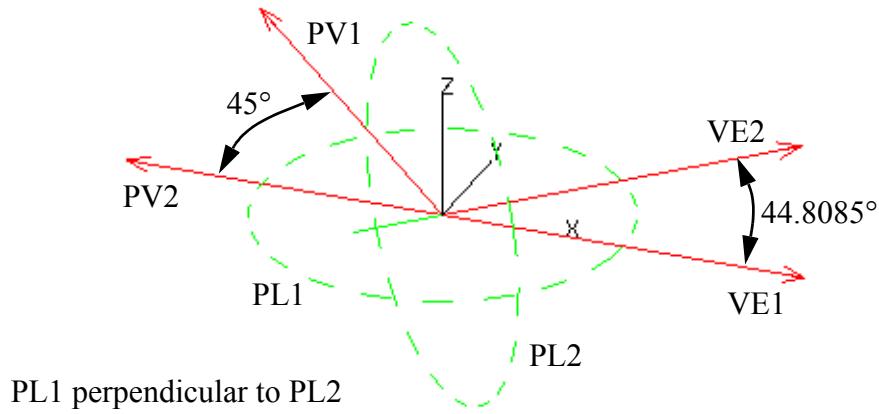
```
DOT (vector1,vector2)
DOTF pntvec1,pntvec2
      vector1,pntvec1
      pntvec2,vector2
      plane1,plane2
```

Examples:

```
COSA1=DOT(VE1,VE2)
      then COSA1=.7094657

COSA2=DOT(PV1,PV2)
      then COSA2=.7071068

COSA3=DOT(PL1,PL2)
      then COSA3=0
```



2.3.6.4.17 Canonical Form Function

Returns the value of an element in the canonical form of a geometric expression. Where the first value specified indicates the form of geometry and the second indicates the element desired (by its position). This function may also be used to obtain the current individual element value of the [CONTCT](#), [FEDRAT](#), [NUMPTS](#), [MAXANG](#), [MAXDP](#), [TOLER](#), [CUTTER](#), [THICK](#) and [UNITS](#) setting.

Function Syntax:

```
CAN (entity,position)
CANF word
```

Where:

position: the positional number of the element that needs to be extracted.

entity: a valid geometric label

word: [CONTCT](#) 1 value available: 1 if ON and -1 if OFF

[FEDRAT](#) 1 value available: the primary feed rate

[NUMPTS](#) 1 value available

MAXANG	1 value available
MAXDP	1 value available
TOLER	4 values available: chordal tolerance, positional tolerance, AUTOST positional tolerance, AUTOST angular tolerance
THICK	7 values available: part surface, drive surface and 5 check surfaces
UNITS	1 value available: 1 for INCHES and 2 for MM
CUTTER	Up to 6 values available: in the order they are input in the CUTTER statement (except the APT 7 parameter CUTTER).
	Note: If a *SET/VER command is issued with a version of less than 9.6, the values returned are in the order they are stored internally rather than in the order they are input in the CUTTER statement.
	APT 7 parameter cutter input format would be converted to the NCL CUTTER format before the values can be obtained.
PSEUDO	Up to 6 values available. The values returned represent the parameters of the CUTTER/DISPLAY,dia,... command, if one is defined, or if a profile or symbol is used to define a mill cutter, then NCL will estimate the largest diameter and height of the cutter as represented by the geometry defining the cutter shape. If the cutter type is a blade or a lathe bit, or a display cutter had not been defined, then the actual values of the CUTTER/dia,.. command will be returned.
SHANK	Up to 5 values available. The values returned represent the parameters of the CUTTER/DISPLAY,SHANK command. The values returned as follows: Mill: Diameter, Height, Angle, Z-offset

2 NCL LANGUAGE

Lathe: X-width, Y-length, Z-height, X-offset, Y-offset

The diameter and height will be estimated when a profile or symbol is used to define a mill tool. The width, length, and height will be estimated when a profile or symbol is used to define a lathe tool.

HOLDER

Up to 5 values available. The values returned represent the parameters of the CUTTER/DISPLAY,HOLDER command. The values returned as follows:

Mill: Diameter, Height, Angle, Z-offset

Lathe: X-width, Y-length, Z-height, X-offset, Y-offset

The diameter and height will be estimated when a profile or symbol is used to define a mill tool. The width, length, and height will be estimated when a profile or symbol is used to define a lathe tool.

Examples:

```
CONTCT/ON
A1      =CAN(CONTCT,1)
then A1=1

CUTTER/0.5,0.125,2
DIA     =CAN(CUTTER,1)
then DIA=0.5
COR     =CAN(CUTTER,2)
then COR=0.125
HGT     =CAN(CU,3)
then HGT=2

CUTTER/DISPLAY,HOLDER,0.5,3,10
DIA     =CAN(HOLDER,1)
then DIA=0.5
HGT     =CAN(HOLDER,2)
then HGT=3
SANG   =CAN(HOLDER,3)
then SANG=10
```

```

PT1 =POINT/2,2
ELPT1=CAN(PT1,1)
    then ELPT1=2
ELPT2=CAN(PT1,2)
    then ELPT2=2
ELPT3=CAN(PT1,3)
    then ELPT3=0

```

X,Y,Z coordinate values of PT1


```

PT2 =POINT/0,0
PV1 =PNTVEC/PT2,PT1
ELPV1=CAN(PV1,1)
    then ELPV1=0
ELPV2=CAN(PV1,2)
    then ELPV2=0
ELPV3=CAN(PV1,3)
    then ELPV3=0
ELPV4=CAN(PV1,4)
    then ELPV4=2
ELPV5=CAN(PV1,5)
    then ELPV5=2
ELPV6=CAN(PV1,6)
    then ELPV6=0

```

X,Y,Z start point coordinates

I, J, K values


```

LN1 =LINE/4,0,4,1
ELLN1=CAN(LN1,1)
    then ELLN1=4
ELLN2=CAN(LN1,2)
    then ELLN2=0
ELLN3=CAN(LN1,3)
    then ELLN3=0
ELLN4=CAN(LN1,4)
    then ELLN4=0
ELLN5=CAN(LN1,5)
    then ELLN5=1
ELLN6=CAN(LN1,6)
    then ELLN6=0

```

X,Y,Z start point coordinates

X,Y,Z delta values


```

PL1 =PLANE/1,0,0,2
ELPL1=CAN(PL1,1)
    then ELPL1=1
ELPL2=CAN(PL1,2)
    then ELPL2=0
ELPL3=CAN(PL1,3)
    then ELPL3=0

```

X,Y,Z vector components normal to the plane

2 NCL LANGUAGE

```
ELPL4=CAN(PL1, 4) → offset distance from origin
then ELPL4=2

CI1 =CIRCLE/-2,1,2
ELCI1=CAN(CI1,1)
then ELCI1=-2
ELCI2=CAN(CI1,2)
then ELCI2=1
ELCI3=CAN(CI1,3)
then ELCI3=0
ELCI4=CAN(CI1,4)
then ELCI4=0
ELCI5=CAN(CI1,5)
then ELCI5=0
ELCI6=CAN(CI1,6)
then ELCI6=1
ELCI7=CAN(CI1,7)
then ELCI7=2
ELCI8=CAN(CI1,8)
then ELCI8=0
ELCI9=CAN(CI1,9)
then ELCI9=0
ELCI10=CAN(CI1,10)
then ELCI10=0
ELCI11=CAN(CI1,11)
then ELCI11=0
```

The diagram illustrates the mapping of NCL variables to geometric parameters for a circle. It consists of two columns of code snippets and arrows pointing to their corresponding interpretations:

- Column 1:** ELPL4=CAN(PL1, 4)
then ELPL4=2
- Column 2:** → offset distance from origin

- Column 1:** CI1 =CIRCLE/-2,1,2
ELCI1=CAN(CI1,1)
then ELCI1=-2
- Column 2:** → X,Y,Z center

- Column 1:** ELCI2=CAN(CI1,2)
then ELCI2=1
- Column 2:** → offset distance from origin

- Column 1:** ELCI3=CAN(CI1,3)
then ELCI3=0
- Column 2:** → X,Y,Z vector components normal to the circular plane

- Column 1:** ELCI4=CAN(CI1,4)
then ELCI4=0
- Column 2:** → radius of circle

- Column 1:** ELCI5=CAN(CI1,5)
then ELCI5=0
- Column 2:** → limit plane components

- Column 1:** ELCI6=CAN(CI1,6)
then ELCI6=1
- Column 2:** → limit plane components

- Column 1:** ELCI7=CAN(CI1,7)
then ELCI7=2
- Column 2:** → limit plane components

- Column 1:** ELCI8=CAN(CI1,8)
then ELCI8=0
- Column 2:** → limit plane components

- Column 1:** ELCI9=CAN(CI1,9)
then ELCI9=0
- Column 2:** → limit plane components

- Column 1:** ELCI10=CAN(CI1,10)
then ELCI10=0
- Column 2:** → limit plane components

- Column 1:** ELCI11=CAN(CI1,11)
then ELCI11=0
- Column 2:** → limit plane components

As it can be seen from the above examples, the canonical form for each geometric expression differs. However, during the interactive session, the canonical form can be displayed for any desired entity by clicking the graphic interface as follows:

STATUS > GEOMETRY > “The Desired Entity”

2.3.6.4.18 Num Function

This function returns the number of elements in a pattern or a composite curve, the highest subscript value assigned to a reserved variable, or the number of elements in a DATA statement. See Chapter 4, Geometry Control Statements for a complete description of the [DATA](#) statement.

Function Syntax:

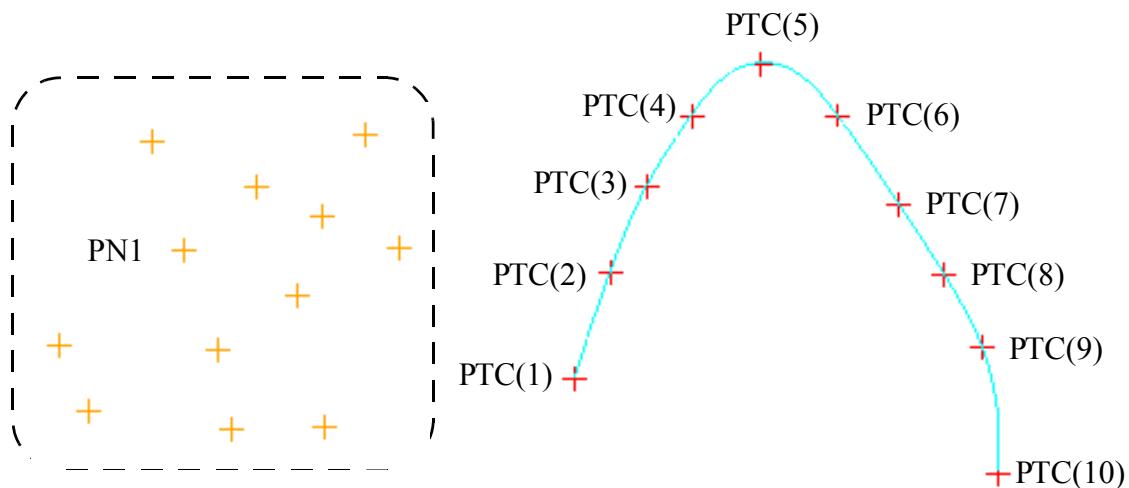
```
NUM (name-of-pattern)
NUMF name-of-composite-curve
      name-of-array
      name-of-DATA-statement
```

Examples:

```
N1=NUMF ( PN1 )
      then N1=12
```

```
N2=NUMF ( PTC )
      then N2=10
```

```
T1=DATA/.5,.25,2,rpm1,dir
N3=NUMF(T1)
      then N3=5
```



2.3.6.4.19 Type Function

This function returns the type value of a variable (n).

Function Syntax:

```
TYPE (n)
TYPEF
```

Where:

<u>Variable (n) Type</u>	<u>Value Returned</u>
A scalar less than or equal to 1.0	1
A scalar greater than 1.0	2
A vocabulary word.	3
A quoted text string (DATA element)	4
An unknown variable.	5
A point.	7
A vector.	8
A line.	9
A plane.	10
A circle.	11
A curve.	12
A surface.	13
A matrix.	14
A macro.	15
A macro parameter.	16
A label.	17
A reserved identifier.	18
An index range identifier.	19
A nshape.	22
A pattern.	24
A point-vector.	25
A pocket.	26
A data statement.	27
A Text Variable.	28
An Annotation.	34
A Symbol	35
A Symbol Placement	36
A Solid.	37

Examples:

```
N1=-1
X1=TYPE(N1)
    then X1=1  (a scalar <= 1.0)

N2=1.01
X2=TYPE(N2)
    then X2=2  (a scalar > 1.0)

CI1=CIRCLE/1,2,1
X3=TYPE(CI1)
    then X3=11  (a circle)

PN1=PATTERN/LINEAR,PT1,PT2,8
X4=TYPE(PN1)
    then X4=24  (a pattern)

D1=DATA/"text string",GOTO,1,2
X5=TYPE(D1[1])
    then X5=4  (a quote text string)

D2=D1[1]
X6=TYPE(D2)
    then X6=28  (a text variable)
```

2.3.6.4.20 XTYPE Function

This function returns the geometry subtypes for curves, surfaces, patterns and visual solids. Function Syntax:

```
XTYPE (geometry)
XTYPEF
```

Where:

<u>Geometry</u>	<u>Value Returned</u>
-----------------	-----------------------

Curves

NCL Curve	1201
B-Spline	1202
Surfaces Spline	1203
Composite Curve	1204

2 NCL LANGUAGE

<u>Pattern</u>	
Point-Pattern	2401
Point-Vector Pattern	2402
<u>Surface</u>	
NCL Surface	1301
NURB Surface	1302
Revolved Surface	1303
Mesh Surface	1304
Quilt Surface	1305
Trimmed Surface	1306
<u>Solid</u>	
Box Solid	3701
Cylinder Solid	3702
Tours Solid	3703
Sphere Solid	3704
Cone Solid	3705
Extruded Solid	3706
Contour Solid	3707
Revolved Solid	3708
STL Solid	3709

All other geometry types will return the same value as the TYPE function.

2.3.6.4.21 Vocabulary Function

This function returns the numerical value of a vocabulary word or its synonym, otherwise it returns a value of 0.

Function Syntax:

VOCABF (Vocabulary_Word)

The numerical values of the vocabulary words are defined in the following files:

C:\NCCS\NCL101\interface\nclvoc.ncl
C:\NCCS\NCL101\interafce\nclvoc.syn

You can also refer to “[Appendix B: Vocabulary](#)” for the numerical values.

Examples:

1) A = VOCABF (LOADTL)

This scalar assignment statement assigns the numerical value “1055” to the numerical scalar variable A.

```
2) TCH=MACRO/TLN, CUTDIR=OMIT
   LOADTL/TLN
   IF ( VOCABF(CUTDIR) 'NE' VOCABF( OMIT ) ) $
   CUTCOM/CUTDIR, TLN
   TERMAC
```

If the above macro is called and the parameter CUTDIR does not equal the vocabulary word “OMIT”, this macro call will output the following postprocessor statements

```
LOADTL/TLN
CUTCOM/CUTDIR, TLN
```

to the .cl or .as file. Otherwise, only the postprocessor statement

```
LOADTL/TLN
```

will be output to the .cl or .as file.

2.3.7 Text Control

Text can be edited, formatted in **NCL** through the use of “Text String”, “Text Variable”, “Text Element” and “Text Functions”.

2.3.7.1 Text String

Text string is the basic element of Text Handling. A text string consists of any valid characters enclosed in double quotes. A maximum of 64 characters is allowed in a text string.

2.3.7.2 Text Variable

A text variable may contain up to 255 characters while a text string by itself can only contain up to 64 characters. The syntax for defining a text variable is as follows:

```
label = text-element [ & text-element [...] ]
           format-function    format-function
```

2 NCL LANGUAGE

The **ampersand** (&) is used to concatenate text elements and the output from format functions together.

A text variable consists of a combination of text element, which can be a text string or other text element, and format-function.

Text variables are stored in the unibase and may be subscripted. They can also be used as macro arguments.

A text variable may be inserted into a PARTNO, **PPRINT** or **INSERT** statement by preceding it with an **at sign** (@) in the same manner that a scalar variable can be inserted into these statements.

Text variables can be redefined without **CANON** being set to ON.

2.3.7.3 Text Element

A text element consists of either a text string or a text variable followed by an optional substring clause.

Text elements may be used in:

- **LOADU**, **SAVEU**, **LOADPP**, **SAVEPP**, **UBFN**, **INCLUD**, **READ** statements to specify the filename.
- **CUTTER** statements to specify a tool or symbol library.
- **DEFNAM** statements to specify the default name.
- **PROMPT** statements to specify the prompt string. A text variable cannot be prompted.

If a text element is used in any statement in place of a minor word or on the right hand side of a statement in place of a variable, **NCL** will substitute the contents of the text element for the minor word or variable. This in effect, treats text variables in a manner similar to macro variables. Text substitution is not done on the left hand side of an equals sign or slash or for a label definition, but it is done when a label is used in a **JUMPTO** or **IF** statement. Scalar variables are substituted but literal numbers or scalar expressions are not. A text string is substituted when used in place of a macro name in a call statement but is not substituted when used as a macro argument.

2.3.7.4 Sub-String Clause

```
Text-element-label [n : [m] ]
```

A substring clause consists a start character scalar value “n” followed by a colon and an optional end character scalar value “m” all enclosed in square brackets “[]” and is used to select a portion of a text string or text variable. If the end character scalar value is not present, it default to the length of the text string or text variable. If the start value is greater than the end value or greater than the length of the string, the result will be a string containing no characters. If the end value is greater than the length of the string, it will be set to the length of the string.

Example:

text1[1:5]	- The first five characters of text1
text1[n+1:]	- Characters n+1 through the last character of text1.
"ABCDEFG" [3:5]	- Character 3 through 5 of the text string "ABCDEFG", i.e. "CDE"

2.3.7.5 Format Function

Function Syntax:

```
FORMAT(format-string,variable-list)
```

The variable list is a list of scalar expressions and text elements. The format-string is a text element.

For each scalar expression in the variable list, the format string must contain a corresponding format code as follows:

```
% [-] [+][0][w][.][d]f
```

- If the minus sign (-) is present, the value will be left justified, otherwise it will be right justified.
- If the plus sign (+) is present, positive numbers will be preceded by a plus sign, otherwise they will be preceded by a space. Negative numbers are always preceded by a negative sign.

2 NCL LANGUAGE

- If the zero (0) is present, the value will be zero filled, otherwise it will be blank filled.
- The (w) is an integer and specified the total field width, including a preceding plus or minus sign, the number of digits to the left of the decimal point, the decimal point, and the number of digits to the right of the decimal point. If it is not present, the field width will be set to the minimum required to hold the number. If it is less than what should be required, **NCL** will default to whatever number of digits needed to be output.
- The (d) is an integer and represents the number of digits to the right of the decimal point. If it is zero or not present, there will be no decimal point and no digits to the right of the decimal point. If the decimal point and the (d) are not present, 6 digits will be output to the right of the decimal point. If the numerical value next to the last output digit is larger than or equal to 5, the last output digit will up by 1.

Example:

```
A1 =123.456
FM1="%10.2f"
TX1="VALUE OF A1 IS " & FORMAT(FM1,A1)
    TX1 contains the string "VALUE OF A1 IS      123.46"

FM2="%-10.2f"
TX2="VALUE OF A1 IS " & FORMAT(FM2,A1)
    TX2 contains the string "VALUE OF A1 IS 123.46      "

FM3="%+010.2f"
TX3="VALUE OF A1 IS " & FORMAT(FM3,A1)
    TX3 contains the string "VALUE OF A1 IS +000123.46"

FM4="%.2f"
TX4="VALUE OF A1 IS " & FORMAT(FM4,A1)
    TX4 contains the string "VALUE OF A1 IS 123.46"

FM5="%1.6f"
TX5="VALUE OF A1 IS " & FORMAT(FM5,A1)
    TX5 contains the string "VALUE OF A1 IS 123.456000"
```

For each text-element in the variable list, the format string must contain a corresponding format code as follows:

% [-] [w] s

If the minus sign (-) is present, the string will be left justified, otherwise it will be right justified. If the w is present, it specifies the total field width. Otherwise the field width will be set to the length of the corresponding string.

The format codes may be separated by any valid characters, which will be transferred to the output string unchanged. Two consecutive percent sign characters will be transferred to the output string as one percent sign.

Example:

```
A2=45
FM6="%1f %s"
TX6="ANGLE IS " & FORMAT(FM6,A2,"DEGREES")
TX6 contains the string "ANGLE IS 45.0 DEGREES"

FM7="%1f %10s"
TX7="ANGLE IS " & FORMAT(FM7,A2,"DEGREES")
TX7 contains the string "ANGLE IS 45.0          DEGREES"

FM8="%1f %-10s"
TX8="ANGLE IS " & FORMAT(FM8,A2,"DEGREES")
TX8 contains the string "ANGLE IS 45.0 DEGREES   "
```

2.3.7.6 Text Functions

2.3.7.6.1 The LNTH Function

Function Syntax:

```
LNTH (text-element)
LNTHF
```

This function returns the number of characters in the text-element.

Example:

```
TXT1="TOOL NUMBER:"
A1 =LNTH(TXT1)
then A1=12

A2 =LNTH("TEXT STRING")
then A2=11
```

2 NCL LANGUAGE

2.3.7.6.2 The FINDEX Function

Function Syntax:

```
FINDEX(text-element-1, text-element-2)
```

This function returns the character position in the text-element-1 where the first occurrence of text-element-2 begins, or zero if text-element-1 does not contain text-element-2.

By default, the FINDEX function is not case sensitive, i.e. upper and lower case letters are treated as being the same. This can be changed by entering the [*SET/CASE](#) command. Entering the [*RESET/CASE](#) command will revert it back to the default state.

Example:

```
TXT1="OPERATION 1, LOOP 1, Sub-Loop 1"  
TXT2="1"  
A1 =FINDEX(TXT1,TXT2)  
then A1=11  
  
TXT3="NUMBER"  
A2 =FINDEX(TXT1,TXT3)  
then A2=0  
  
TXT4="Loop"  
A3 =FINDEX(TXT1,TXT4)  
then A3=14  
  
*SET/CASE  
A4 =FINDEX(TXT1,TXT4)  
then A4=26
```

2.3.7.6.3 The RINDEX Function

Function Syntax:

```
RINDEX(text-element-1, element-2)
```

This function returns the character position in text-element-1 where the last occurrence of text-element-2 begins, or zero if text-element-1 does not contain text-element-2.

By default, the RINDEX function is not case sensitive, i.e. upper and lower case letters are treated as being the same. This can be changed by entering the ***SET/CASE** command. Entering the ***RESET/CASE** command will revert it back to the default state.

Example:

```
TXT1="OPERATION 1, LOOP 1, Sub-Loop 1"
TXT2="1"
A1 =RINDEX(TXT1,TXT2)
then A1=31

TXT3="NUMBER"
A2 =RINDEX(TXT1,TXT3)
then A2=0

TXT4="LOOP"
A3 =RINDEX(TXT1,TXT4)
then A3=26

*SET/CASE
A4 =RINDEX(TXT1,TXT4)
then A4=14
```

2.3.7.6.4 The STRCMP Function

Function Syntax:

```
STRCMP(text-element-1, text-element-2)
```

This function returns a value of 1 if text-element-1 is exactly the same as text-element-2, otherwise a value of -1 will be returned.

By default, the STRCMP function is not case sensitive, i.e. upper and lower case letters are treated as being the same. This can be changed by entering the ***SET/CASE** command. Entering the ***RESET/CASE** command will revert it back to the default state.

Example

```
TXT1="OPERATION 1, LOOP 1"
TXT2="LOOP 1"
A1 =STRCMP(TXT1[14:],TXT2)
then A1=1
```

2 NCL LANGUAGE

```
TXT3=LNTH( "TEXT STRING")
TXT4="TEXT"
A2 =STRCMP(TXT3,TXT4)
then A2=-1

TXT5=LNTH( "TEXT STRING")
TXT6="Text"
A3 =STRCMP(TXT5[1:4],TXT6)
then A3=1

*SET/CASE
A4 =STRCMP(TXT5[1:4],TXT6)
then A4=-1
```

2.3.7.7 The TEXTF Function

Function Syntax:

```
TEXTF(label)
```

This function converts a geometry label, scalar label, data label or vocabulary word into a text string. The text string will be all upper case. If an unidentified label is passed to this function, this function converts it to a text string too.

Example

```
CI1 =CIRCLE/0,0,2
TXT1=TEXTF(CI1)
then TXT1="CI1"

TXT2=TEXTF(Unknown)      $$ Unknown not defined
then TXT2="UNKNOWN"

a1 =30
TXT3=TEXTF(a1)
then TXT3="A1"
```

2.3.7.8 Text Entity Removal

Text entity can be removed by using the REMOVE command. See [Section 4.17](#) in Geometry Control for the detailed interaction between REMOVE command and text entity.

2.3.7.9 Text String/Variable Evaluation

NCL has the ability to evaluate text strings or text variables and use them as regular commands.

The evaluation occurs when a text string or a text variable is enclosed by a pair of curly braces “{}”.

Examples:

```
1. PL1    =PLANE/0,1,0,1  
S1      ="PL1"  
PL2    =PLANE / PARREL, {S1}, YLARGE, 1
```

above statement is interpreted as:

PL2 =PLANE / PARREL, PL1, YLARGE, 1

```
2. TX2    ="1,2"  
PT1    =POINT/ {TX2}, 3
```

above statement is interpreted as

PT1 =POINT/1,2,3

```
3. TX3    ="2,3"  
ABC    ="{TX3}"  
PT2    =POINT / PT1, {ABC}, -1
```

above statement is interpreted as

PT2 =POINT/PT1,2,3,-1

Notice that in above example the evaluation occurs twice: first the text variable ABC is evaluated, and the command is internally read as

POINT/PT1, {TX2}, -1

then the text variable TX2 is evaluated and the command is interpreted as

POINT/PT1,1,2,-1

```
5. T1    ="PL1=PLANE/0,0,1,0"  
T2    ="-2,-1,0"  
T3    ="2,-1,0"  
T4    ="POINT/ {T2}"
```

2 NCL LANGUAGE

```
T5      ="POINT/ {T3}"
T6      ="LINE/ ({T4}), ({T5})"
T7      ="-1.5,2,0"
T8      ="-1.5,-3,0"
T9      ="LINE/ {T7}, {T8}"
T10     ="ON"
T11     ="TO"
T12     ="GO/ {T10}, ({T9}), ({T1}), {T11}, ({T6})"
{T12}
```

above statement is interpreted as

```
GO/ON, (LINE/-1.5,2,0,-1.5,-3,0)           $
(PL1=PLANE/0,0,1,0),TO,                      $
(LINE/(POINT/-2,-1,0),(POINT/2,-1,0))
```

Notice that in above example some of the evaluations occur in several steps: first the expression $\{T12\}$ is evaluated, and the command is internally read as

```
GO/ON, (LINE/ {T7}, {T8}),                   $
(PL1=PLANE/0,0,1,0),TO,                      $
(LINE/ ({T4}), ({T5}))
```

then the above expression is evaluated again and the command is internally read as

```
GO/ON, (LINE/-1.5,2,0,-1.5,-3,0),           $
(PL1=PLANE/0,0,1,0),TO,                      $
(LINE/(POINT/{T2}),(POINT/{T3}))
```

then the above expression is evaluated again and the command is interpreted as

```
GO/ON, (LINE/-1.5,2,0,-1.5,-3,0)           $
(PL1=PLANE/0,0,1,0),TO,                      $
(LINE/(POINT/-2,-1,0),(POINT/2,-1,0))
```

2.3.8 Expressions

An expression is a valid **NCL** statement used to define one of the following:

- A [Number](#)
- A [Geometric Element](#)
- A [Macro](#)

- A [Matrix](#)
- A [Text String](#)

2.3.8.1 Scalar Expressions

Scalar expressions may be composed of arithmetic operators, real numbers, integers, scalar variables, and function designators. Parentheses are used to signify order of precedence of arithmetic operators. The maximum number of arithmetic operators in a single expression is 20. **NCL** allows computing expressions not only in statements by themselves but also nested within other types of statements where a number or scalar identifier would normally appear.

Examples of Scalar Expressions:

<u>Expression</u>	<u>Value</u>
1	1
(8)	8
-16+20	4
-(16+20)	-36
1-6/3	-1
((1.2+3.4)/.5)	9.2
SQRT(2)	1.41421356
A+(B/2)	
INT(3.2)	3
INT(B+C)	

2.3.8.2 Geometric Expressions

Geometric expressions are valid **NCL** statements used to define the various geometric entities that are stored in the mathematical computer model which describes the part to be machined. Each geometric expression is a stand-alone statement which must include all the information necessary to define a “unique” geometric entity. The information must be stated in a specific sequential order (a “Fixed Format”) **NCL** will attempt to “match” the information specified in a geometric expression to a valid fixed format for the geometric entity to be defined. **NCL** supports several Fixed Formats for each of the following geometry types (see section titled [GEOMETRY EXPRESSIONS](#) for a list of all the valid geometric expressions for each of the 12 basic geometry types: [POINT](#), [PNTVEC](#), [LINE](#), [VECTOR](#), [CIRCLE](#), [PLANE](#), [CURVE](#) (includes [SPLINE](#), [SSPLIN](#)) [SURF](#) (includes [NSURF](#)), [MATRIX](#), [PATERN](#), [SOLID](#) and [ANOTE](#).

2.3.8.3 Nested Expressions

A Nested Expression is a statement within a statement. When a **NCL** statement contains a reference to a previously defined identifier, it is possible to insert (“Nest”) the equivalent information in the statement through the use of parentheses which enclose the Nested Expression.

A nested expression may be any legal **NCL** expression, such as a Scalar expression, a Geometric expression, a Matrix expression or a Text expression. A nested expression may be named or unnamed.

The maximum is 99 unnamed nested definitions for any one geometry type in any single statement.

Example:

```
GOLFT/ (LINE/ (POINT/1,2), (PT1=POINT/2,3)), TO, LN3
```

This example contains three (3) nested geometric expressions which are nested two (2) levels deep. Two of the nested expressions are unnamed and the third is named PT1.

The following rules apply to the use of nested expressions:

- Parentheses must be used in pairs and do not replace any other punctuation.
- The result of each nest must be logically correct for the statement in which it is used.
- Processing is from the innermost nest outward and from left to right.
- There is no maximum to the number of levels deep that an expression may be nested within a statement. The limit is determined by the number of characters in the statement. The maximum numbers of characters per statement is 1536.
- Unnamed nesting is not allowed in a **MACRO** definition statement or in a MACRO call statement.
- In **NCL**, a nested **MATRIX** expression in a **REFSYS** statement may be named or unnamed. This differs from APT which must be named.

In general, nesting should not be used indiscriminately. Excessive nesting may extend processing time and, due to the complexity of the statement, may increase programming errors. Nested geometric expressions buried in the motion section of a program will make the program very difficult to debug and to maintain.

2.3.9 Assignment Statements

The assignment statement assigns an identifier to an expression.

Syntax:

```
IDENTIFIER = ASSIGNMENT/EXPRESSION  
(item name) = (major word) / (definition)  
                           (this form does not apply to scalars or text string)
```

There are five categories of assignment statements:

- [Scalar Assignment](#)
- [Geometric Assignment](#)
- [Macro Assignment](#)
- [Matrix Assignment](#)
- [Text String Assignment](#)

Once an identifier has been defined, it may not be redefined. However, if [CANON/ON](#) is in effect, an identifier may be redefined to an expression of the same type.

2.3.9.1 Scalar Assignment Statements

Scalar assignment causes an Arithmetic Value to be associated with an identifier. If the identifier has not been previously defined, the scalar assignment statement causes it to be defined. If the identifier has been previously defined as a scalar, the scalar assignment statement causes the new value to be associated with that identifier and the old value discarded. Scalar identifiers are not automatically generated.

Example:

<u>Valid</u>	<u>Invalid</u>
A=1	PTS=PT3 (if PT3 is defined as a geometric entity such as a point)
B(2)=A*(3/4)	
PTNUM=PTNUM+1	

2.3.9.2 Geometric Assignment Statements

Geometric assignment causes a geometric expression to be associated with an identifier. If the identifier has not been previously defined, the geometric assignment causes it to be defined. If the identifier has been previously defined, the **CANON** setting is ON, and the geometry type is the same, the geometric assignment statement causes the new geometric value to be associated with that identifier and the old value discarded. If the CANON setting is OFF, an error is generated.

Syntax: IDENTIFIER = TYPE/EXPRESSION

- Identifier (item name)

A name is assigned to each geometric entity as it is defined. It establishes a unique “identity” for each entity in the geometric model. This concept allows each entity to be referred to by its name. The unique name of each geometric entity is its identifier.

Examples:

P1 could be point 1
VL2 could be vertical line 2

- Type (major word)

Each geometric entity is stored, retrieved and manipulated by type. Each one is represented in a statement by its major word:

CIRCLE is a circle	PLANE is a plane
CURVE is a curve	POINT is a point
SPLINE non uniform RB-spline	NSURF surface with RB-splines
LINE is a line	SURF is a surface
MATRIX is a matrix	VECTOR is a vector
PATTERN is a pattern	PNTVEC is a point vector
Anote is an annotation	SOLID is a solid

- Expression (definition)

This portion of the definition statement must include all the information necessary to define a “unique” geometric entity. The information must be stated in a specific sequential order: a “fixed format.” The information consists of modifier/data couplets. The modifier/data couplets describe the

size, orientation and location of the geometric entity being defined relative to the pre-defined geometric entities being referenced.

Example:

PT1=POINT/0,0,0

PT1 is the name of the geometric entity being defined

POINT specifies the type of geometric entity being defined

0,0,0 specifies the data which defines the “unique” location of the geometric entity being defined. In this example, the data 0,0,0 specifies that the point being defined will be located at X=0, Y=0, Z=0. As a geometric item is defined, it is checked for valid syntax, assigned a name (if none was specified) and then stored in its canonical form. **NCL** automatically generates an identifier for you if you write a geometric expression instead of a geometric assignment statement. These identifiers are generated by geometric type in the order of definition. **NCL** gives a unique identifier to each expression.

For example, if you write three expressions to define three different lines; **NCL** will identify them as LN1, LN2, and LN3 according to the order in which they were defined.

Examples of Automatic Identifier Generation:

- POINTS

POINT/X1,Y1	PT1
PT/X1,Y1,Z1	PT2
PT/TE	PT3

- LINES

LINE/PT1, PT2	LN1
LN/PT1, PARREL, LN1	LN2
LN/PT1, PERPTO, LN1	LN3

- CIRCLES

CIRCLE/X1,Y1,Z1, radius	CI1
CI/X1,Y1, radius	CI2
CI/PT1, PT2, PT3	CI3

2 NCL LANGUAGE

- **VECTORS**

VECTOR/X1,Y1,Z1	VE1
VE/X1,Y1	VE2
VE/FWD	VE3

- **PLANES**

PLANE/I1,J1,K1,dist	PL1
PL/PT1,PT2,PT3	PL2
PL/PT1,PARLEL,PL1	PL3

- **CURVES**

CURVE/PT1,PT2,...PTn	CV1
SPLINE/FIT,PT1,PT2,...PTn	CV2
SSPLIN/INTOF,SF1,SF2	CV3

- **SURFACES**

SURF/LN1,CI3	SF1
SF/PT3,VE2,CV5,CV6,LN2,CI4	SF2
SF/CV1,THRU,CV2	SF3

- **PATERNS**

PATTERN/LINEAR,PT1,PT2,5	PN1
PN/ARC,CI1,15,125,CLW,6	PN2
PN/PARLEL,PN1,VE1,6	PN3

- **SOLID**

SOLID/BOX,PT1,PT2,0,2	SO1
SOLID/SPHERE,CI2	SO2
SOLID/CYLNDR,CI1,2	SO3

- **ANOTE**

ANOTE/"String 1"	AN1
ANOTE/"String 2",AT,PT1	AN2
ANOTE/"String",AT,PT1,CURVE,CI1	AN3

- **SYMBOL**

SYMBOL/SF1,CV1	SY1
SYMBOL/SF2,SF3,AT,PT1	SY2
SYMBOL/SO1,SF2	SY3

2.3.9.3 Macro Assignment Statements

Macro assignment causes a macro expression to be associated with an identifier. Macro identifiers may not be automatically generated.

Examples:

```
M1=MACRO  
RUFFIT=MACRO/A=3, PS=SF1
```

2.3.9.4 Matrix Assignment Statements

Matrix assignment causes a matrix expression to be associated with an identifier.

Examples:

```
MXPART=MATRIX/X(1),X(2),X(3),PXO, $  
Y(1),Y(2),Y(3),PYO, $  
Z(1),Z(2),Z(3),PZO  
MX1=MX/INVERS, MXPART
```

2.3.9.5 Text String Assignment Statements

Text String assignment causes a string of up to 64 characters to be associated with an identifier. The [string of characters](#) must be enclosed in a pair of double quotes, i.e. the (“) characters. A double quote character may be included in a text string by entering two consecutive double quotes. If the identifier has not been previously defined, the text string assignment statement causes it to be defined. If the identifier has been previously defined as a text string, the text string assignment statement causes the new string of characters to be associated with that identifier and the old string of characters discarded. Text string identifiers are not automatically generated. Text string only allow assignment in one line, no multi-line assignment is allowed.

Example:

Valid:

```
TXT ="This is a valid text string."  
ABC = " " $$ A string with an empty space character  
QUOT="Value of ""L"" is 30."  
BC = "String 1 $"
```

2 NCL LANGUAGE

Invalid:

TX1="Invalid string
No closing double quote

TX2="Value of "L" is 30."

Double quote in string must be represented by two consecutive double quote.

TX3="String continue on \$
next line"
No multi-line string allowed.

2.4 File Specifications

When referencing a file name within a **NCL** part program or from the **NCL** interface it is possible to proceed the file name with a directory path. The filename size cannot be more than 1024 characters or the Operation System limit, whichever is less. This includes the file name itself, the directory/folder path and all the delimiter symbols. Also supported are the use of environmental variables. Only alphabetic, numerical, and the “_”, “-”, “+” and “ ” (blank space) sign are allowed for file specifications. No other symbols are allowed for file specifications. Both the “/” and “\” symbols can be used as a valid directory/folder delimiter. The examples below show file specifications being used with the **NCL INCLUD** command, however the same formats can be used anywhere a file specification is required.

Examples:

includ/C:\user\luke\macros\tlchg.mac
includ/D:/disk2/user/solo/startup.mac

Assuming the environmental variable “mac” with a value of “C:\user\system\ncl\macro” was set with the

Start - Control Panel - Performance and Maintenance - System - Advanced - Environmental Variables

sequences.

A **NCL** file specification could be made as follows:

includ/mac\tlchg.mac

or

includ/mac/tlchg.mac

3 GEOMETRY EXPRESSIONS

3.0 Introduction

Geometry items in **NCL**.

Canonical Form:

CIRCLE/ X, Y, Z, I, J, K, R, I_{Limit_Plane}, J_{LP}, K_{LP}, D_{LP}

CURVE/ Due to the complexity of the **NCL** CURVE, the canonical form cannot be included in this manual.

SPLINE/ Due to the complexity of the **NCL** SPLINE, the canonical form cannot be included in this manual.

LINE/ X, Y, Z, DELTA X, DELTA Y, DELTA Z

PATTERN/ is a series of X, Y, Z coordinates for a point-pattern or a series of X, Y, Z coordinates and I, J, K components for a point-vector-pattern.

PLANE/ I, J, K, D

POINT/ X, Y, Z

PODDEF/ Due to the complexity of the **NCL** PODDEF, the canonical form is not included in this manual.

SURF/ Due to the complexity of the **NCL** SURFACE, the canonical form cannot be included in this manual.

NSURF/ Due to the complexity of the **NCL** NSURF, the canonical form is not included in this manual.

VECTOR/ I, J, K

PNTVEC/ X, Y, Z, I, J, K

MATRIX/ I_x, J_x, K_x, D_x, I_y, J_y, K_y, D_y, I_z, J_z, K_z, D_z

SOLID/ Refers to the corresponding sections for each type of solids.

NOTE:

- a) MATRIX is included in this section because it is used to Modify Geometric Entities and Cutter Motion.
- b) PODDEF is used to define vacuum pod fixture system for American GFM Ultrasonic Cutting Machine. It is not a regular type of geometry. A label is not allowed for the definition and it cannot be displayed.

As a geometric item is defined, it is checked for valid syntax, assigned a name, if none was specified, then stored in its canonical form. It may be redefined, assuming that **CANON** is ON, however, the type of the item may not be changed. For example, the statements:

```
PT1=POINT/1,2,3  
PT1=POINT/2,2,0
```

are valid with CANON/ON, however, the second statement in the set

```
PT1=POINT/1,2,3  
PT1=CIRCLE/X,Y,R
```

is not valid because it attempts to assign the label PT1 to be a different geometric type (**CIRCLE**) than it was previously (**POINT**).

Geometry in the **NCL** system is bounded by its definition. Therefore, a line defined by two points only exists between those two points. **NCL** generates an extension to all geometry for tool motion.

This means that even though a line is two inches long, a tool can be driven on the line extension infinitely in either direction. However, if a SURF is defined using that line as a boundary, the surface on that side will be displayed as two inches long even though a tool can be driven on the surface extension.

The remainder of the Reference Manual contains a description of each **NCL** Language Statement. Each statement is described using the following general format.

Command:	Definition
Syntax:	Acceptable Statement Format with parameters and options.

Icon Menu Sequence Shows the required interactive menu picks to activate the corresponding command.

Examples: This section will either describe an example or show a sample drawing.

Calculation Method: Explains how **NCL** calculates the canonical data of the geometry.

Error Conditions: Explains conditions which will produce an error message.

Note: The color **ORANGE** will be used instead of the default color **YELLOW** in all the examples for better contrast with a white background.

CIRCLES

The following list gives an abbreviated notation of all the valid CIRCLE definition formats. See the individual sections for a complete explanation of each format.

1. `CI/x,y[,z],radius`
2. `CI/PT1,PT2,PT3`
3. `CI/PV1,PV2`
or `CI/PV1 , PT2
 PT1,VE1`
or `CI/PT1,PV2
 PT1,VE2`
4. `CI/TANTO,LN1,direction,PT1,RADIUS,radius`
5. `CI/direction,LN1,direction,LN2,RADIUS,radius`
6. `CI/direction,IN ,CI1,IN ,CI2,RADIUS,radius
 OUT OUT`
7. `CI/direction,LN1,direction,IN ,CI1,RADIUS,radius
 OUT`
8. `CI/direction,LN1,direction,LN2,direction,LN3`
9. `CI/direction,LN1,direction,CV1,PT1,RADIUS,radius
 PV1`
10. `CI/IN ,CI1,direction,CV1,PT1,RADIUS,radius
 OUT PV1`
11. `CI/PT1,direction,IN ,CI1,RADIUS,radius
 PV1 OUT`
12. `CI/CENTER,PT1,TANTO,LN1
 PV1`
13. `CI/CENTER,PT1,RADIUS,radius
 PV1`
14. `CI/CENTER,PT1,PT2
 PV1 PV2`

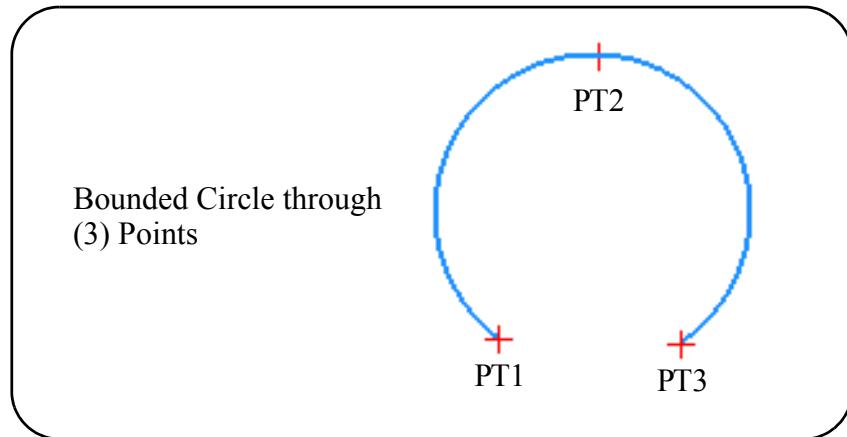
15. **CI**/CENTER, PT1, LARGE, TANTO, CI1
PV1 SMALL
16. **CI**/direction, PT1, PT2, RADIUS, radius
PV1 PV1
17. **CI**/CI1[, Dx[, Dy[, Dz]]]
18. **CI**/OFFSET, OUT, CI1, offset-value
IN
19. **CI**/CANON, x, y, z, i, j, k, radius[, i_{Limit-Plane}, j_{Lp}, k_{Lp}, d_{Lp}]
or **CI**/CANON, PV1 , radius[, PL1]
 PV2, VE2
20. **FILLET**/radius, LN1, LN2[[...], LNn]]
CI1 CI2 CIn

Example Circle Expressions:

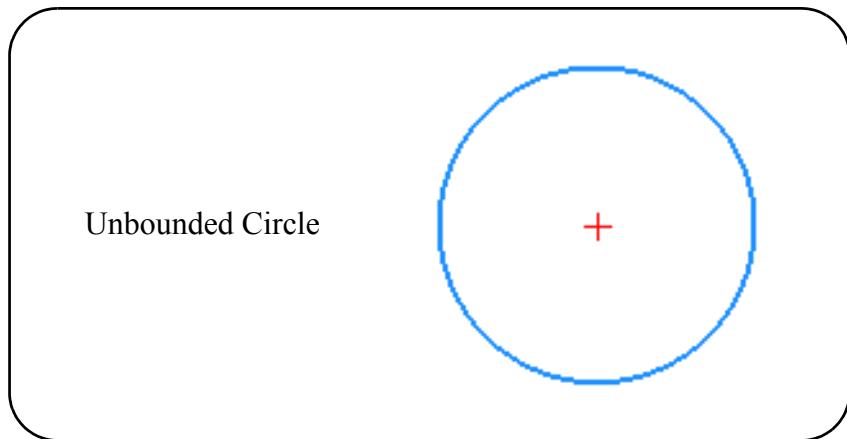
```
CI/2,3,1,175
CIRCLE/PT1,PT2,PT3
CI/PT1,VE1,PT2
CI/PV1,PT2
CI/PT1,PT2,VE2
CIRCLE/X+2/3,Y+2/3,2.5
CI/XSMALL,LN1,YLARGE,LN2,RADIUS,50
CIRCLE/XS,LN1,XL,LN2,YL,LN3
CI/XS,IN,CI1,OUT,CI2,RADIUS,1.2
CI/PT24,XLARGE,OUT,CI3,RADIUS,1.125
CI/XL,LN1,YL,IN,CI1,RADIUS,5
CI/CENTER,PTCP,TANTO,LN8
CI/CE,PV1,TT,LN2
CI/CENTER,PT2,RADIUS,1.25
CI/CE,PV1,RA,2
CI/CENTER,PT3,P(44)
CI/CE,PV1,PV2
CI/CENTER,PT9,LARGE,TANTO,CI2
CI/CE,PV1,SMALL,TT,CI2
CIRCLE/TANTO,LN5,YSMALL,PT2,RADIUS,2.25
CI/XL,PT1,PT2,RADIUS,1.250
CI/XS,PV1,PV2,RA,1
CI/XS,LN1,YL,CV2,PV3,RA,1
CI/CANON,X,Y,Z,0,0,1,.75
CI/CANON,1,1,0,0,0,1,3,1,0,0,0
CI/CANON,PV1,2
CI/CANON,PT1,(VE/0,1,0),1.75,(PL/0,0,1,0)
CI/CI1
CI/CI1,0,2
```

3.1 CIRCLE Expressions

The dictionary defines a circle as "a closed planar curve every point of which is equidistant from a fixed point within the curve." However, geometry in the **NCL** system is bounded by its definition. Therefore, a circle defined by three points only exists between those three points. **NCL** generates an extension to all bounded geometry, as necessary, to satisfy requirements for geometric definitions and tool motion. The extension of a circle, is the remainder of the bounded circular arc to form a closed curve (a 360 degree circle), for example: a circle defined by three points begins at the first point, passes through the second point and ends at the third point. The extension of the circle continues through the third point, in the same direction and terminates at the first point (at 360 degrees). The extension is also referred to as the complement of the circular arc.



When defined by a method which does not specify the boundaries for the circle, for example: a circle defined by its center point and its radius, the circle will be an unbounded closed curve of 360 degrees.



Canonical Form:

CIRCLE/X, Y, Z, I, J, K, R, ILimit_Plane, J_{LP}, K_{LP}, D_{LP}

Where:

X, Y, Z = the center coordinate of the circle.

I, J, K = the vector normal to the circular plane with counter clockwise rotation direction

R = the radius

I_{Limit_Plane}, J_{LP}, K_{LP}, D_{LP} = canonical form of the limiting plane

Special Conditions:

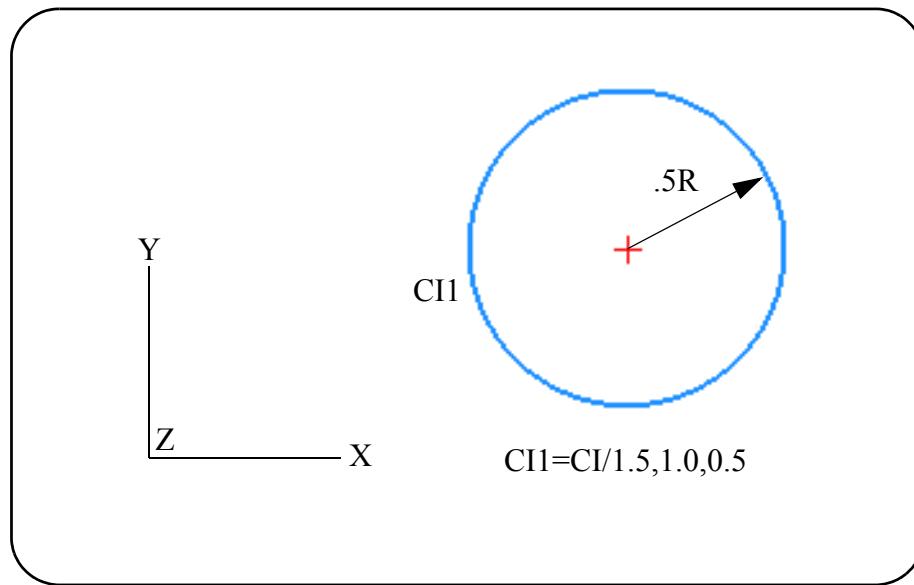
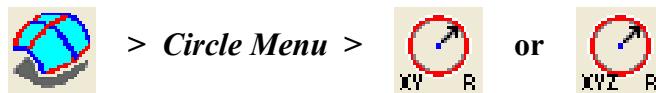
- Dark **BLUE** is the **NCL** System DEFAULT color for CAM CIRCLES.

3.1.1 A Circle By Center Coordinates And Radius

Command Syntax

```
CIRCLE/X-coordinate, Y-coordinate, [Z-coordinate,] $  
radius
```

Icon Menu Sequence:



NOTE: If only three scalar expressions/identifiers are specified, **NCL** will use the System DEFAULT Z-value which is Z=0 (the XY-plane of the current REFERENCE SYStem).

Calculation Method:

- The plane of the circle will be parallel to the XY-plane of the "current" REFERENCE SYStem, at the Z-coordinate of its center point.
- The arc length of the circle will be a full unbounded 360 degrees.

Error Condition:

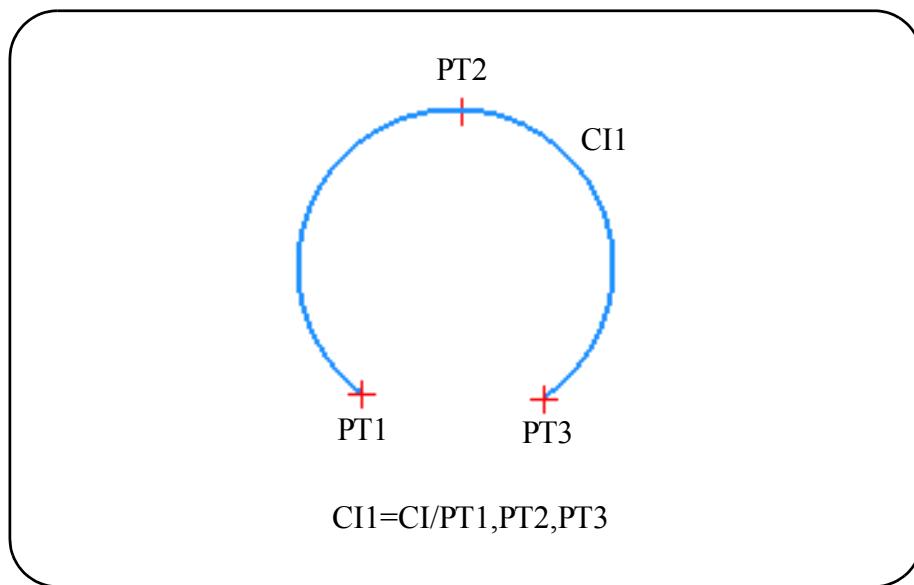
- The radius must be greater than .001.

3.1.2 A Circle By Three Points On Its Circumference

Command Syntax

CIRCLE/point,point,point

Icon Menu Sequence:



Calculation Method:

- The plane of the circle contains the three referenced points.
- The length of the circle is calculated from the first referenced point to the third referenced point.
- The direction of the circle is from the first referenced point to the second referenced point.

Error Conditions:

- The three referenced points must not be colinear or coincident.
- The calculated radius must be greater than .001.

3.1.3 A Circle By A Combination Of Points, Vectors And Point-Vectors

Command Syntax:

CIRCLE/point-vector,point-vector

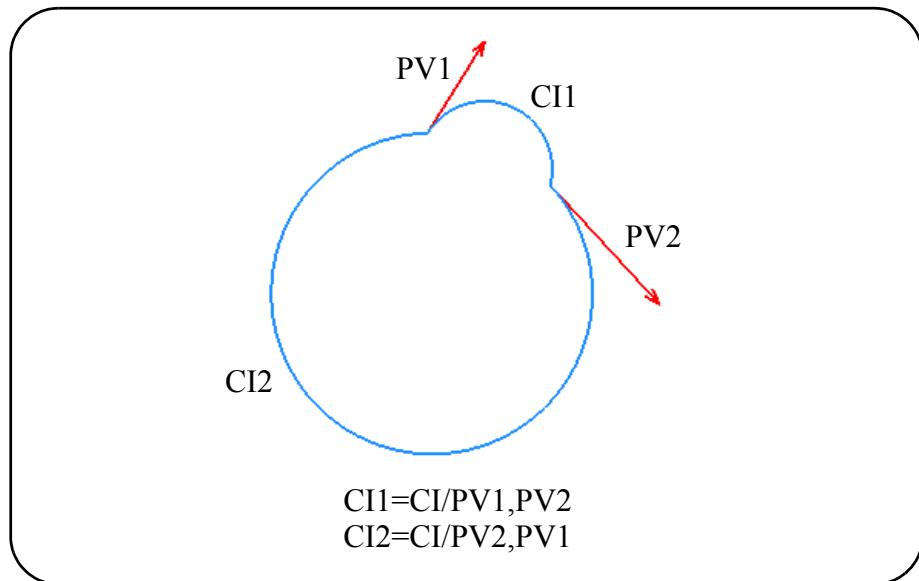
or

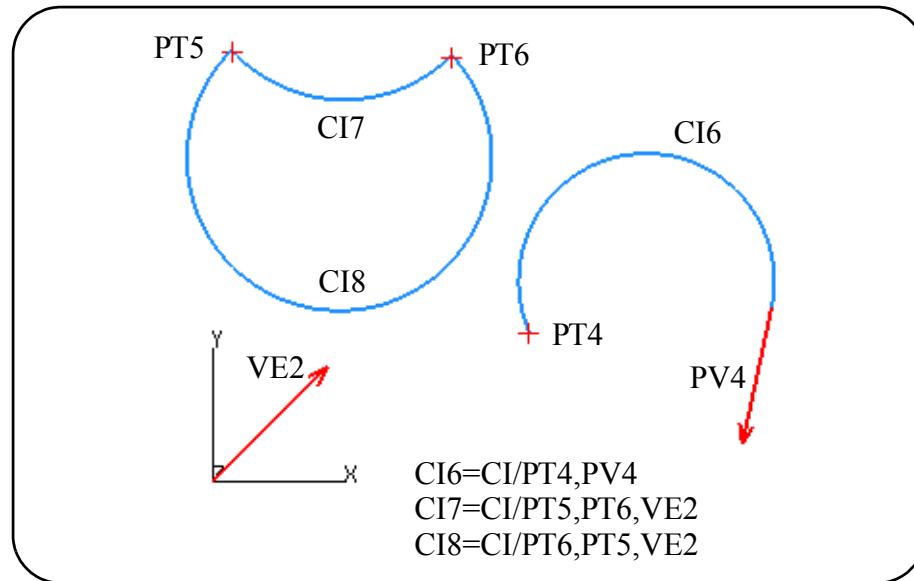
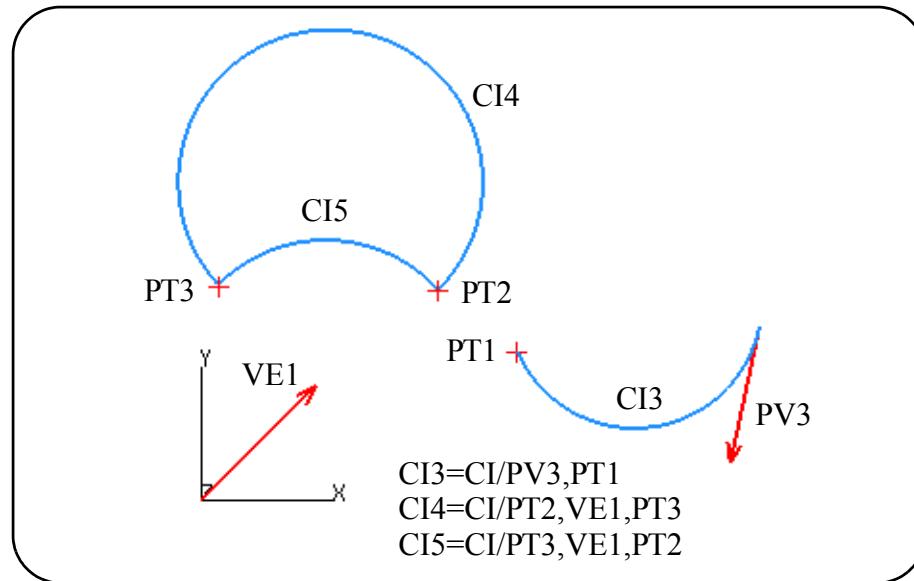
CIRCLE/point,point-vector
point,vector

or

CIRCLE/point-vector,point
point,vector

Icon Menu Sequence:





Calculation Method:

- The plane of the circle contains the two referenced points (or the referenced point-vectors origin) and the “implied point” which is the non-associated tip of the referenced vector/point-vector.

- The arc length of the circle is calculated from the first referenced point (or the point-vector origin) to the second referenced point (or the point-vector origin).
- The direction of the circle is from the first referenced point (or the point-vector origin) to the second referenced point (or the point-vector origin).
- If at least one point is specified, the circle is tangent to the referenced vector (or the point-vector) at the associated point. The vector always points in the forward sense direction of the circle no matter which point it is associated with.
- If two point-vectors are specified, the circle is tangent to the first referenced point-vector and points in the forward sense direction of the circle.

Error Conditions:

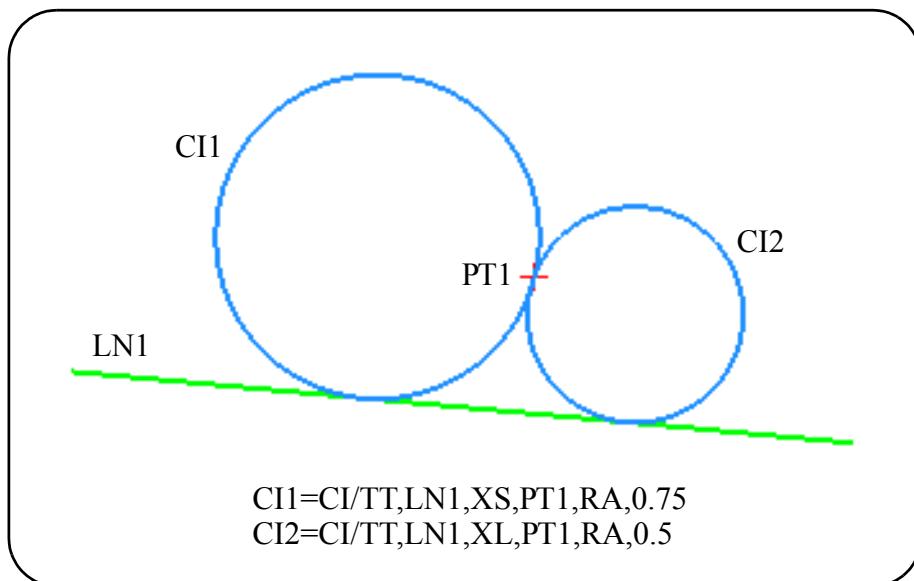
- The three points (the two referenced points and the implied point) must not be colinear or coincident.
- The calculated radius must be greater than .001.

3.1.4 A Circle Tangent To A Line, A Point On Its Circumference And A Radius

Command Syntax:

```
XLARGE
XSMALL
CIRCLE/TANTO, line, YLARGE, point, RADIUS, radius
YSMALL
```

Icon Menu Sequence:



NOTE: The modifiers **XLARGE**, **XSMALL**, **YLARGE** and **YSMALL** designate which of the two possible circles is to be defined. The modifiers describe the location of the center of the desired circle relative to the center of the other possible circle in relation to the “current” **REFerence SYStem**.

Calculation Method:

- The point and the line are projected onto the XY-plane of the “current” REference SYstem.
- The tangency point of the projected-line/circle, using the line’s extension if necessary, will be calculated.
- The plane of the circle will be the XY-plane of the ‘current’ REference SYstem.
- The arc length of the circle will be a full, unbounded 360 degrees.

Error Conditions:

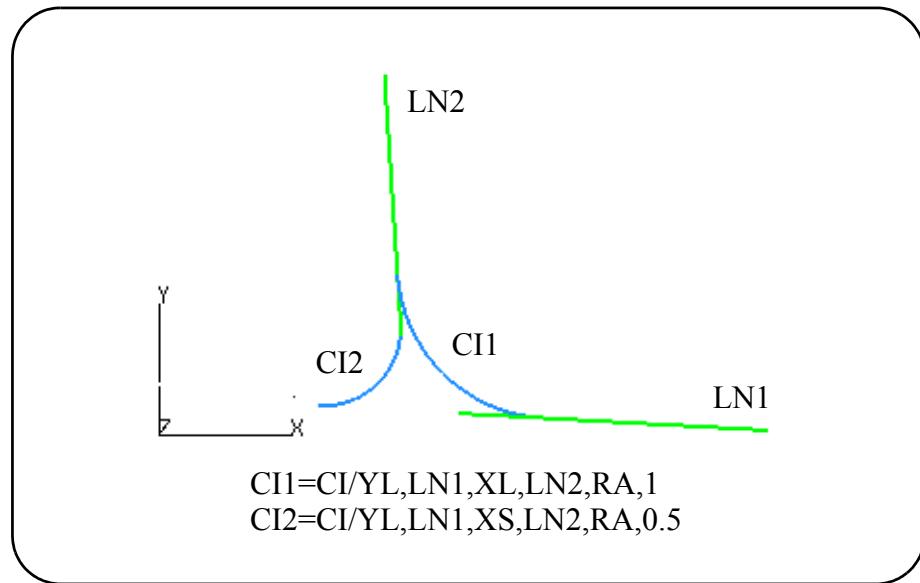
- The radius must be greater than .001.
- The line must not be a zero length line when projected.

3.1.5 A Circle Tangent To Two Intersecting Lines And A Radius

Command Syntax:

```
XLARGE      XLARGE
XSMALL      XSMALL
CIRCLE/YLARGE, line, YLARGE, line, RADIUS, radius
           YSMALL      YSMALL
```

Icon Menu Sequence:



NOTE: The modifiers **XLARGE**, **XSMALL**, **YLARGE** and **YSMALL** designate the position of the center of the desired circle relative to the predefined line specified in each "modifier, line" pair.

Calculation Method:

- The lines are projected onto the XY-plane of the "current" **REFERenCe SYStem**.

- The tangency points of the projected-lines/circle, using the line's extensions if necessary, will be calculated.
- The plane of the circle will be the XY-plane of the "current" REference SYstem.
- The arc length of the circle will be calculated from the first line/circle tangency point to the second line/circle tangency point.
- The direction of the circle will be from the first tangency point to the second tangency point.

Error Conditions:

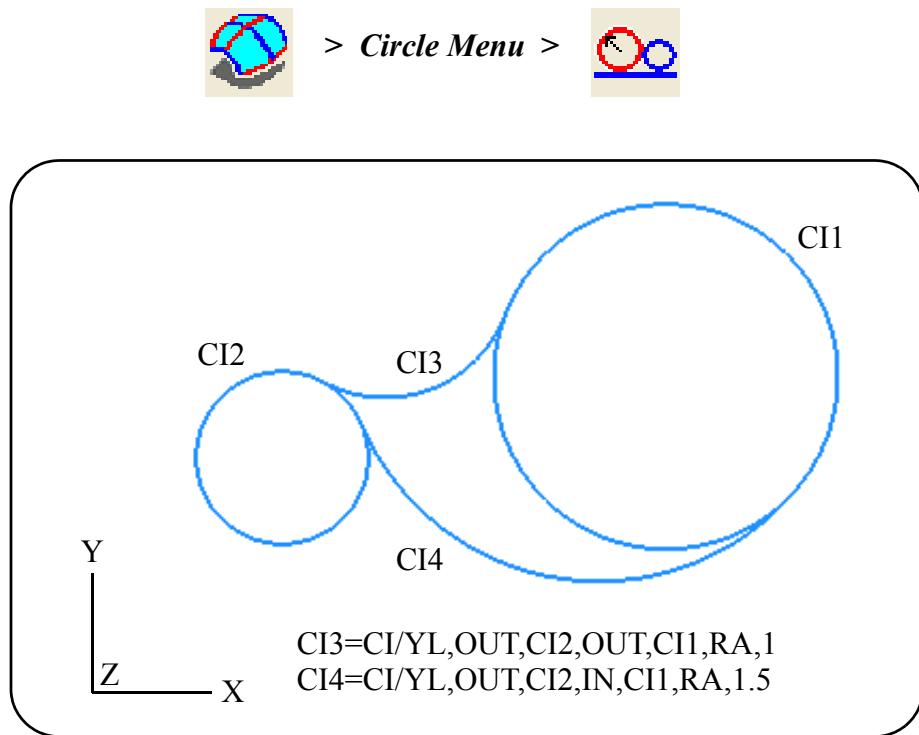
- The line must not be a zero length line when projected.
- The lines or their extensions must intersect when projected.

3.1.6 A Circle Tangent To Two Circles And A Radius

Command Syntax:

```
XLARGE
XSMALL IN           IN
CIRCLE/YLARGE,OUT,circle,OUT,circle,circle,RADIUS,Radius
YSMALL
```

Icon Menu Sequence:



NOTE: The modifiers **XLARGE**, **XSMALL**, **YLARGE** and **YSMALL** designate which of the two possible circles is to be defined. The modifier describes the location of the center of the desired circle relative to the other possible choice. The modifiers **IN** and **OUT** specify if the defining circle they are associated with lies inside or outside the circle being defined.

Calculation Method:

- The circles are projected onto the XY-plane of the "current" REference SYstem.
- The tangency point of the projected-circle/projected-circle, using the circle extensions if necessary, will be calculated.
- The plane of the circle will be the XY-plane of the "current" REference SYstem.
- The arc length of the circle will be from the first tangency point to the second tangency point.
- The direction of the circle will be from the first tangency point to the second tangency point.

Error Conditions:

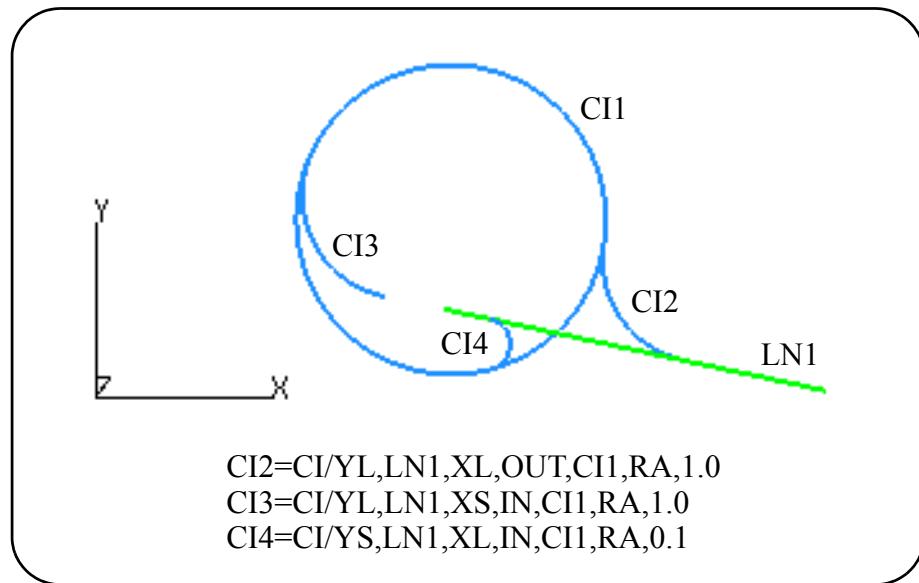
- The radius must be greater than .001.
- The referenced circles may not be "tipped" relative to the XY-plane of the "current" REference SYstem.

3.1.7 A Circle Tangent To A Circle And A Line With A Specified Radius

Command Syntax:

```
XLARGE      XLARGE
XSMALL      XSMALL IN
CIRCLE/YLARGE, line, YLARGE, OUT, circle, RADIUS, radius
YSMALL      YSMALL
```

Icon Menu Sequence:



NOTE: The modifiers **XLARGE**, **XSMALL**, **YLARGE** and **YSMALL** designate which of the two possible circles is to be defined. The modifier describes the location of the center of the desired circle relative to the other possible choice. The modifiers **IN** and **OUT** specify if the defining circle they are associated with lies inside or outside the circle being defined.

Calculation Method:

- The circles are projected onto the XY-plane of the "current" REFERENCE SYStem.
- The tangency point of the projected-line/projected-circle, using the line and circle extensions if necessary, will be calculated.
- The plane of the circle will be the XY-plane of the "current" REFERENCE SYStem.
- The arc length of the circle will be the "shortest arc" as calculated from the line/circle tangency point to the circle/circle tangency point.
- The direction of the circle will be counterclockwise.

Error Conditions:

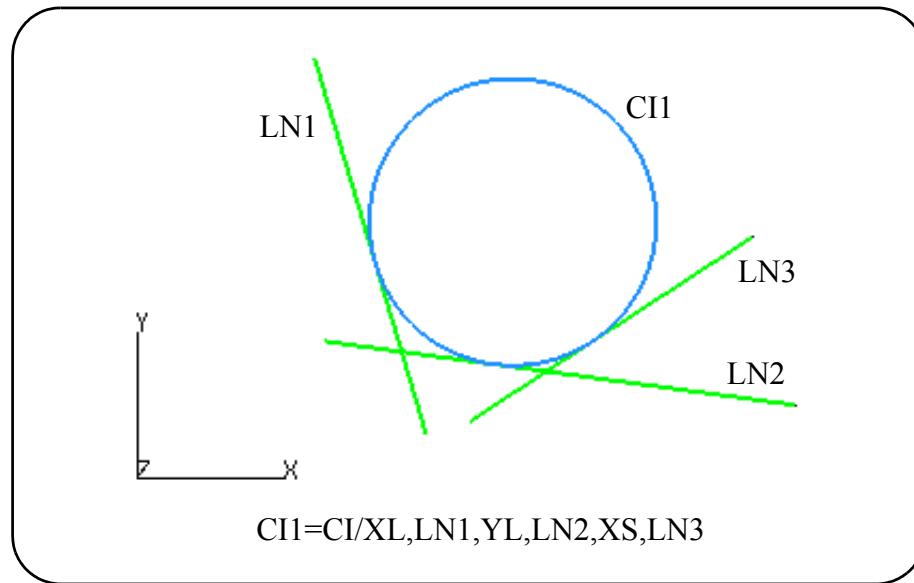
- The radius must be greater than .001.
- The line may not be zero length when projected.
- The referenced circle may not be "tipped" relative to the XY-plane of the "current" REFERENCE SYStem.

3.1.8 A Circle Tangent To Three Lines

Command Syntax:

```
XLARGE      XLARGE      XLARGE
XSMALL      XSMALL      XSMALL
CIRCLE/YLARGE, line, YLARGE, line, YLARGE, line
          YSMALL      YSMALL      YSMALL
```

Icon Menu Sequence:



NOTE: The lines may be at any orientation, and do not need to be perpendicular to each other.

Calculation Method:

- The lines are projected onto the XY-plane of the "current" REFERENCE SYStem.

- The tangency points of all the projected-line/projected-circle, using the line extensions if necessary, will be calculated.
- The plane of the circle will be the XY-plane of the "current" REference SYStem.
- The arc length of the circle will be a full, unbounded 360 degrees.
- The lines do not need to physically intersect each other as long as the projections (extensions if necessary) intersect.

Error Conditions:

- The lines may not be zero length when projected.
- The calculated radius must be greater than .001.

3.1.9 A Circle Tangent To A Line And A Curve/B-Spline With A Specified Radius And A Near Point/Point-Vector

Command Syntax:

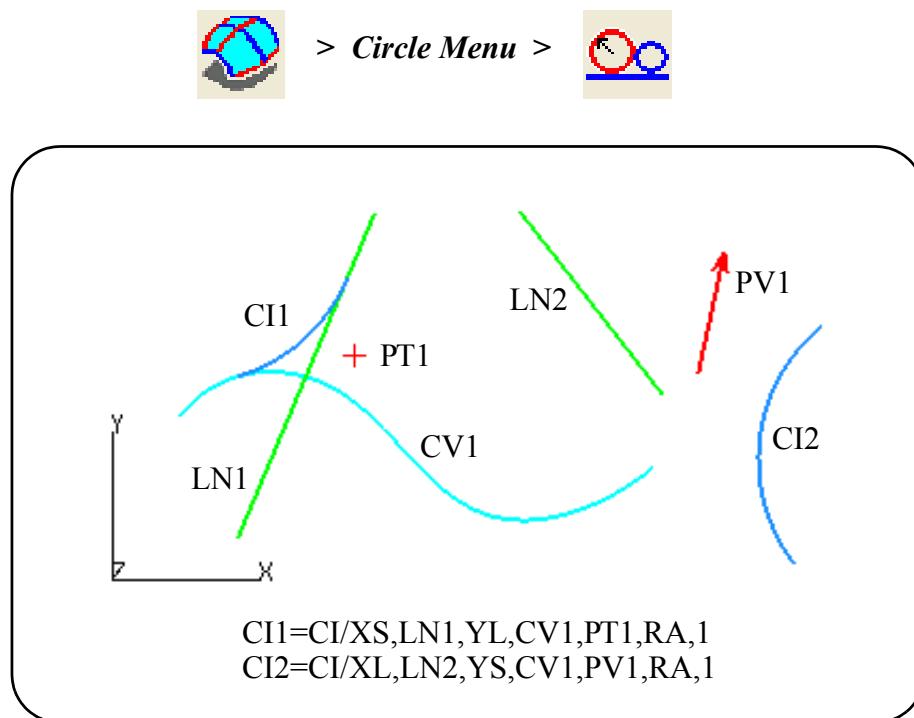
```

XLARGE      XLARGE
XSMALL      XSMALL
CIRCLE/YLARGE, line, YLARGE, curve, near-point ,      $
              YSMALL      YSMALL           pntvec

RADIUS, radius

```

Icon Menu Sequence:



NOTE: The near-point or the point-vector origin is used to determine which of several possible intersections is the desired location.

Calculation Method:

- The curve and line are projected onto the XY-plane of the "current" REference SYstem followed by;
- The tangency points of the circle/projected-line and circle/projected-curve, using the line and curve extensions if necessary, will be calculated.
- The plane of the circle will be the XY-plane of the "current" REference SYstem.
- The arc length of the circle will be from the first tangency to the second tangency.

Error Conditions:

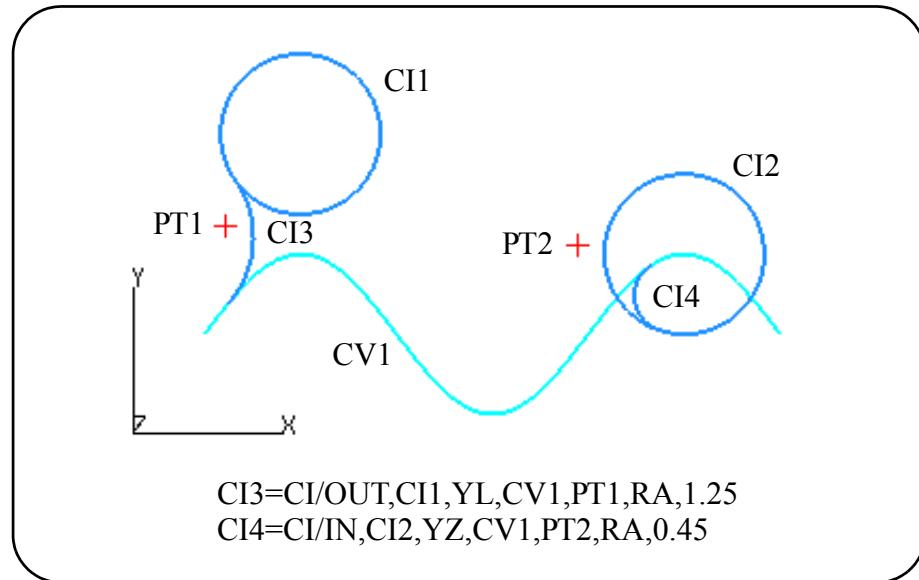
- The line/curve must not be a zero length line/curve when projected.
- The line/curve or their extensions must intersect when projected.

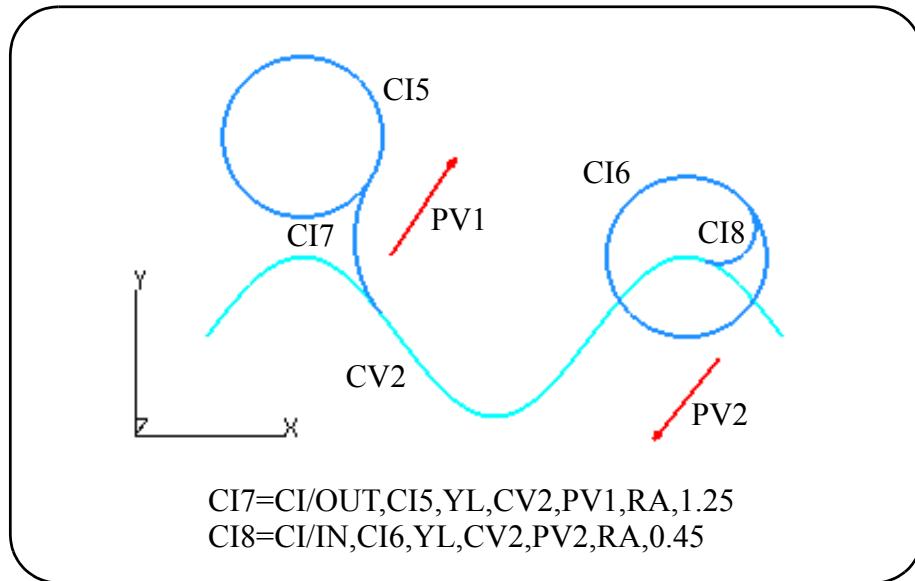
3.1.10 A Circle Tangent To A Circle And A Curve/Spline, With A Specified Radius And A Near Point/Point-Vector

Command Syntax:

```
CIRCLE/IN ,circle,XLARGE,curve,point      ,      $  
        OUT          XSMALL      point-vector  
                  YLARGE  
                  YSMALL  
  
RADIUS, radius
```

Icon Menu Sequence:





Calculation Method:

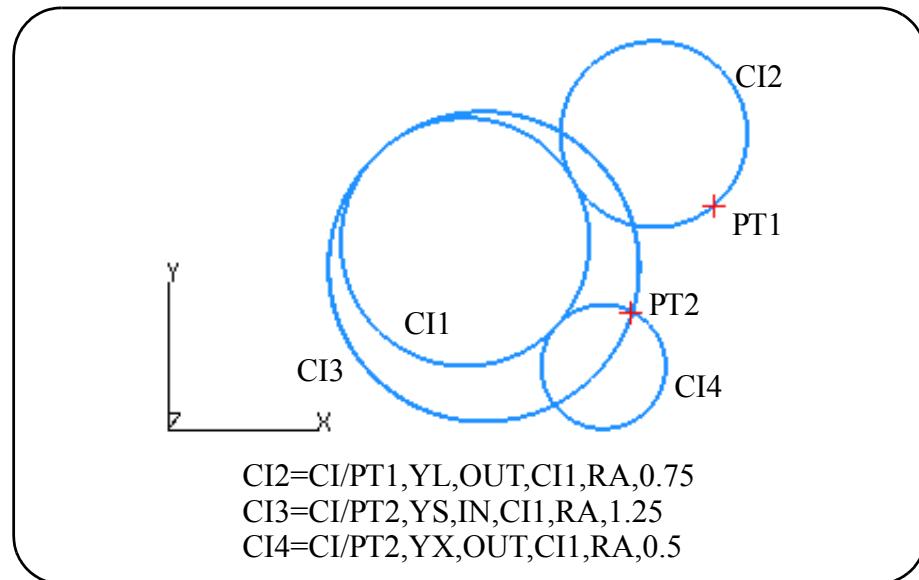
- The curve and existing circle are projected onto the XY-plane of the current REFERENCE SYStem.
- The tangency points of the projected-curve/projected-circle, using the curve and circle extension if necessary, will be calculated.
- The plane of the circle will be the XY-plane of the current REFERENCE SYStem.
- The arc length of the circle will be governed by the two tangent points.

3.1.11 A Circle By A Point On Its Circumference, A Tangent Circle And A Specified Radius

Command Syntax:

```
XLARGE
XSMALL IN
CIRCLE/point, YLARGE, OUT, circle, RADIUS, radius
YSMALL
```

Icon Menu Sequence:



NOTE: The modifiers **XLARGE**, **XSMALL**, **YLARGE** and **YSMALL** designate which of the two possible circles is to be defined. The modifier describes the location of the center of the desired circle relative to the other possible choice. The modifiers **IN** and **OUT** specify if the defining circle they are associated with lies inside or outside the circle being defined.

Calculation Method:

- The circles are projected onto the XY-plane of the "current" REference SYStem.
- The tangency point of the projected-circle/circle, using the circle extensions if necessary, will be calculated.
- The plane of the circle will be the XY-plane of the "current" REference SYStem.
- The arc length of the circle will be a full, unbounded 360 degrees.

Error Conditions:

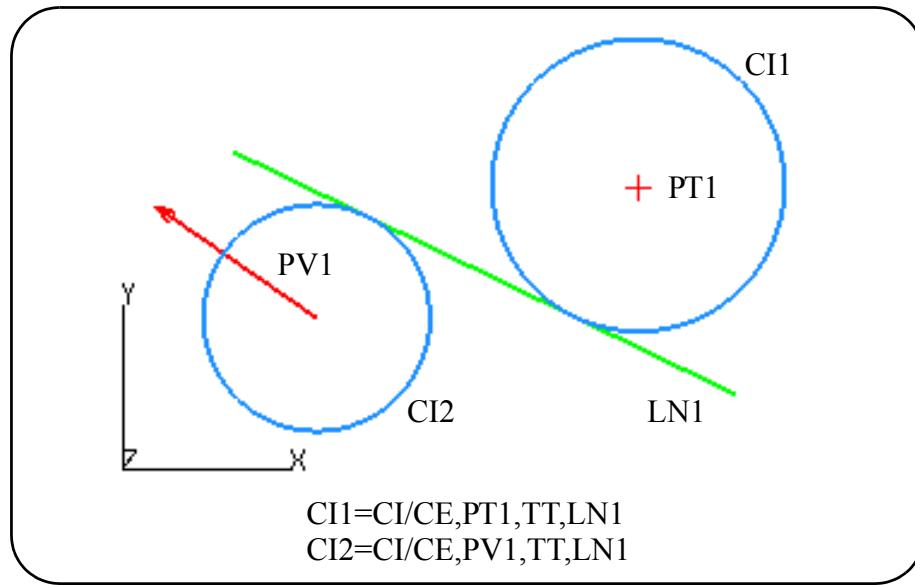
- The radius must be greater than .001.
- The referenced circles may not be "tipped" relative to the XY-plane of the "current" REference SYStem.
- The radius must be large enough to create a circle whose circumference passes through the point and is tangent to the circle.

3.1.12 A Circle By A Center At A Point/Point-Vector And Tangent To A Line

Command Syntax:

```
CIRCLE/CENTER,point      ,TANTO,line
                      point-vector
```

Icon Menu Sequence:



Calculation Method:

- The line is projected onto a Z-plane (through the point) which is parallel to the XY-plane of the "current" REference SYstem.
- The tangency point of the project line/circle, using the line extension if necessary, will be calculated.
- The plane of the circle will be parallel to the XY-plane of the "current" REference SYstem at the Z value of the point or the point-vector origin.
- The arc length of the circle will be a full, unbounded 360 degrees.

Error Conditions:

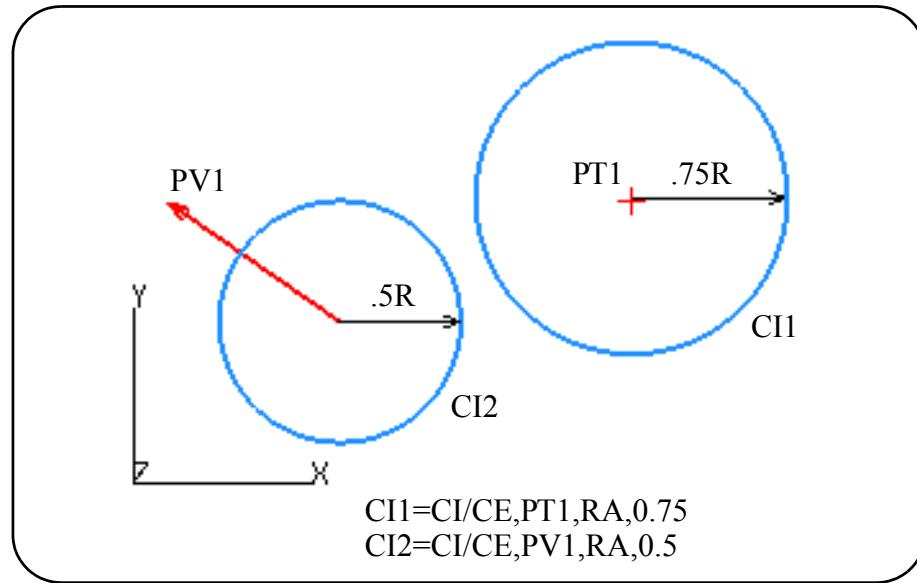
- The line must not be zero length when projected.
- The radius must be greater than zero (circle center point must not lie on the line).

3.1.13 A Circle With Center At A Point/Point-Vector With A Specified Radius

Command Syntax:

```
CIRCLE/CENTER,point      ,RADIUS,radius
          point-vector
```

Icon Menu Sequence:



Calculation Method:

- The plane of the circle will be parallel to the XY-plane of the "current" REFERENCE SYSTEM at the Z-coordinate of the Center Point.
- The arc length of the circle will be a full, unbounded 360 degrees.

Error Condition:

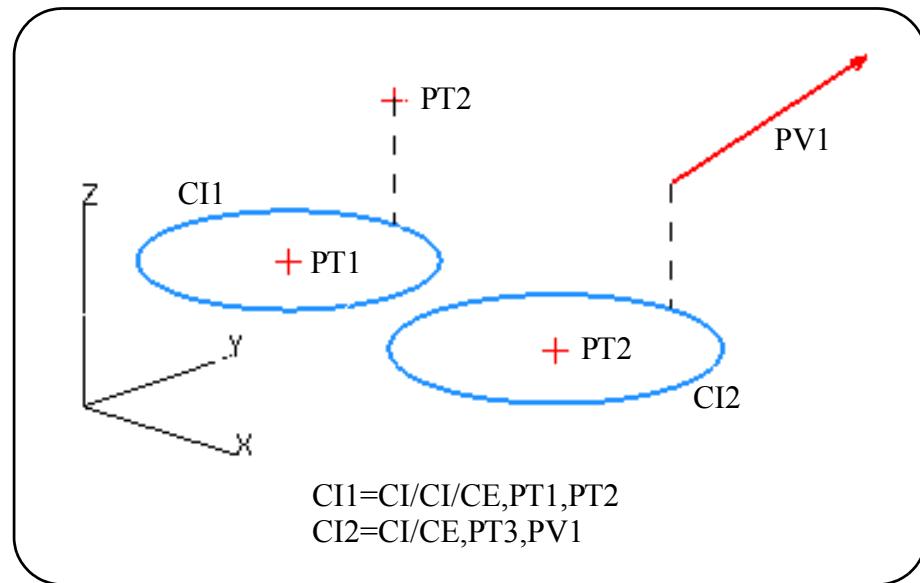
- The radius must be greater than .001.

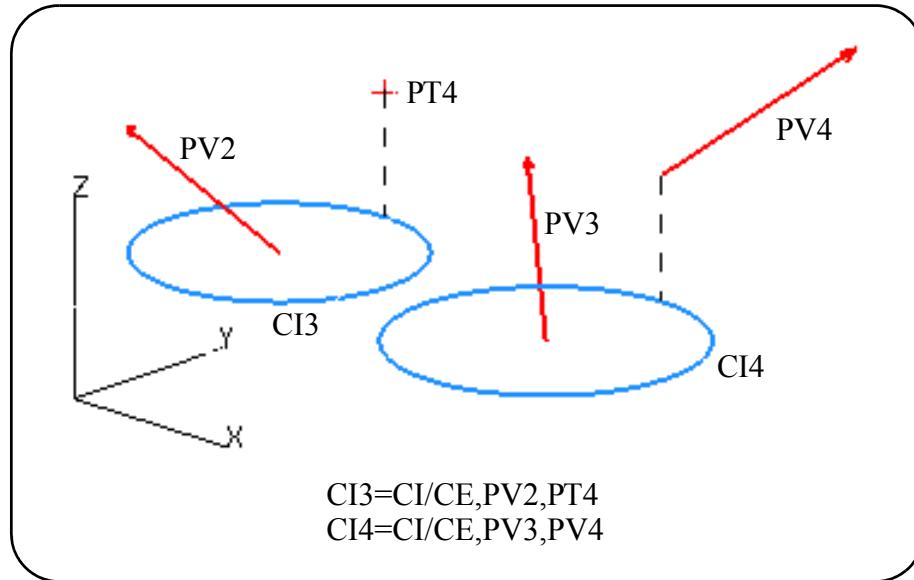
3.1.14 A Circle With Center At A Point/Point-Vector And A Point/Point-Vector On Its Circumference

Command Syntax:

```
CIRCLE/CENTER,point      ,point  
          point-vector point-vector
```

Icon Menu Sequence:





Calculation Method:

- The point or the point-vector origin on the circumference is projected onto a Z-plane containing the center point which is either the specified point or the point-vector origin.
- The plane of the circle will be parallel to the XY-plane of the "current" REFERENCE SYStem at the Z value of the center point.
- The arc length of the circle will be a full, unbounded 360 degrees.

Error Condition:

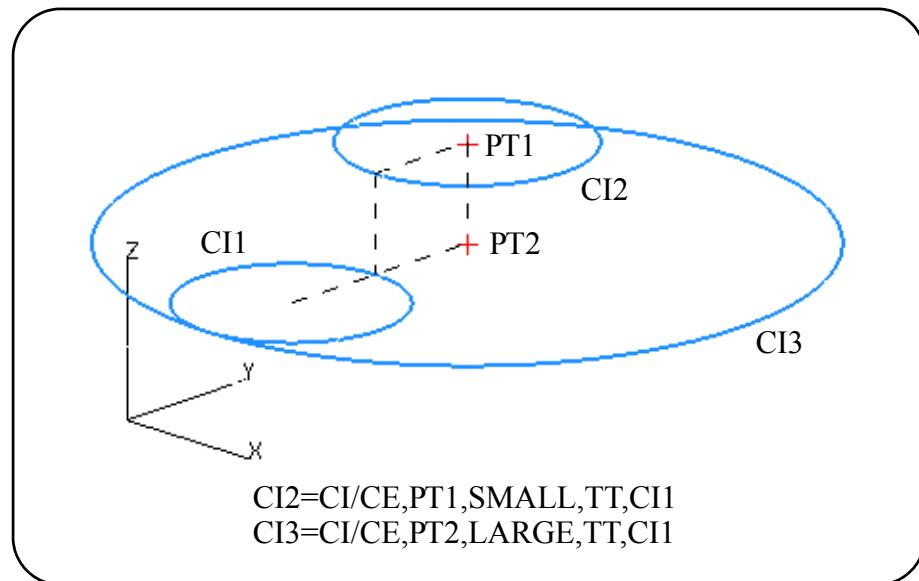
- The points or the point-vectors origin must not be coincident when projected.

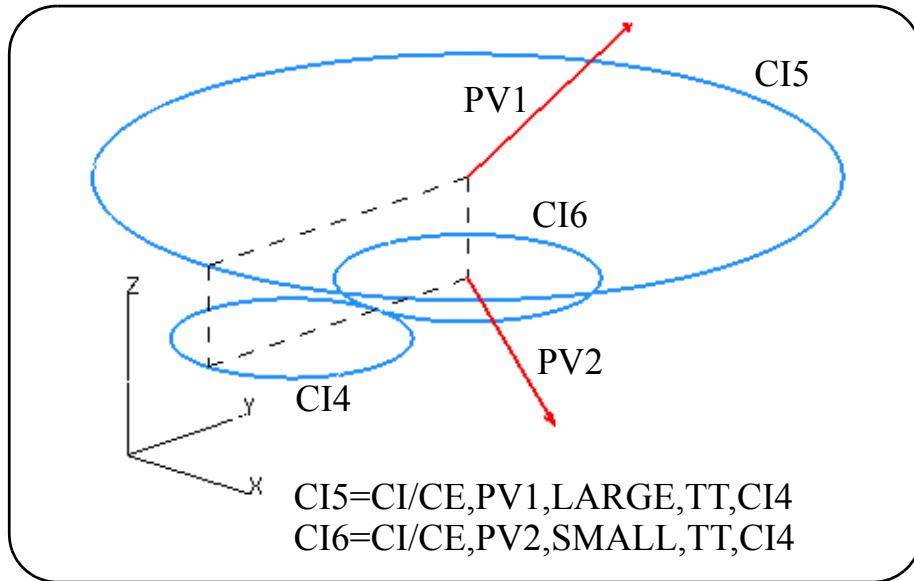
3.1.15 A Circle With Center At A Point/Point-Vector And Tangent To A Circle

Command Syntax:

```
CIRCLE/CENTER,point      ,LARGE,TANTO,circle  
point-vector SMALL
```

Icon Menu Sequence:





NOTE: The modifiers **LARGE** and **SMALL** indicate which of the two possible circles that could be defined is the one desired.

Calculation Method:

- The referenced circle will be projected onto a plane parallel to the XY-plane through the referenced point or the point-vector origin.
- The calculation of the projected-circle/circle tangency point using the circle extensions if necessary.
- The plane of the circle will be parallel to the XY-plane of the "current" REference SYstem at the Z-value of the point.
- The arc length of the circle will be a full, unbounded 360 degrees.

Error Conditions:

- The referenced point (or the referenced point-vector origin) and the center of the referenced circle must not be coincident.
- The referenced circle may not be "tipped" relative to the XY-plane of the "current" REference SYstem.

3.1.16 A Circle By Two Points/Point-Vectors Or A Point And A Point-Vector On Its Circumference With A Specified Radius

Command Syntax:

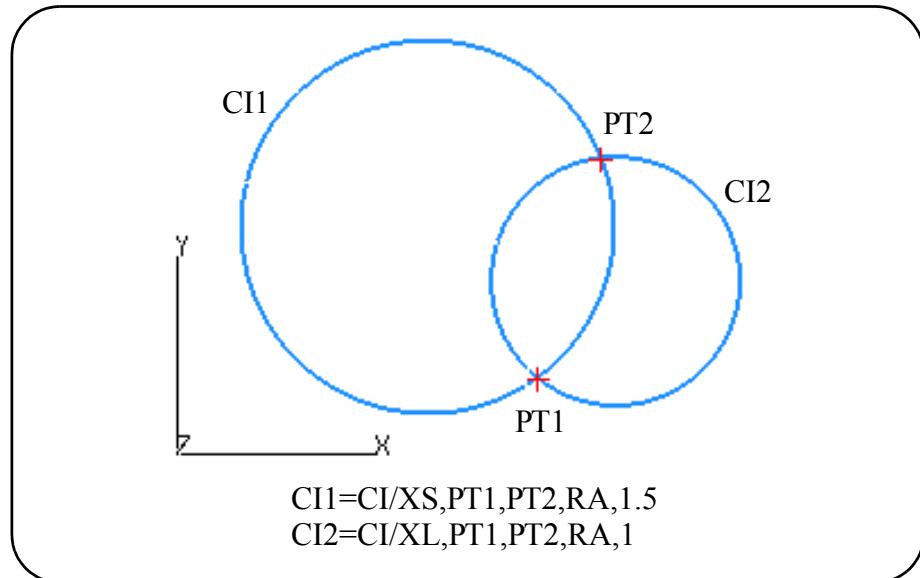
```

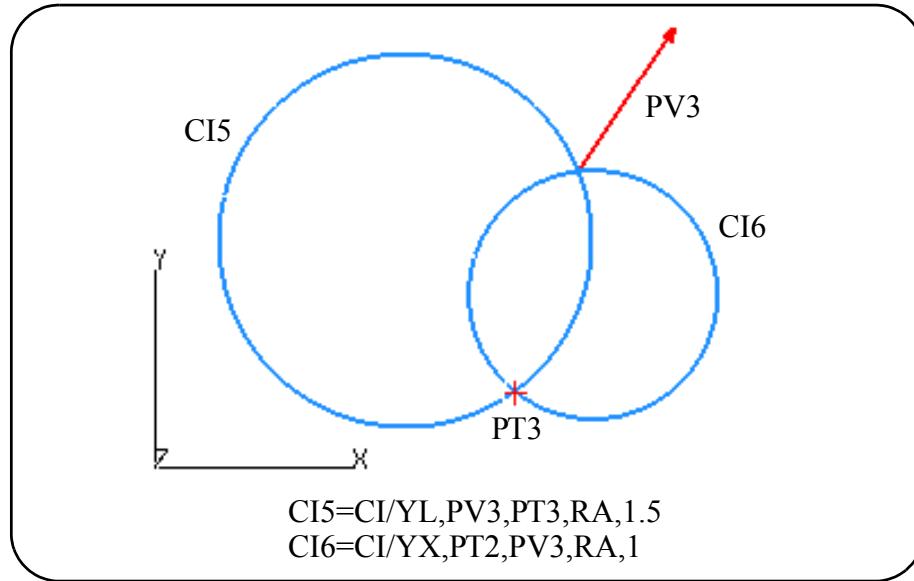
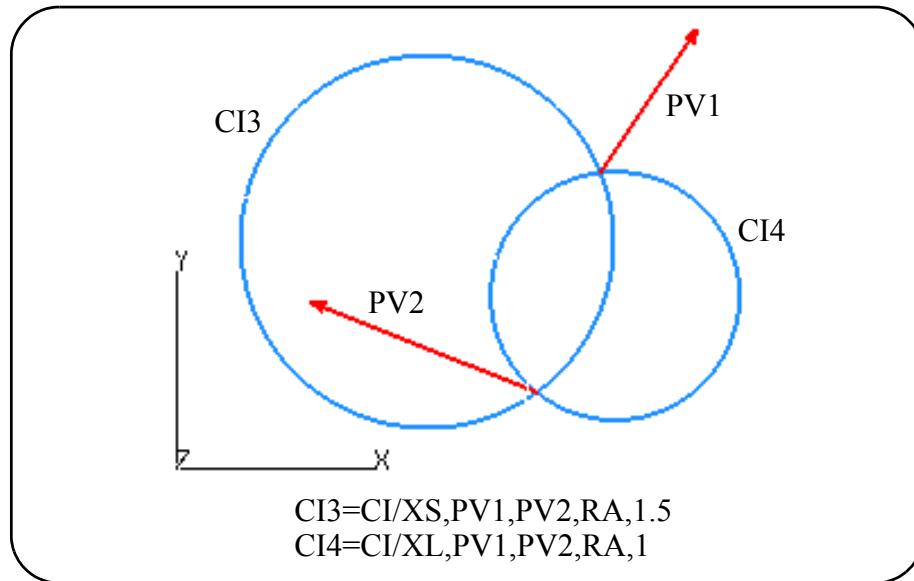
XLARGE
XSMALL point      point
CIRCLE/YLARGE,point-vector,point-vector,      $
YSMALL

RADIUS, radius

```

Icon Menu Sequence:





NOTE: The modifiers **XLARGE**, **XSMALL**, **YLARGE** and **YSMALL** designate which of the two possible circles is to be defined. The modifier describes the location of the center of the desired circle relative to the center of the other possible choice.

Calculation Method:

- The points or the point-vectors origin are projected onto the XY-plane of the "current" REference SYStem.
- The plane of the circle will be the XY-plane of the "current" REference SYStem.
- The arc length of the circle will be a full, unbounded 360 degrees.

Error Conditions:

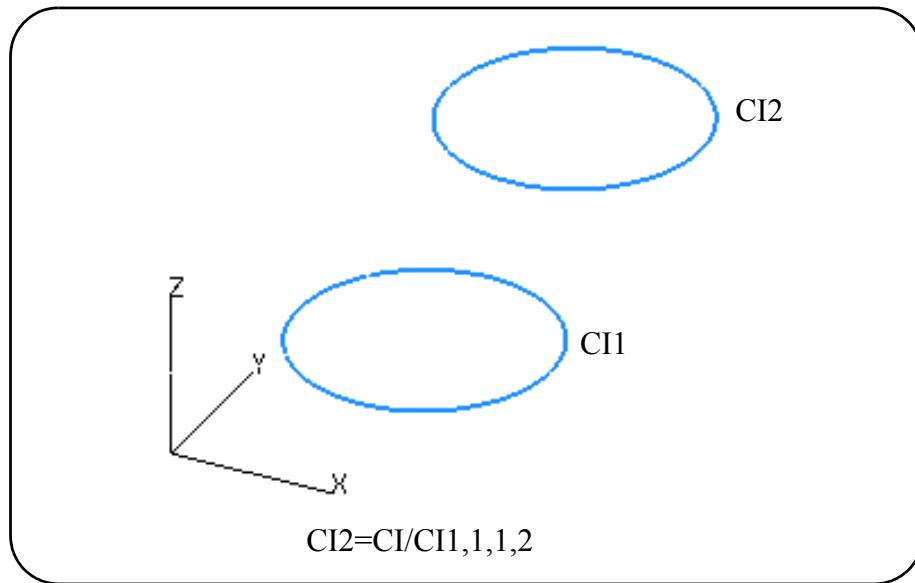
- The points or the point-vectors origin must not be coincident when projected.
- The diameter of the circle must be greater than the distance between the points and/or the point-vectors origin when projected.

3.1.17 A Circle Delta X, Y, Z From Another Circle

Command Syntax:

```
CIRCLE/circle[,delta-x[,delta-y[,delta-z]]]
```

Icon Menu Sequence:



NOTE: The delta values are optional, but the fixed field format MUST be observed; specify (0) for NO delta value.

Calculation Method:

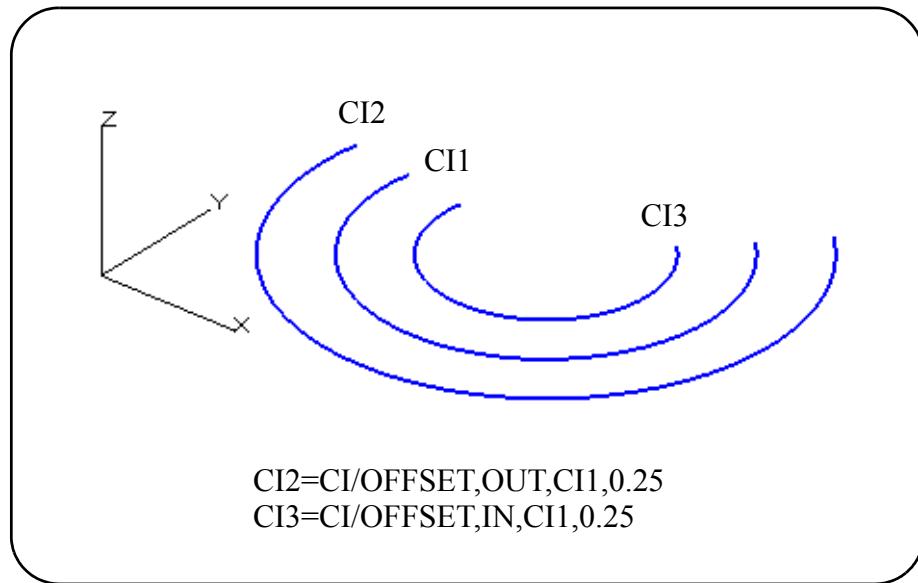
- The center of the defined circle will be offset by the given delta values from the center of the referenced circle.
- The radius, axis and limit plane values will be the same as the referenced circle.
- The arc length of the circle will be the same as the referenced circle.
- The direction of the circle will be the same as the referenced circle.

3.1.18 A Circle Offset From Another Circle

Command Syntax:

```
CIRCLE/OFFSET,OUT,circle,offset-value
      IN
```

Icon Menu Sequence:



Calculation Method:

- Both the defined circle and the referenced circle have the same center and circular plane.
- The radius and axis values will be the same as the referenced circle.
- The direction of the circle and the center angle will be the same as the referenced circle.

Error Condition:

- The offset-value specified is larger than or equal to the radius of the reference circle with IN specified.

3.1.19 A Circle By Canonical Form

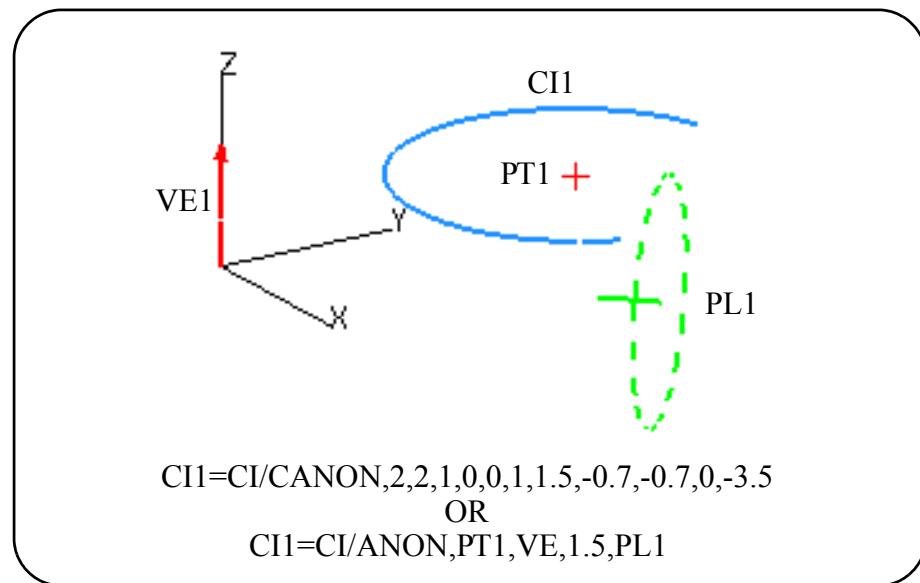
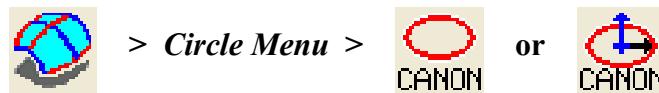
Command Syntax:

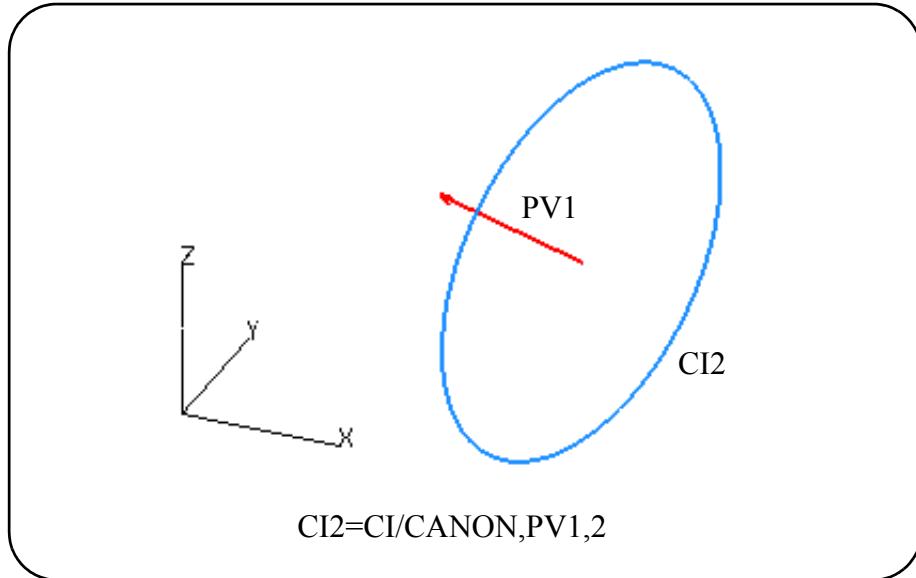
```
CIRCLE/CANON,x,y,z,i,j,k,r[,a,b,c,d]
```

or

```
CIRCLE/CANON,point-vector, radius[,plane]  
point, vector
```

Icon Menu Sequence:





Calculation Method:

- The plane of the circle is perpendicular to the axis of the circle as specified either by the i, j, k scalars or the vector.
- The arc length of the circle will be determined by the limit plane. If the optional limit plane is not specified, then the circle will be a full, unbounded 360 degrees.
- The direction of the circle will be determined either by the i, j, k scalars or the vector. Direction has no meaning for an unbounded circle.
- The normal vector of the limit plane points towards the section of the circle that will be kept if limit plane is specified.
- “d” is the actual distance along the normal vector of the cutting plane from the origin of the “current” REference SYstem. It will not be normalized as in the regular definition of a plane.

Error Conditions:

- If specified, the limit plane vector must be perpendicular to the circle axis vector.
- The limit plane must cut the defined circle.

3.1.20 Multiple Fillet

Command Syntax:

```
[nam(n) ]=FILLET/radius, line , line [...] , line ]
                           circle   circle      circle
```

Where:

nam(n) - Optional subscripted variable name given to arcs generated by the FILLET command. The initial subscript value (n) can be any positive integer in the range of 1 through 32767. If “nam” is not specified, then **NCL**’s auto naming feature will be used for naming the arcs.

Caveat: If “n” is specified as 32767, only the fillet created between the last two entities in the statement will exist due to the maximum allowable subscripted variable size of 32767. The rest of the entities would be trimmed to where the rest of the fillets would have been created, i.e. they would no longer be connected.

radius - Radius of the fillets.

Icon Menu Sequence:



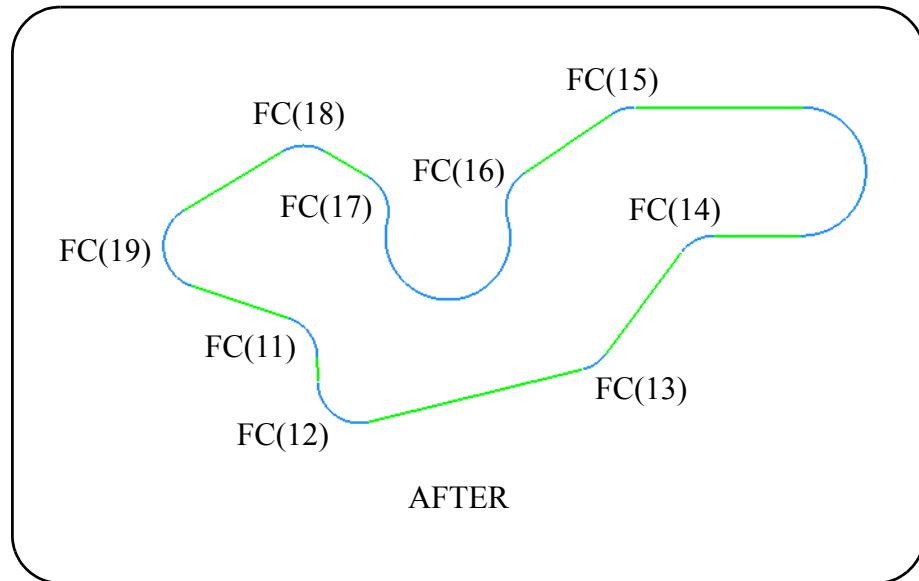
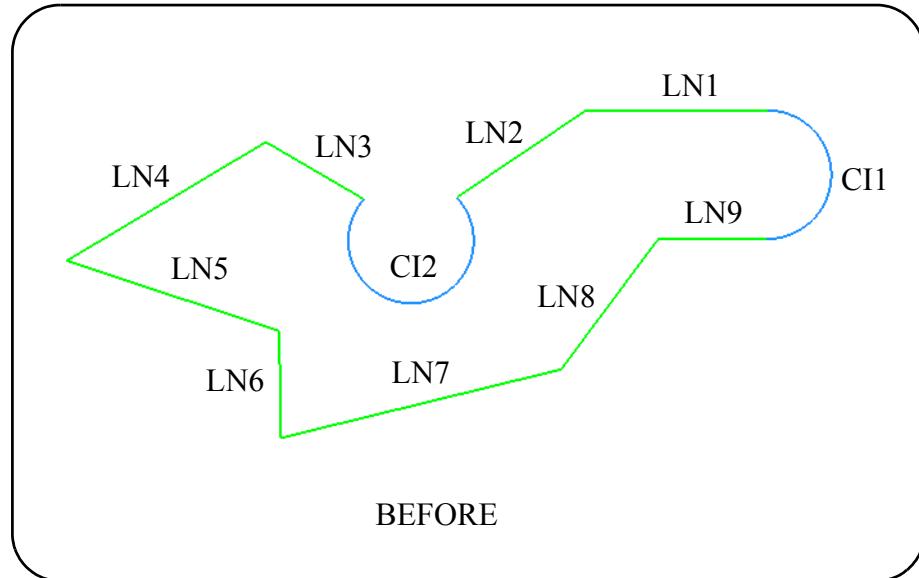
Allows you to create multiple fillets between coplanar connected lines and/or circles with a single command.

Note: If the first line or circle and the last line or circle are connected then the fillet will also be applied to this pair.

Example:

Illustration on next page shows the result of the statement:

```
FC(11)= FILLET/.5, LN5, LN6, LN7, LN8, LN9, CI1, LN1,      $
          LN2, CI2, LN3, LN4
```



CURVES

The following list gives an abbreviated notation of all the valid CURVE definition formats. See the individual sections for a complete explanation of each format.

1. **SPLINE/[FIT,]PV1 [. . .], PVn**
CURVE PT1[, VE1] PTn[, VEn]
 or **SPLINE/[FIT,]point, THRU, point, INCR, m**
CURVE
 or **SPLINE/[FIT,]pntvec, THRU, pntvec, INCR, m**
CURVE
 or **SPLINE/[FIT,] "any combination of above syntax"**
CURVE
2. **CURVE/CONIC, PT1, PT2, PT3, PT4, PT5**
 or **CURVE/CONIC, PT1, THRU, PT5**
3. **CURVE/CONIC, PT1, PT2, PT3, PT4, VE1**
 or **CURVE/CONIC, PT1, PT2, PT3, PV1**
 or **CURVE/CONIC, PT1, VE1, PT2, PT3, PT4**
 or **CURVE/CONIC, PV1, PT1, PT2, PT3**
4. **CURVE/CONIC, PT1, VE1, PT2, PT2, VE2**
 or **CURVE/CONIC, PV1, PT1, PV2**
 or **CURVE/CONIC, PV1, PV2, PV3**
 or **CURVE/CONIC, PV1, THRU, PV3**
5. **CURVE/COMPOS, [LINEAR, [CLOSE,]]line [[. . .], line]**
 SMOOTH OPEN circle circle
 CHAMFR curve curve
 OMIT spline spline
6. **SPLINE/curve [, Dx[, Dy[, Dz]]]**
CURVE spline
line
circle
7. **SPLINE/OFFSET, curve , modifier, distance**
CURVE spline
line
circle

-
8. **SPLINE**/OFFSET, compos_cv, ALL , modifier, \$
 CURVE comp1 [, comp2]
 distance [, LINEAR]
 SMOOTH
 CHAMFR
9. **SPLINE**/INTOF, SF1, SF2 [, near-point]
 CURVE pntvec
10. **SPLINE**/INTOF, SF1, PL1 [, near-point]
 CURVE pntvec
 or **SPLINE**/INTOF, PL1, SF1 [, near-point]
 CURVE pntvec
11. **SPLINE**/SF1 [, edge[[, PERCNT] , U-or-V[, num-points]]]
 CURVE
12. **SPLINE**/PART, ofset, surf[[, THRU] [, surf]] [...], \$
 CURVE n1 n2
 LAYER=n
 [LAYER=m , [...] [...]]\$
 surface[[, THRU] [, surface]]
 n3 n4
 AT, zlev
 plane
 planar-surf
13. **SPLINE**/OUT, SF1
 SSPLIN
14. **SPLINE**/OUT, SF1, bn[, en1[, CLW , en2]]
 SSPLIN CCLW
15. **SPLINE**/PROJCT, CV1, ATANGL, start, [end,] [SF2,] \$
 SSPLIN
 SF1[[u,v]]
16. **SPLINE**/PROJCT, CV1, [VE1,] SF1[[u,v]] [, NOWRAP] \$
 SSPLIN WRAP
 REVOLV
 RADIAL
 AT, PT3
 [, START] [, near-point]

MIDDLE	pntvec
END	
CENTER	

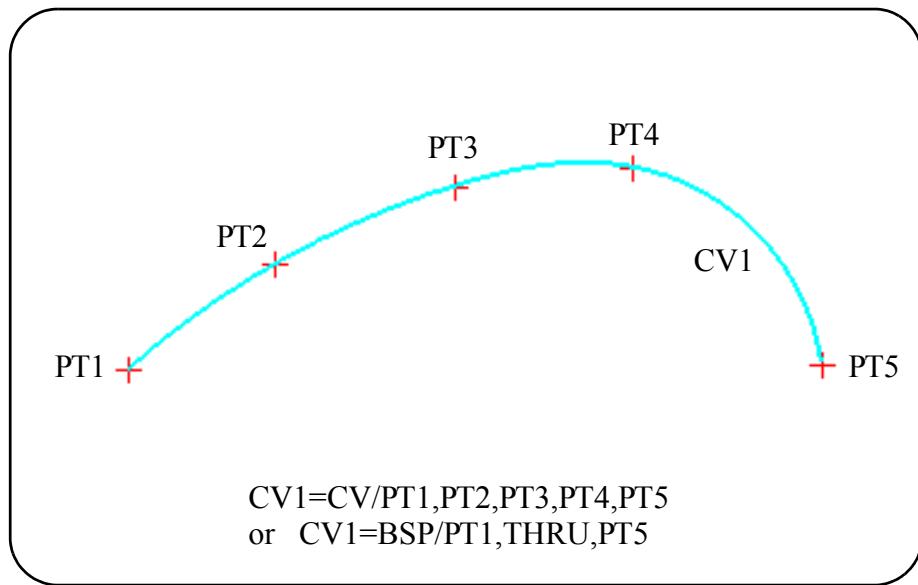
17. **SSPLIN**/surface[[,n] ,roff]
18. **SSPLIN**/INTOF,SF1,SF2[,near-point]
 PL1 pntvec
19. **SSPLIN**/COMPOS,ssplin[[...],ssplin]
20. **SSPLIN**/OFFSET,ssplin,modifier,distance

Example Curve Expressions

```
SPLINE/PT1,VE1,PT2,PT3,VE2,PT14
SPLINE/PT1,PV1,PT3,VE1,PT4
SPLINE/PV1,PV2,PV3,PV4,PV5,PV6
SPLINE/PV1,THRU,PV10
CURVE/PT1,PT2,PT3,PT4,PT5,PTX
SPLINE/PT1,THRU,PT5,PTX
SPLINE/FIT,1,THRU,30
CURVE/P(5),THRU,P(2)
CURVE/CV1
SPLINE/CV1,0,Y
SPLINE/INTOF,SF2,SF2A,PTSF2
CV/IO,SFH,SFV
CURVE/CONIC,PT1,VE22,PT3,PT35,PT46
CV/CONIC,PTA,PTB,PTC,PTD,PTE
CV/CONIC,PTC(1),VEC(1),PTC(2),PTC(3),VEC(2)
CV/CONIC,PV1,PV2,PV3
CV/CONIC,PV1,PT2,PT3,PT4
CV/CONIC,PT1,PT2,PT3,PV4
CV/COMPOS,LN1,LN3,CIA,CIB,CVCC1,CVCC2,LN9
SPLINE/PTNO(1),THRU,PTNO(120)
BSP/INTOF,SF1,SF2,PV3
BSP/IO,PL1,SF2,PV3
BSP/PTX(1),THRU,PTX(50),INCR,2
BSP/PTX(ALL)
SPLINE/OUT,SF1,0,1
```

3.2 CURVE Expressions

A Curve is a three-dimensional curve; it is the path of the least resistance through an ordered series of 3D points. The points must appear in sequence from the first to the last. However, they need not be in any particular plane. The tangent slope of the curve may be controlled at any point by specifying a tangent slope vector which is associated with the point. If the tangent slope vector is not input, **NCL** will generate the most appropriate slope at that point - based upon the position of the neighboring points. The extension of the curve is tangent to the slope at each end of the curve and extends infinitely in either direction.



Canonical Form:

- Due to the complexity of the curve, the canonical form is not included in this manual.

Special Conditions:

- **Light BLUE/CYAN** (pen 5) is the **NCL** System DEFAULT color for CAM CURVES.
- **Yellow** (pen 7) is the **NCL** System DEFAULT color for CAM Composite Curves.

Note: Instead of **Yellow** color, the color **ORANGE** will be used in examples of this chapter for better contrast with a white background.

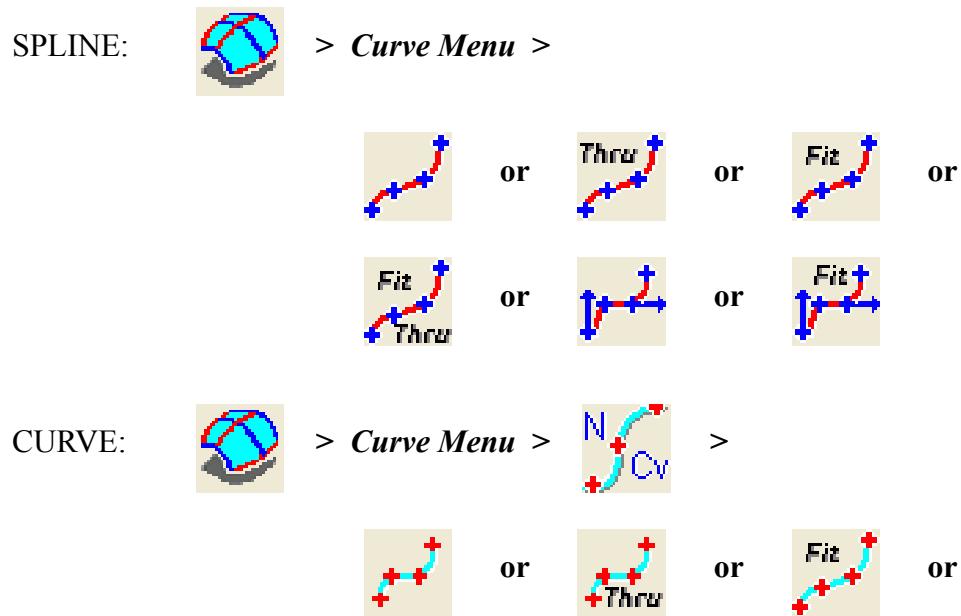
3.2.1 A Spline/Curve [FIT] Through A Series Of Combination Of Point-Vectors, And/Or Points With Optional Slope Control Vectors, With Optional Input Entities Skipped By A Constant Step

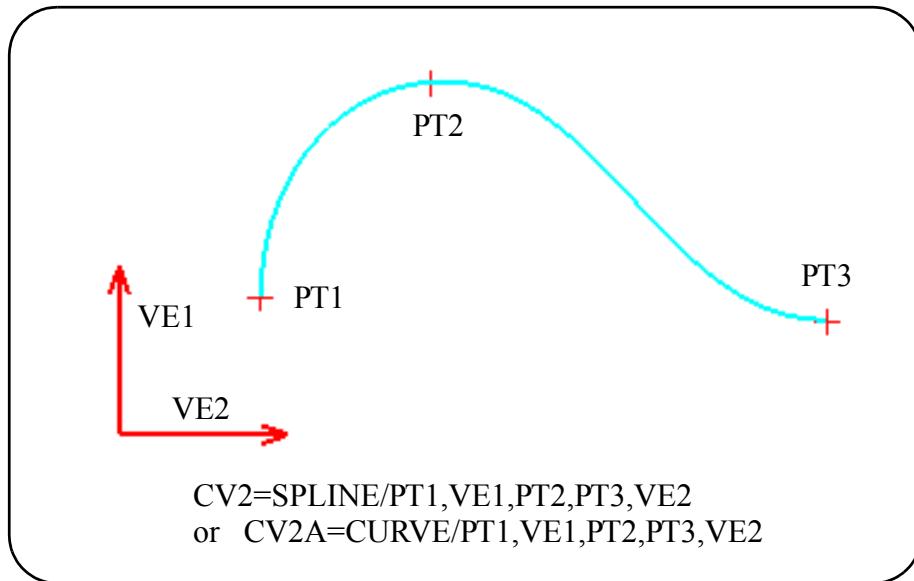
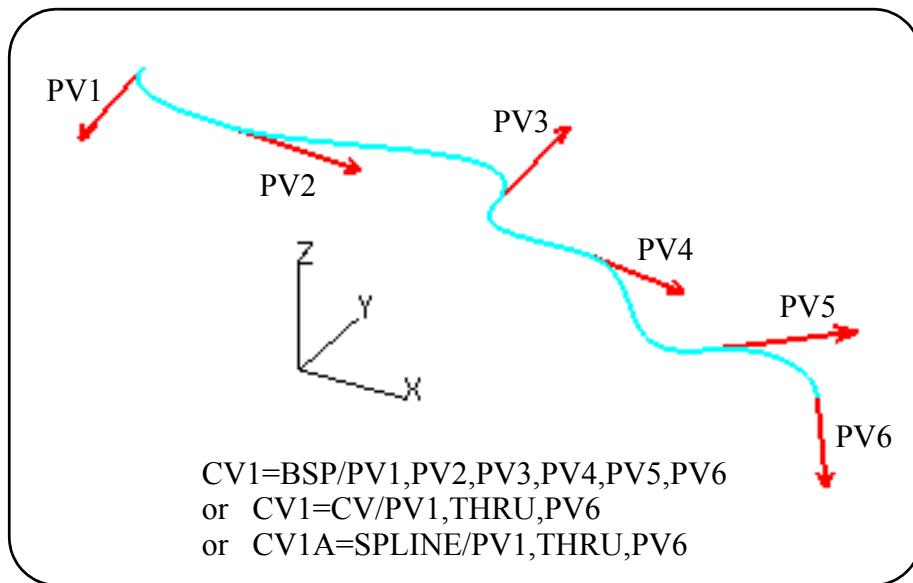
Command Syntax:

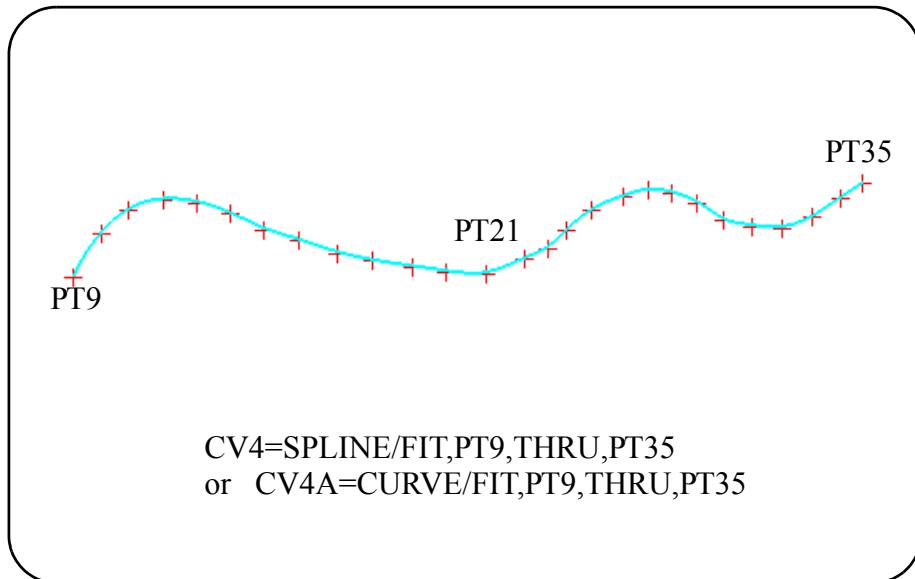
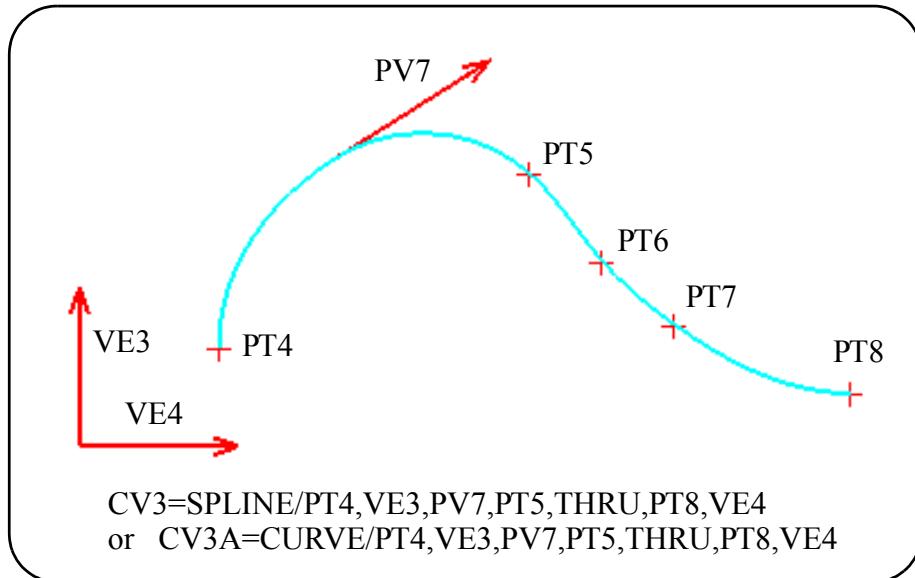
```
SPLINE/ [FIT,]point-vector , [...]point-vector
CURVE           point[,vector]      point[,vector]
                or
SPLINE/ [FIT,]point-vector,THRU,point-vector,
CURVE
                $  

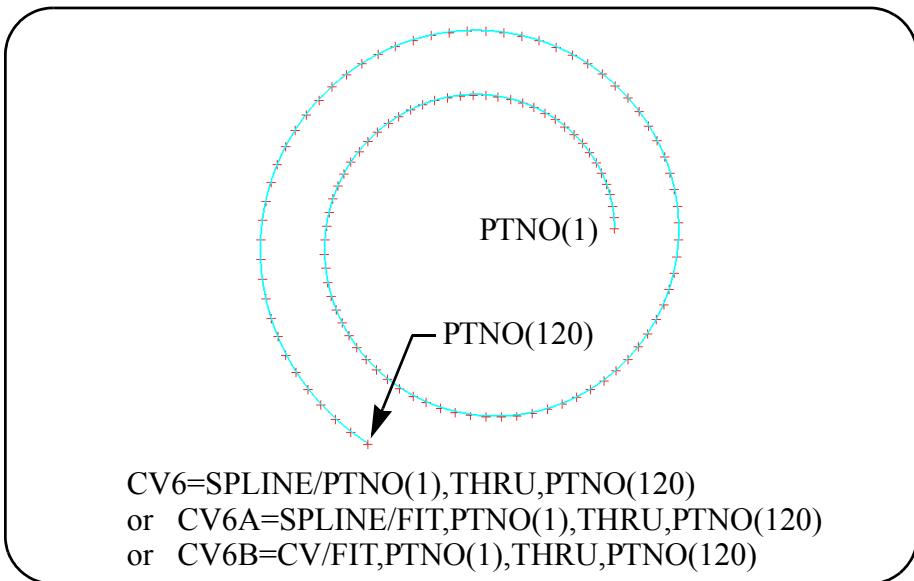
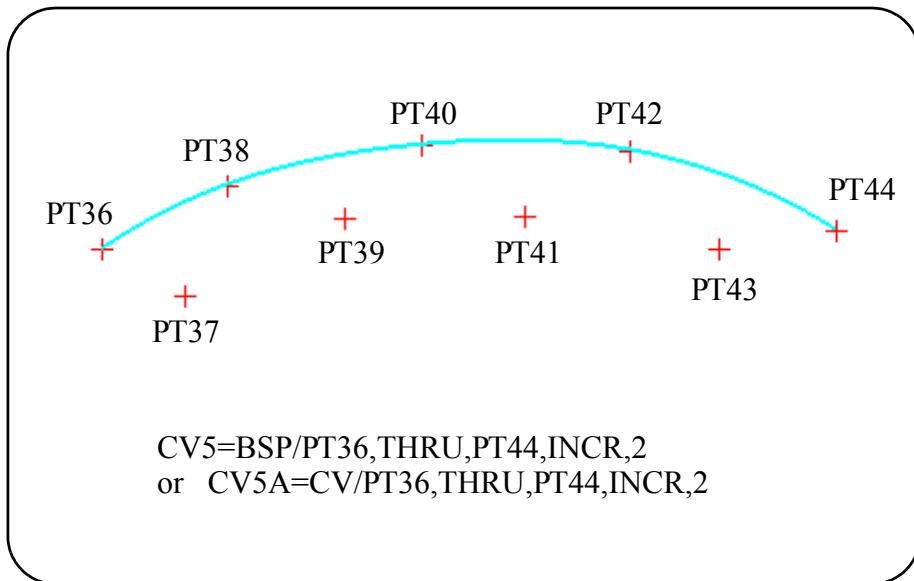
                INCR,m
                or
SPLINE/ [FIT,]point,THRU,point,INCR,m
CURVE
                or
SPLINE/ [FIT,] "any combination of above syntax"
CURVE
```

Icon Menu Sequence:









NOTE: If a subscripted array of points/point-vectors is defined and all its elements will be used to form a section of the curve/spline, this section of the curve/spline definition can be written as: point-array-name(ALL) or point-vector-array-name(ALL) providing that the optional "INCR/DECR,m" parameters are not specified with the definition.

Calculation Method:

- If SPLINE is specified instead of CURVE, a three-dimensional Rational B-spline will be created. Otherwise, a native **NCL** three-dimensional curve will be created.
- The optional "INCR.m" parameters specified how many points/point-vectors to be skipped. The number of points/point-vectors skipped will be determined by "m-1", i.e. if 2 is specified after INCR, every other input point/point-vector after the first input entity will be excluded from the calculation.
- Without the optional FIT parameter, the curve is calculated from start point (point-vector origin) to end point (point-vector origin), passing through all included points (point-vector origins) except those specified by the INCR amount "m".
- The vector portion of the point-vector determines the tangent slope at the associated point-vector.
- The optional vector determines the tangent slope at the associated point, i.e. the point specified before it.
- The following applies to SPLINES:
 - With the optional FIT parameter specified, **NCL** will attempt to eliminate as many points/point-vectors (except those excluded by the INCR option) as possible from the final curve while still maintaining a specified tolerance (as specified by the TOLER statement) to all input points/point-vectors.
 - Using FIT will result in greater efficiency during subsequent operations that may use this curve, especially when a large number of points/point-vectors have been specified.
- The following applies to CURVES:
 - With the optional FIT parameter specified, all points which include the point portion of the point-vectors, except those excluded by the INCR amount "m", are input to a curve fitting routine; a maximum of fifty will be used in the final curve.
 - With the optional FIT parameter specified, all the original points which include the point portion of the point-vectors, except those excluded by

the INCR amount “m”, will lie on the curve within the current value of TOLER.

Error Conditions:

- A minimum of two points, two point-vectors, or one point and one point-vector is required.
- The specified points/point-vectors used with the optional THRU must have been generated by using the autoname generation feature or the default PT/PV identifier prefix in naming. The points/point-vectors may be specified in the statement by their full identifier (PT4/PV4), or an identifier with subscript (PTX(4))/(PVX(4)).
- If the optional “INCR,m” parameters are specified, the total number of input entities must equal to $\text{INT}(n/m)*m+1$.

where n: total number of input entities

m: value specified after the parameter INCR

INT: Integer function

- The following applies to SPLINEs:
 - A maximum combination of 999 point-vectors and/or points with optional vectors is allowed without the optional FIT parameter specified.
 - Errors such as “Conditions too severe” (051) or “More than 50 S-values generated” (156) may occur if the specified points/point-vectors do not form a smooth enough curve.
- The following applies to CURVEs:
 - A maximum combination of fifty point-vectors and/or points with optional vectors is allowed without the optional FIT parameter specified.
 - If the optional FIT parameter is specified, there is no maximum for the number of points specified. The “fitted” curve will only utilize a combination of the maximum fifty point-vectors and/or points with optional vectors. **NCL** will attempt to remove extraneous input entities from the curve calculation using the current TOLER value, however, the point-vectors or points with associated optional vectors will not be removed.

3.2.2 A Curve That Approximates A Conic Section Using Five Points

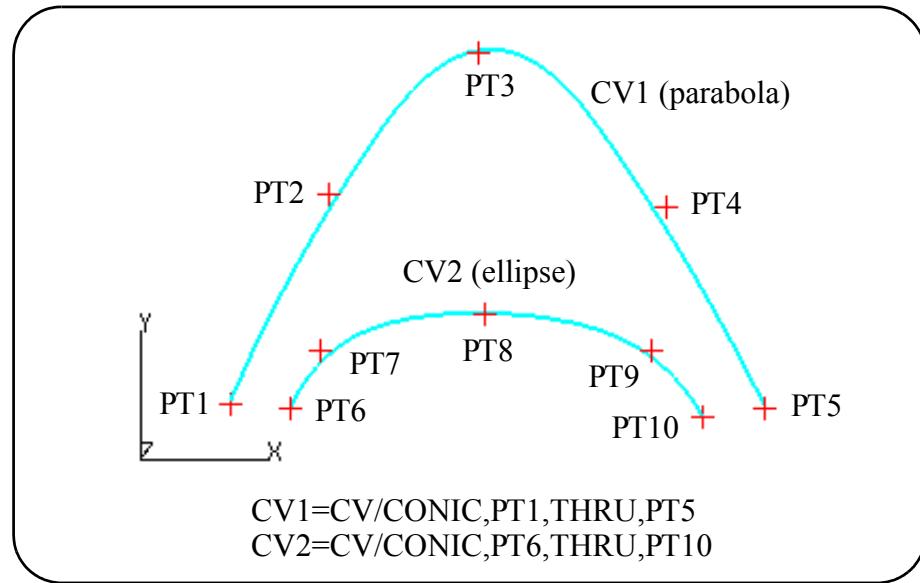
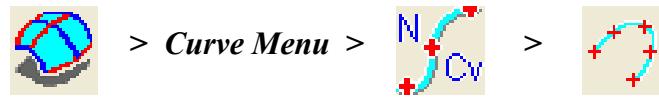
Command Syntax:

CURVE/CONIC,point,point,point,point,point

or

CURVE/CONIC,point,THRU,point

Icon Menu Sequence:



Calculation Method:

- A curve is closely fitted to approximate a conic section if possible.
- The five points must all lie in a common plane.
- The extension of a conic curve is a straight line extending from the end point in the direction of the slope at that end point.

3.2.3 A Curve That Approximates A Conic Section With Three Points And One Point Slope At Either End

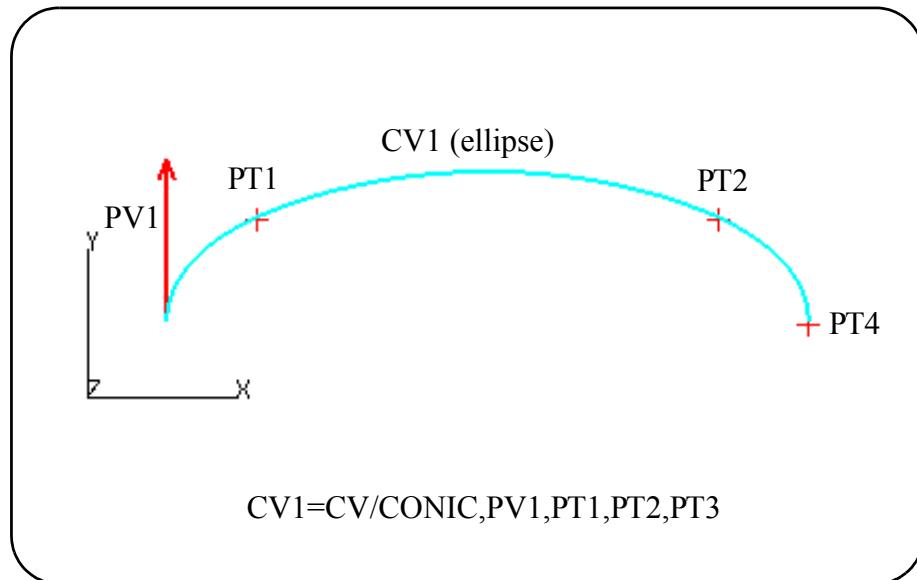
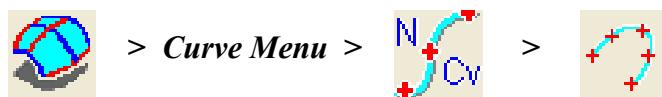
Command Syntax:

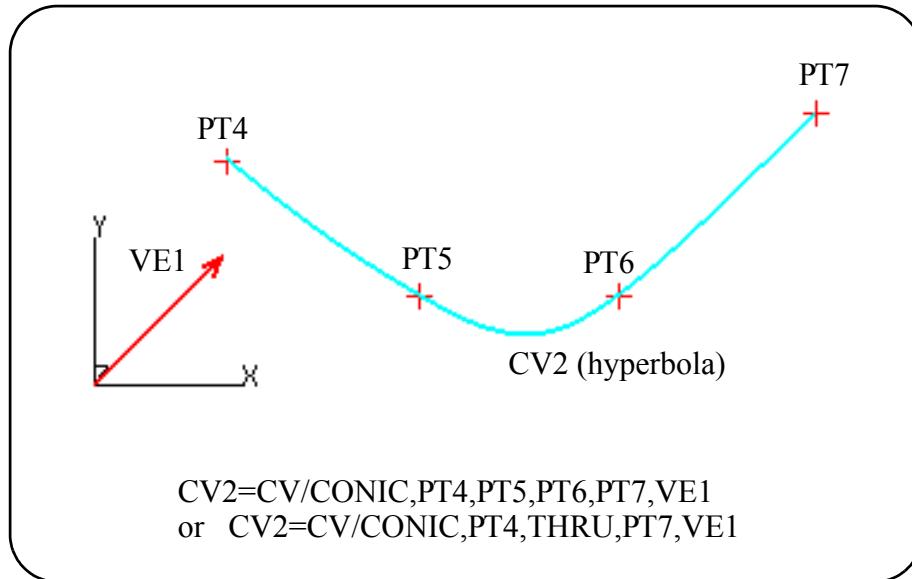
```
CURVE/CONIC,point-vector,point,point,point  
point,vector
```

or

```
CURVE/CONIC,point,point,point,point-vector  
point,vector
```

Icon Menu Sequence:





Calculation Method:

- A curve is closely fitted to approximate a conic section if possible.
- The four or five entities (points, vectors and/or point-vectors), must all lie in a common plane.
- The extension of a conic curve is a straight line extending from the end point in the direction of the slope at that end point.

3.2.4 A Curve That Approximates A Conic Section Using Two Point Slopes At Both Ends And One Additional Point/Point-Vector In Between

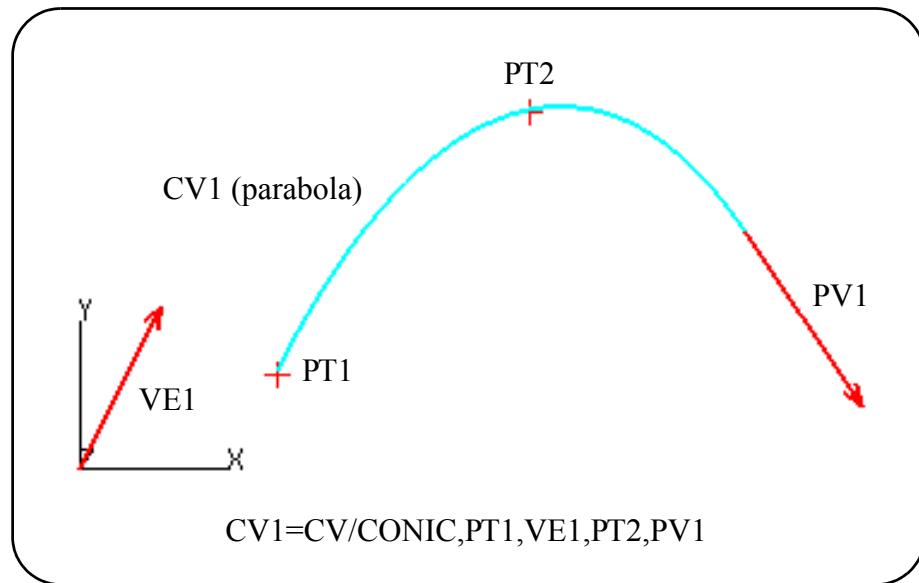
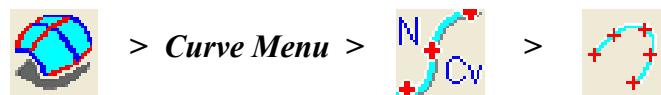
Command Syntax:

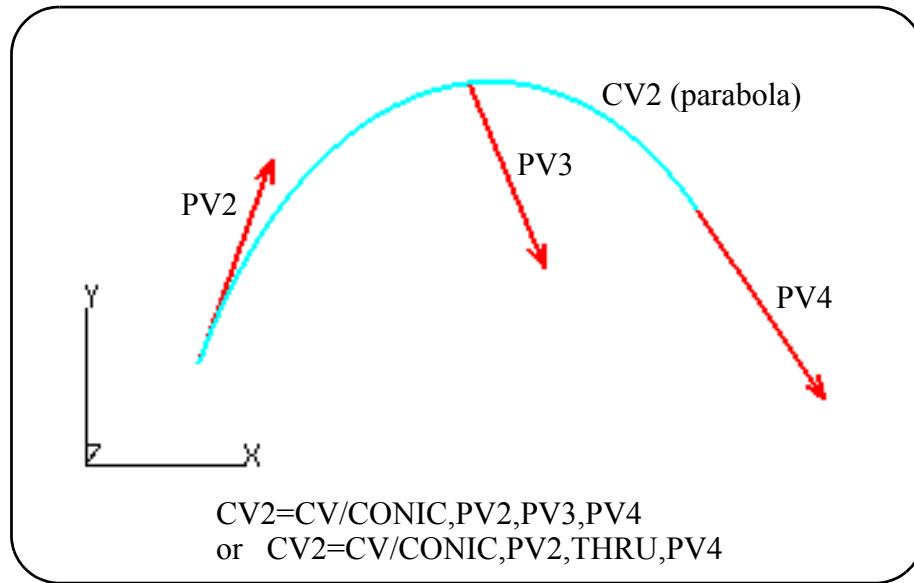
```
CURVE/CONIC,point-vector,point,point-vector  
                  point,vector              point,vector
```

or

```
CURVE/CONIC,point-vector,point-vector,point-vector
```

Icon Menu Sequence:





Calculation Method:

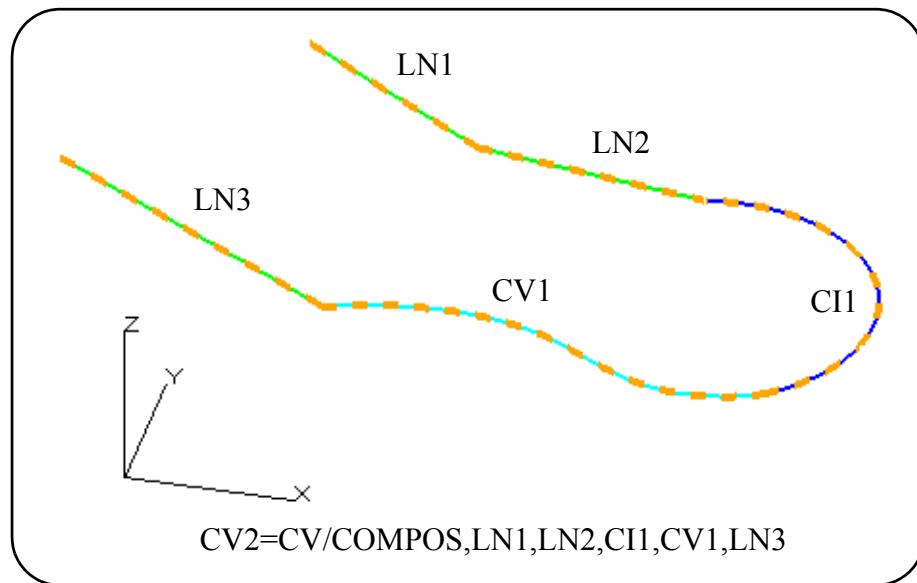
- A curve is closely fitted to approximate a conic section if possible.
- The three or five entities (points, vectors and/or point-vectors), must all lie in a common plane.
- If three point-vectors are specified, the second point-vector origin will be used as the required point and the vector section will be ignored.
- The extension of a conic curve is a straight line extending from the end point in the direction of the slope at that end point.

3.2.5 A Curve As A Composite Collection Of Lines, Circles And/Or Curves

Command Syntax:

```
CURVE/COMPOS, [LINEAR, [CLOSE,]] line [...] ,line ]
              SMOOTH   OPEN    circle    circle
              CHAMFR   curve   curve
              OMIT     spline  spline
```

Icon Menu Sequence:



This type of curve may be used in any geometry or motion type statement except an **NCL SURF** definition. “CURVE” can be specified as “SPLINE”, they are interchangeable in the case of composite curves.

Calculation Method:

- The curve is made up of all the specified geometry.

- A composite curve can be created from disconnected entities by specifying a connection type with the optional parameters as described below.

When LINEAR is specified, any required connections will be made by linearly extending the disjointed geometry so that they intersect. If the extensions do not intersect, then a line connecting the end points of the two entities is created (same as if CHAMFR is specified).

When SMOOTH is specified, all connections will be made using B-Splines to create a smooth transition. The B-Splines are defined using the tangent vectors at the ends of the entities.

When CHAMFR is specified, all connections will be made using a line between the end points of the two entities. Unlike LINEAR, lines will not be extended.

The default connection type is OMIT and no connections will be made, rather an error will be output if the geometry is not connected.

CLOSE specifies that the ends of the curve should also be joined so the resulting curve is closed.

OPEN is the default and specifies that no additional connection will be made to join the ends of the curve.

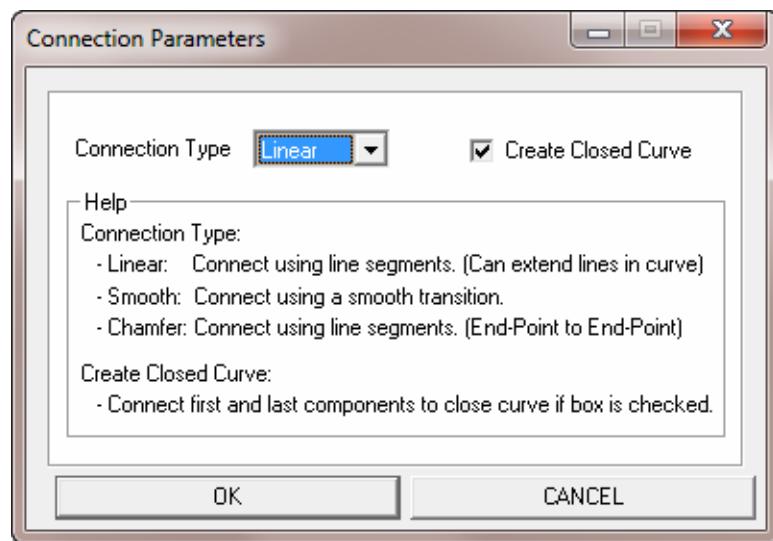
Note: If a connection type is given when defining a composite curve and two entities that are being connected intersect, they will automatically be trimmed. When the second of the two curves is trimmed, the side closest to its next neighbor will be kept.

Additional geometry is defined when making connections between components.

When using LINEAR connections a new line will not be created if one of the entities is a line and extending it will cause it to intersect its neighbor or neighbor's extension.

If the composite curve is created with interface icon sequence, the Connection Parameters interface form as shown on next page pops up if there is disconnection detected or when the first and last components intersect so a closed curve can be created. This form allows the user to:

- Specify the connection type with the Connect Type toggle menu.
- Specify whether or not to create a closed curve by checking the Create Closed Curve checkbox. If the first and last entities selected are already connected, then the Create Closed Curve checkbox will not be enabled.



Error Conditions:

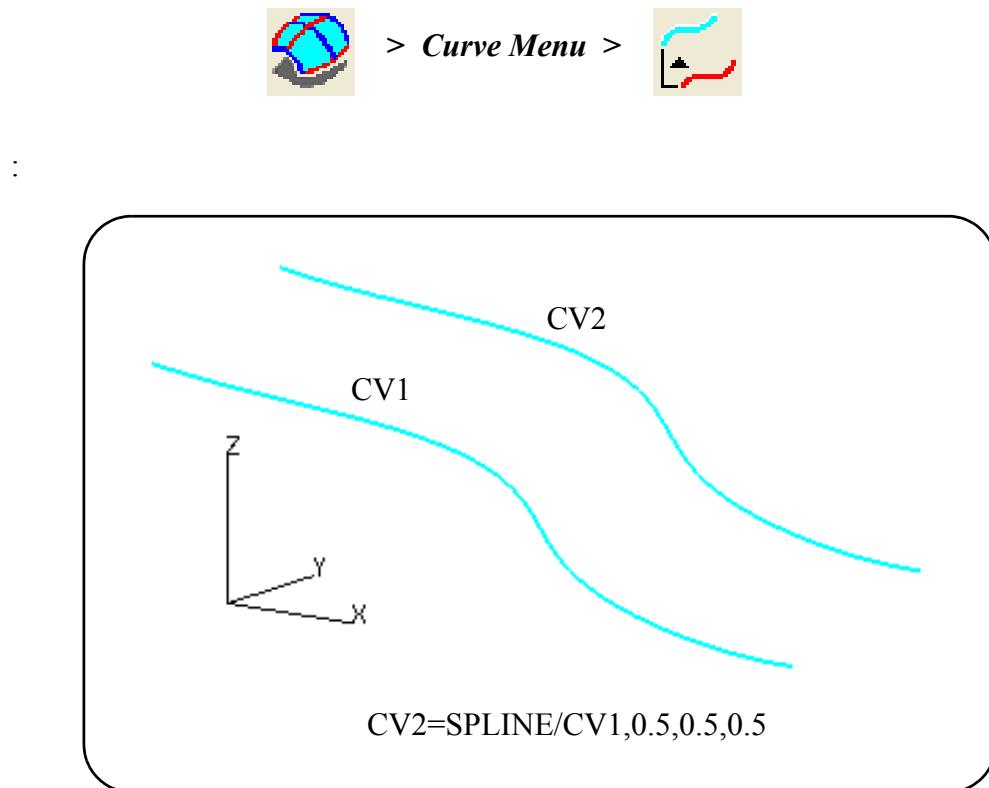
- Only wire frame type entities (line, circle, curve, spline, composite curve) are allowed as input to a composite curve definition.

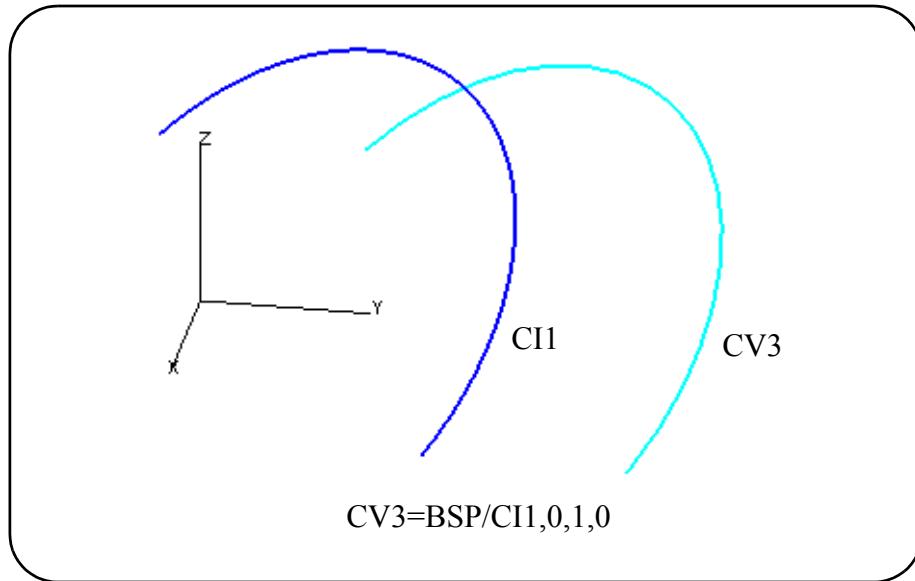
3.2.6 A Spline/Curve Translates From A Wire Entity At A Delta Distance

Command Syntax:

```
SPLINE/curve [,delta-x[,delta-y[,delta-z]]]  
CURVE spline  
circle  
line
```

Icon Menu Sequence:





Calculation Method:.

- If the input wire entity is of the same type specified, the curve created is identical to the one used to calculate it except that it is translated to a different location by the optional delta-x, delta-y, delta-z values given.
- If the input wire entity type is different from what is specified, a copy of it will be converted to the specified type and translated to a different location by the optional delta-x, delta-y, delta-z values to create the new curve. The original input wire entity will not be affected and is still the original type. The new curve will be of the type specified.
- The fixed field format MUST be observed when specifying the delta values. Zero (0) should be used when no value is desired.

3.2.7 A Spline/Curve Offset From A Wire Entity In A Direction At A Distance

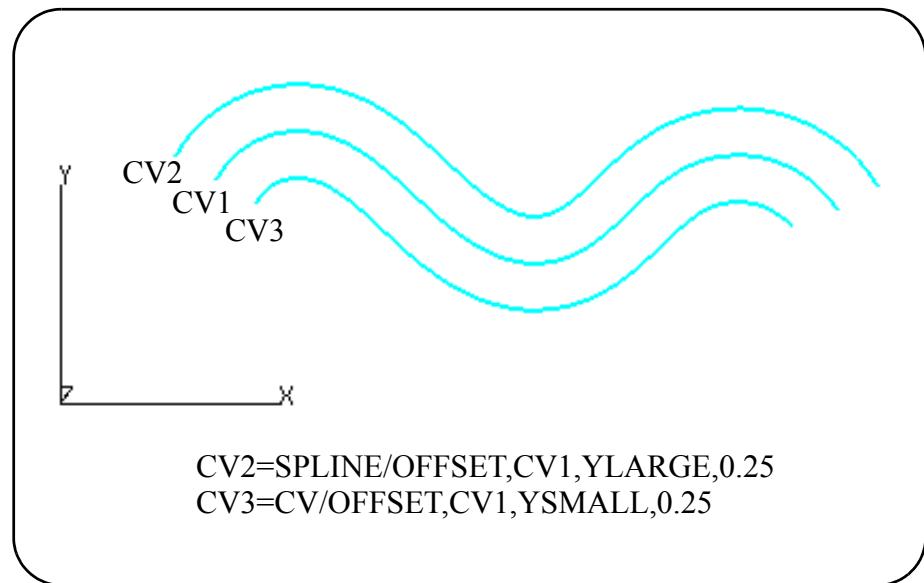
Command Syntax:

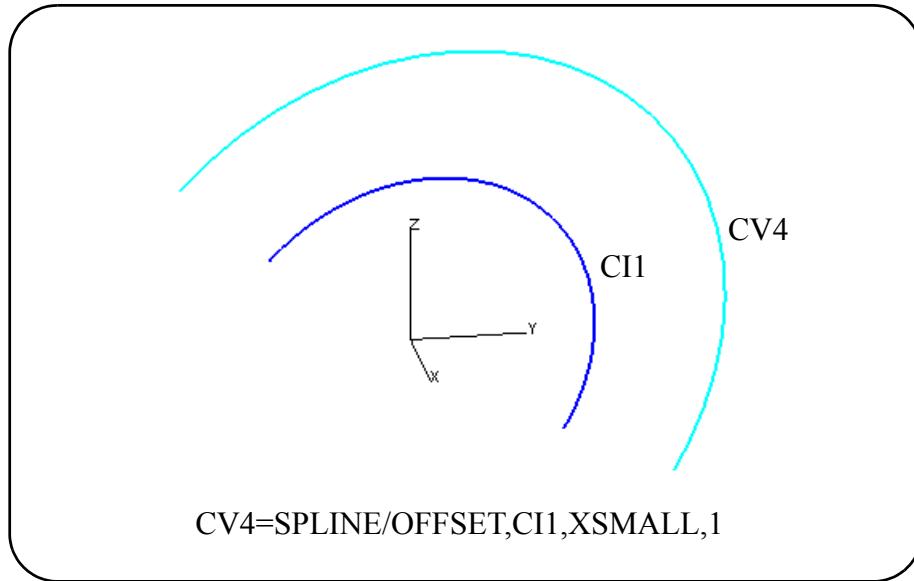
```

          XLARGE
          XSMALL
curve   YLARGE
SPLINE/OFFSET,spline,YSMALL,distance
CURVE      circle ZLARGE
line       ZSMALL
vector
pntvec

```

Icon Menu Sequence:





Calculation Method:

- This curve definition creates a curve by moving points of a given wire entity by a fixed distance along the principal normal at these points.
- The offset direction is specified by a direction modifier, which is applied to the starting point of the curve. (Thus the same offset statement could produce different results if the wire entity is reversed.)
- A B-Spline will be created if “SPLINE” is specified disregard the original type of the specified wire entity.
- A native **NCL** type curve will be created if “CURVE” is specified disregard the original type of the specified wire entity.
- Offsetting a 2-D curve (which lies in the current XY plane) with a 2-D modifier (i.e. **XLARGE**, **XSMALL**, **YLARGE**, or **YSMALL**) results in a 2-D curve. If **ZLARGE** or **ZSMALL** is used with a 2-D curve, the result will be a vertical translation. For example, assuming that “CV1” is a 2D curve lying on the current XY plane:

`CV/OFFSET , CV1 , ZLARGE , 1`

Would be equivalent to:

`CV/CV1 , 0 , 0 , 1`

- A point or a point-vector can be used to define the offset direction.
- SPLINE/OFFSET statement has more rigorous tolerance adjustment logic than CURVE/OFFSET.

Error Condition:

- The created curve may not be zero length when projected.

3.2.8 A Composite Curve Offset From A Composite Curve In A Direction At A Distance

Command Syntax:

```

          XLARGE
          XSMALL
          YLARGE
SPLINE/OFFSET, compos_cv, ALL      , YSMALL,      $
          CURVE           comp1 [, comp2]  ZLARGE
                           ZSMALL
                           vector
                           pntvec

distance[, LINEAR]
          SMOOTH
          CHAMFR

```

Icon Menu Sequence:



where:

- “compos_cv” is the composite curve to be offset
- “comp1[,comp2]” define the numeric index range for the components to be offset. For example, 4,6 specifies that only the fourth through sixth curve of the composite curve will be offset. If the parameter ALL is specified, all components will be offset starting at the first component of the curve.
- The directional modifier (XLARGE, YSMALL, etc.) specifies the direction from the first component referenced to offset the curve. A “vector” or “pntvec” can be used to define the offset direction.
- “distance” specifies the offset distance.
- LINEAR, SMOOTH and CHAMFR specify the connection type to be used if needed. The connection types are the same as defined above. LINEAR is the default connection type.

Calculation Method:

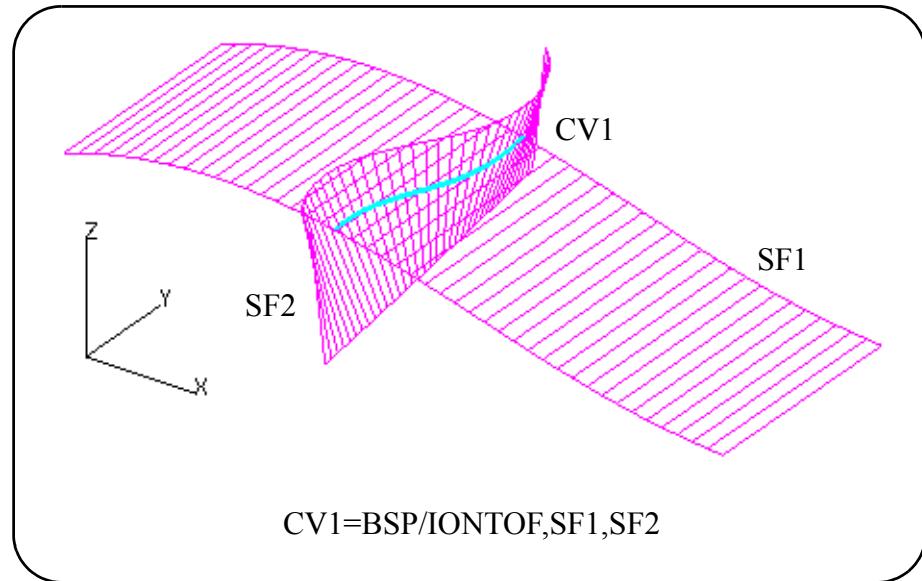
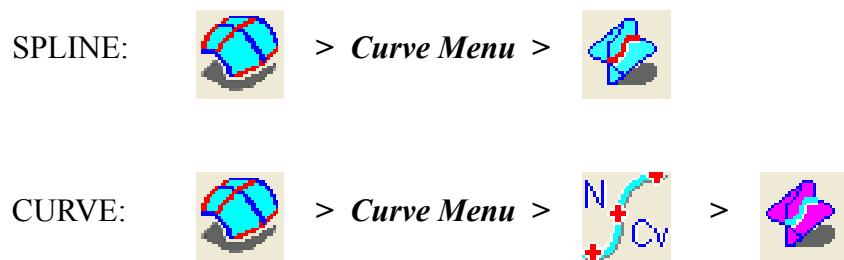
- When components of a composite curve are offset, the component geometry of the original curve is maintained and new geometry is created and offset to define the new curve.
- If a connection type is given when defining a composite curve and two entities that are being connected intersect, they will automatically be trimmed. When the second of the two curves is trimmed, the side closest to its next neighbor will be kept.
- Additional geometry is defined when making connections between components.
- Note when using LINEAR connections a new line will not be created if one of the entities is a line and extending it will cause it to intersect its neighbor or neighbor's extension.
- When using a vector to define the offset direction, the start of the offset will be found using the given vector and the first forward vector of the first component to be offset. The offset direction will be perpendicular to the first forward vector and will be on the plane defined by the forward vector and the given vector.

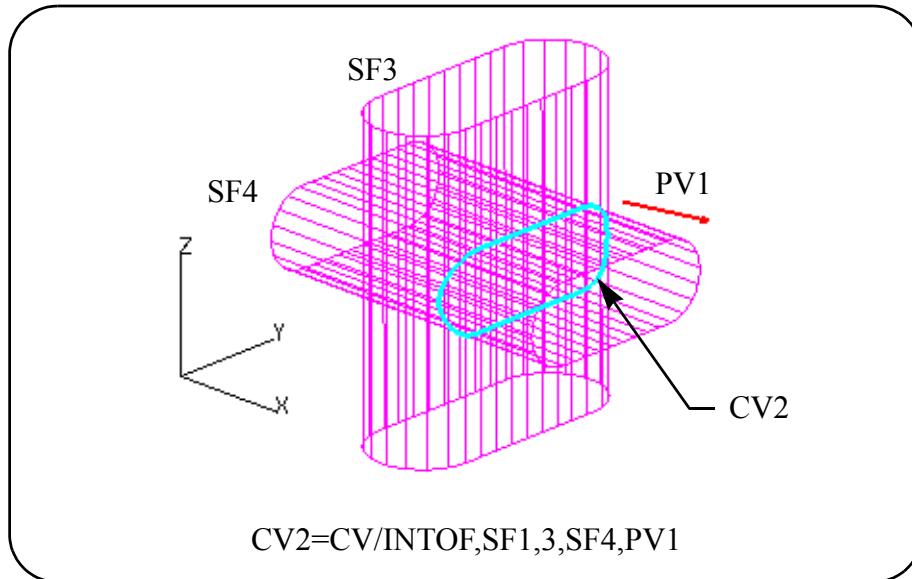
3.2.9 A Spline/Curve Defined As The Intersection Of Two Surfaces With An Optional Near Point Or A Point-Vector

Command Syntax:

```
SPLINE /INTOFSurface,surface[,near-point ]
          CURVE                                pntvec
```

Icon Menu Sequence:





Calculation Method:

- A B-Spline will be created if “SPLINE” is specified. A **NCL** native curve will be created if “CURVE” is specified.
- The curve is created at the intersection of the two referenced surfaces within a tolerance given by the current value of **TOLER**.
- The curve will only exist along the points that are common to the bounded area of both intersecting entities. A near-point may be given to clarify which of several possible intersections of the entities is desired.
- The generated curve will respect the outside and inside boundaries of a trimmed surface. A near-point may be given to clarify which of several possible intersections of the entities is desired.
- If the “***SET/VER,flag**” is used and the value of flag is smaller than 9.15, “SPLINE” cannot be specified. Also, the underlying untrimmed surface will be used for calculation if the input surface is of the type **TRIMMED**, i.e. the outside boundary of the trimmed surface will not be respected.

Error Conditions:

- The extensions of the surfaces are not used when determining the intersection points.
- Surfaces that contain large flat or straight sections next to a section of sharp curvature may not produce a curve that is within the tolerance of the

surface. The only "work-around" would be to lower the current value of TOLER.

NOTE: **NET surface** is not recommended for this statement due to complexity of intersection condition. Unexpected results might be generated.

3.2.10 A Spline/Curve Defined As The Intersection Of A Surface And A Plane With An Optional Near Point Or Point/Vector

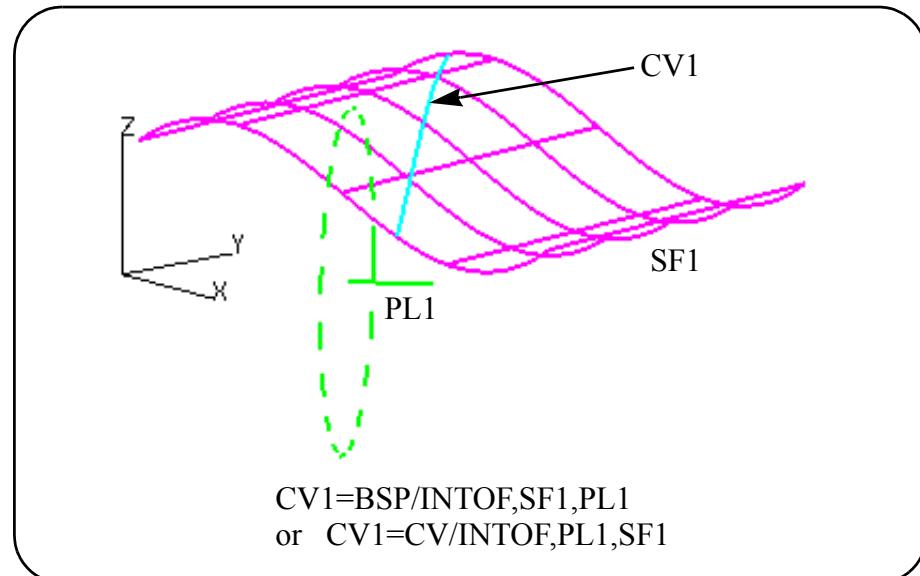
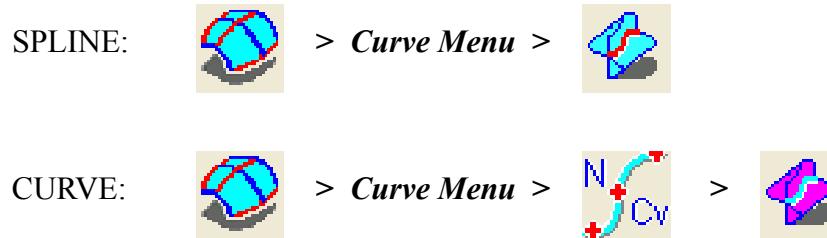
Command Syntax:

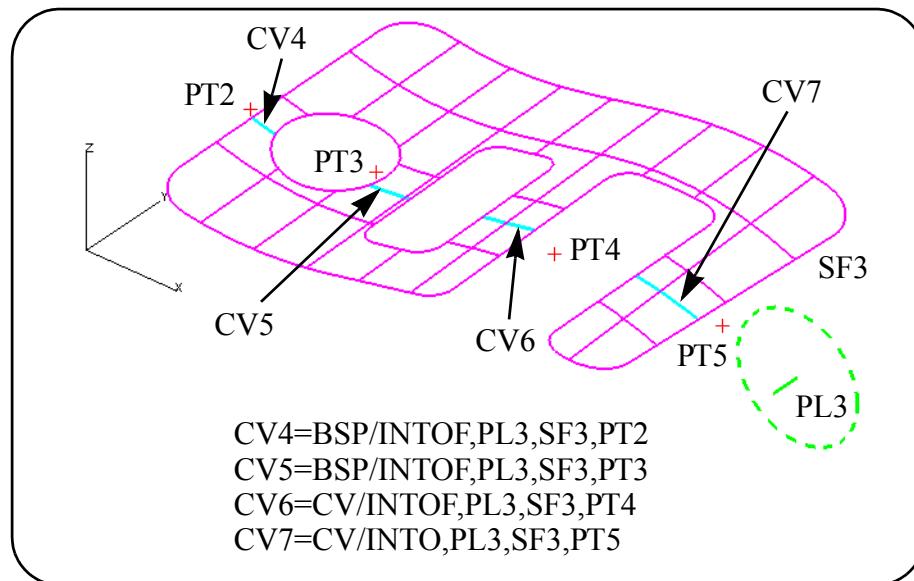
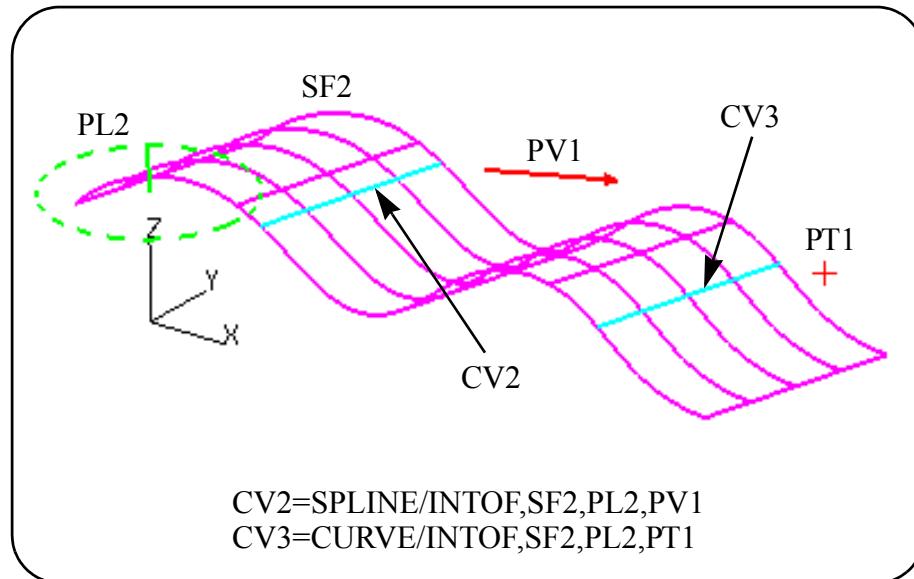
```
SPLINE/INTOF,surf,plane[,near-point ]
CURVE                                pntvec
```

or

```
SPLINE/INTOF,plane,surf[,near-point ]
CURVE                                pntvec
```

Icon Menu Sequence:





Calculation Method:

- A B-Spline will be created if “SPLINE” is specified. A **NCL** native curve will be created if “CURVE” is specified.
- The curve is created at the intersection of the surface and the plane within a tolerance given by the current value of **TOLER**.

- The curve will only exist along the points that are common to the bounded area of both intersecting entities. A near-point may be given to clarify which of several possible intersections of the entities is desired.
- The generated curve will respect the outside and inside boundaries of a trimmed surface. A near-point may be given to clarify which of several possible intersections of the entities is desired.
- If the “***SET/VER,flag**” is used and the value of flag is smaller than 9.15, “SPLINE” cannot be specified. Also, the underlying untrimmed surface will be used for calculation if the input surface is of the type **TRIMMED**, i.e. the outside boundary of the trimmed surface will not be respected.

Error Conditions:

- The extension of the surface is not used when determining the intersection points.
- Surface that contain large flat or straight sections next to a section of sharp curvature may not produce a curve that is within the tolerance of the surface. The only "work-around" would be to lower the current value of **TOLER**.

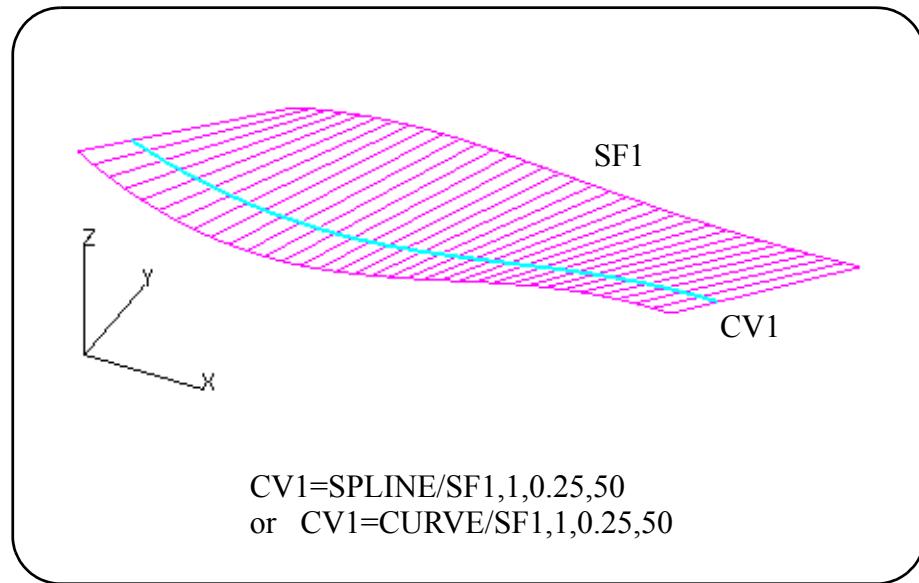
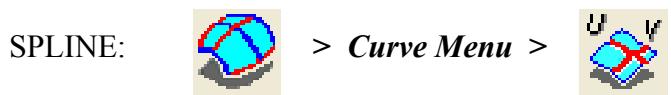
NOTE: **NET surface** is not recommended for this statement due to complexity of intersection condition. Unexpected results might be generated.

3.2.11 A Spline/Curve As The Edge Of A Surface Or Any U Or V Offset On The Surface

Command Syntax:

```
SPLINE/surf [ , edge [ [ , PERCNT] , U- or V-offset      $  
CURVE  
[ , num-points] ] ]
```

Icon Menu Sequence:

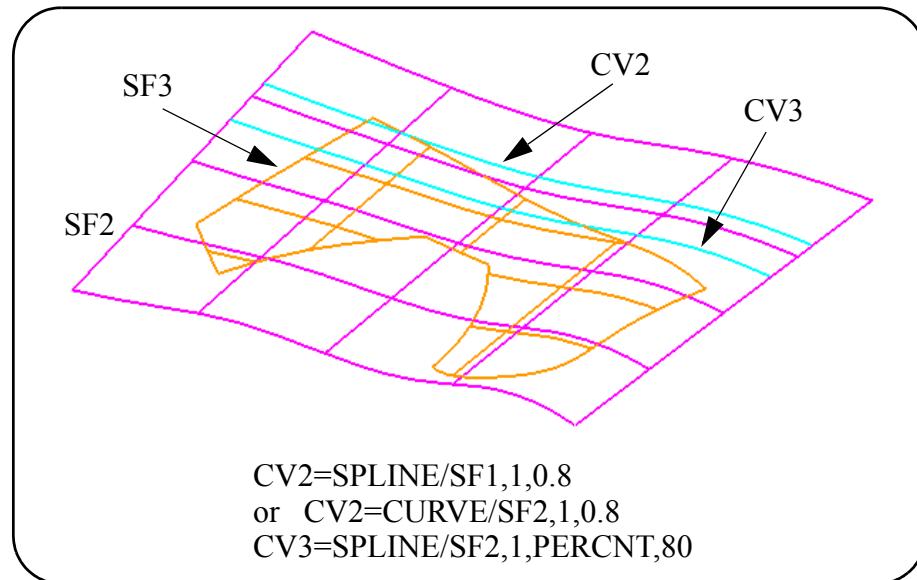


Defines a curve that lies on a surface either at one of the edges of the surface or some distance in from an edge.

The edge of the surface to be used is specified by the "edge" parameter. The value of "edge" may be 1, 2, 3 or 4. A value of one (1) indicates the edge along the U direction should be used. The U direction edge is the first entity specified when the surface was defined. A value of two (2) specifies that the edge opposite the U direction "edge" (last entity specified when surface was defined) should be used. An "edge" value of three (3) will cause the curve to follow the "edge" made up of the beginning points of each defining geometric entity of the surface. The direction of the curve is given by the order of the entities specified when the surface was defined. A value of four (4) indicates the edge opposite the V direction "edge" should be used. The default value for "edge" is 1.

A "U or V-offset" may be specified to indicate the curve should lie on the surface at a certain distance away from the specified "edge." This offset value must be between 0 and 1. A value of 0 is the default and indicates a curve at the specified "edge." A value of 1 indicates a curve at the opposite edge of the surface from that specified by "edge." The value represents a percentage along the surface in either the U or V direction. For example, a value of .5 along with an "edge" value of 1 would indicate a curve half way between the two opposing U edges (50% along the V direction) of a surface.

The optional parameter "PERCNT" denotes the U or V-offset values specified are average percentage along the surface U-lines and V-lines. For a trimmed surface the percentage is taken on the base surface restricted to the (umin,umax,vmin,vmax) bounding box of the outer boundary.



The "num-points" parameter specifies the number of points along the surface to be used in defining the curve.

Calculation Method:

- An equally spaced set of "num-points" is calculated along the specified "edge" of the surface.
- If "SPLINE" is specified, the same methods used by "SPLINE/FIT" are used to define the final canonical data for the curve.
- If "CURVE" is specified, the same methods used by "CURVE/FIT" are used to define the final canonical data for the curve.
- The underlying untrimmed surface will be used for calculation if the input surface is of the type **TRIMMED**.

Error Conditions:

- Surfaces that contain large flat or straight sections next to a section of sharp curvature may not produce a curve that is within the tolerance of the surface. The only "work-around" would be to lower the current value of **TOLER**.
- This statement is not valid for **QUILT** or **NET** surfaces.

3.2.12 A Spline As A 2-Dimensional Outline Of A Group Of Surfaces

Command Syntax:

```
SPLINE/PART,offset,surface-list,AT,zlev
  CURVE                                plane
                                         planar-surf
```

Where:

offset Offset value. A positive value expands the calculated curve and a negative value shrinks the calculated curve by the value specified.

surface-list List of surfaces that the closed spline would be created from. The list can be specified as follows:

- A standard sequence of labels, e.g., sf2, sf3, etc.
- A sequence indicated by a “THRU” clause.

Example:

sf1, THRU, sf5

or

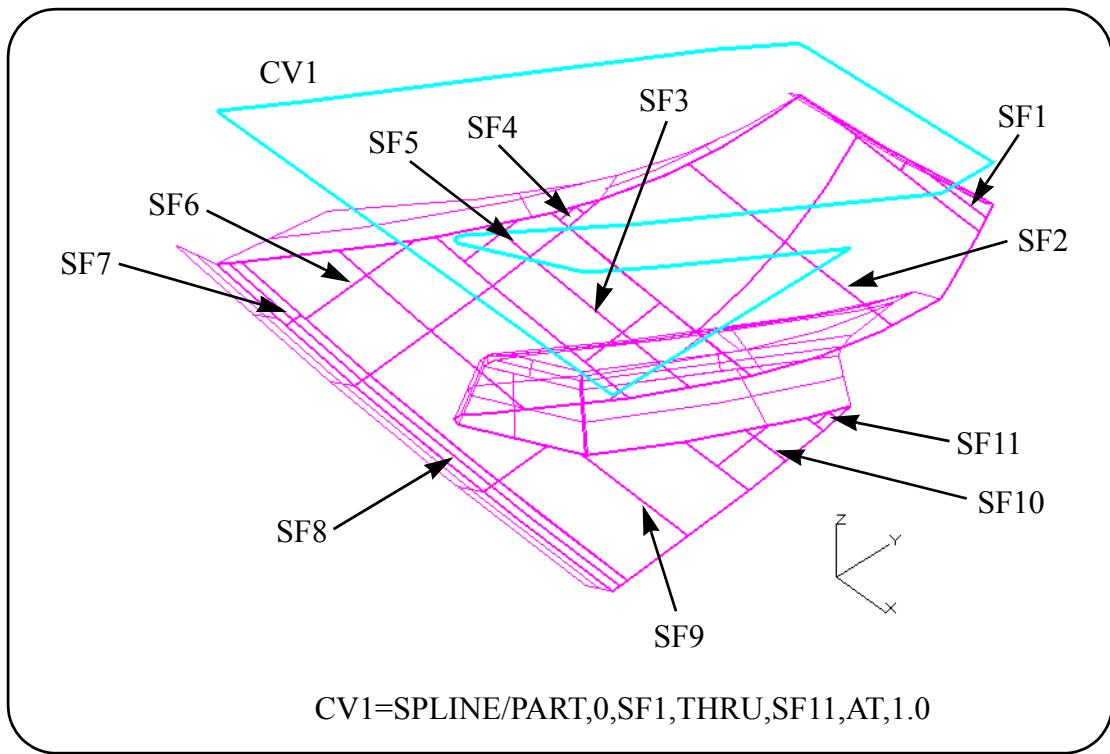
1, THRU, 5

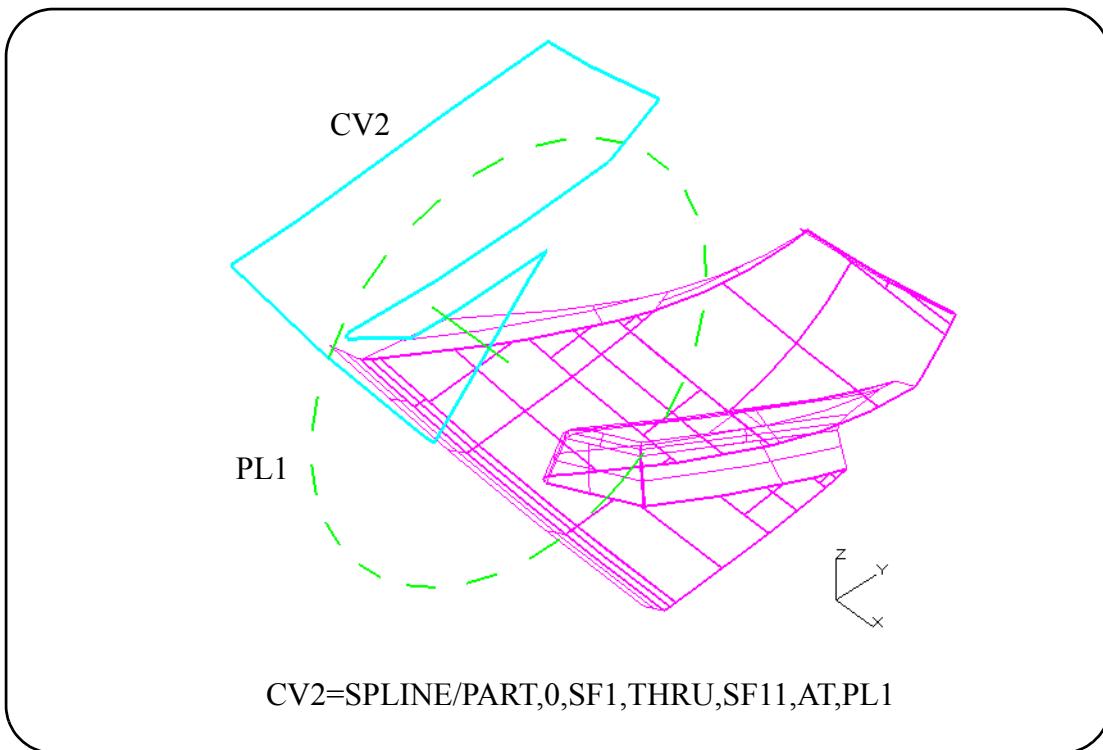
- A net surface - it is treated internally as a collection of individual surfaces.
- A layer number, e.g. LAYER=4.
- Any combination of above, separated by commas.

Note: Surfaces with same label are treated as one surface even though they may be specified more than once either explicitly or implied, e.g. if a net surface is specified all of the components are specified explicitly in the surface list.

zlev, plane or planar-surf Specified the Z-level value, a plane or a planar surface that the 2-D contour curve will be created. The plane or the planar-surface does not require normal to the Z-axis.

Icon Menu Sequence:





Calculation Method:

- There is no difference in specifying “CURVE” or “SPLINE”. Only spline will be created.
- The plane or planar surface does not require to be normal to the Z-axis of the current reference system. It can be any plane or planar surface with the normal vector to the plane or planar-surface at any direction.
- The group outside boundary is equal to the outside boundary of the shadow of the group of surface by shining a light along Z-axis of the current reference system or a vector normal to the specified plane or planar-surface at infinity. The 2-D closed contour curve is then created by projecting this shadow outside boundary to the specified Z-level of the current reference system along the Z-axis, or onto the specified plane or planar-surface along the vector normal to the specified plane or planar-surface.
- Inside boundary(ies) will not be considered.
- Outside boundary of trimmed surface will be utilized instead of the underlying parent surface.

Error Conditions:

- Gap(s) between surfaces within the group specified might create unexpected results.
- Surface with outside boundary projected as a wire entity on the XY-plane along the Z-axis of the current reference system will not be considered.
- Surface with outside boundary projected as a wire entity on the specified plane or planar-surface along the normal vector to the plane or planar-surface will not be considered.

3.2.13 A Spline Or S-Spline Extracted From A Revolved Surface

Command Syntax:

```
SPLINE/OUT,surface  
SSPLIN
```

Where:

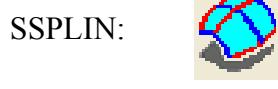
SPLINE Specifies that the original entity which was used to create the revolved surface will be created. The output entity may be of the type line, circle, spline, curve or composite curve.

SSPLIN Specifies that a composite Surface-spline or single Surface-spline will be created.

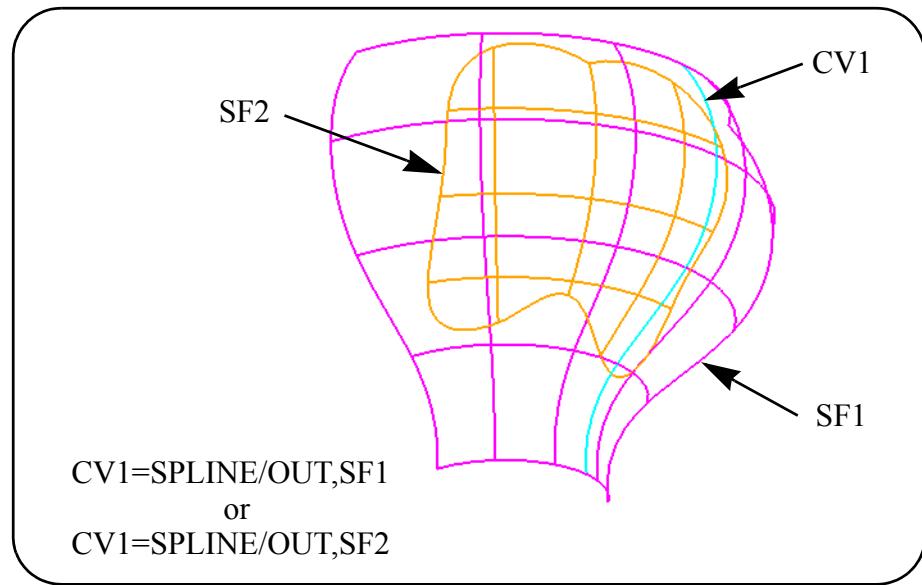
Icon Menu Sequence:



> *Curve Menu* >



> *Curve Menu* >



Calculation Method:

- The curve which was originally used to create the revolved surface will be extracted. The created curve does not necessary lie on the edge of the revolved surface.
- The base surface is considered if a trimmed surface is specified.
- A surface edge curve at U=0 will be created if the specified surface is of the primitive type Cone or Cylinder. Primitive type Sphere will not be considered.

3.2.14 A Spline Or S-Spline Extracted From A Trimmed Surface

Command Syntax:

```
SPLINE/OUT,surface, bn [, en1 [, CLW , en2 ] ]
SSPLIN                               CCLW
```

Where:

SPLINE Specifies that a composite curve or B-Spline will be created.

SSPLIN Specifies that a composite Surface-spline or single Surface-spline will be created.

bn Boundary curve to extract. 0 specifies that the outer boundary will be extracted. A number greater than 0 specifies that an inner boundary will be extracted. “bn” cannot be greater than the number of inner boundary curves contained in the trimmed surface.

en1 Edge to extract. 0 specifies that the entire boundary, specified by “bn”, will be extracted and a composite curve or composite Surface-spline will be created. A number greater than 0 specifies that a single sub-curve (edge) will be extracted and a B-Spline or a Surface-spline will be created. “en1” cannot be greater than the number of sub-curves contained in the boundary specified by “bn”.

en2 The edges from en1 to en2 will be extracted as a composite curve.

CLW Specifies the selected curves will be chosen in ascending order of the way the edges are defined in the trimmed surface.

CCLW Specifies the selected curves will be chosen in descending order of the way the edges are defined in the trimmed surface.

Icon Menu Sequence:

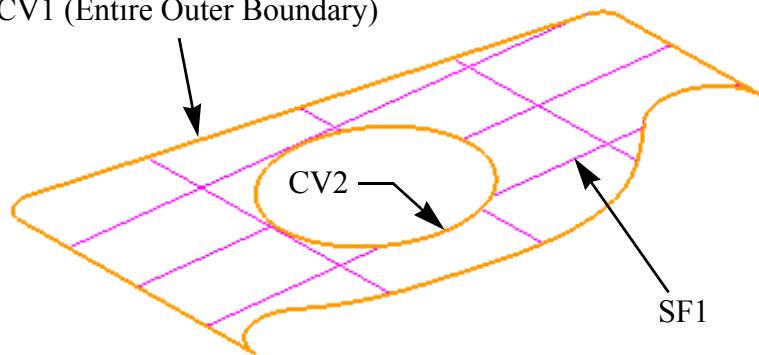




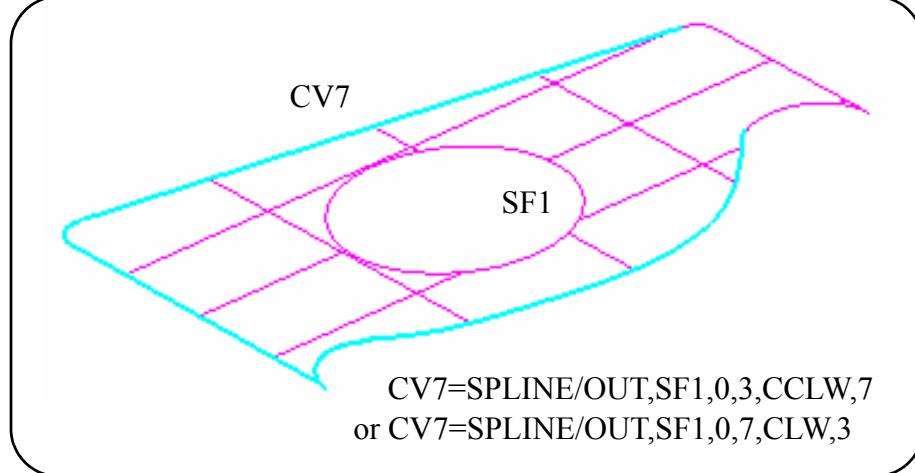
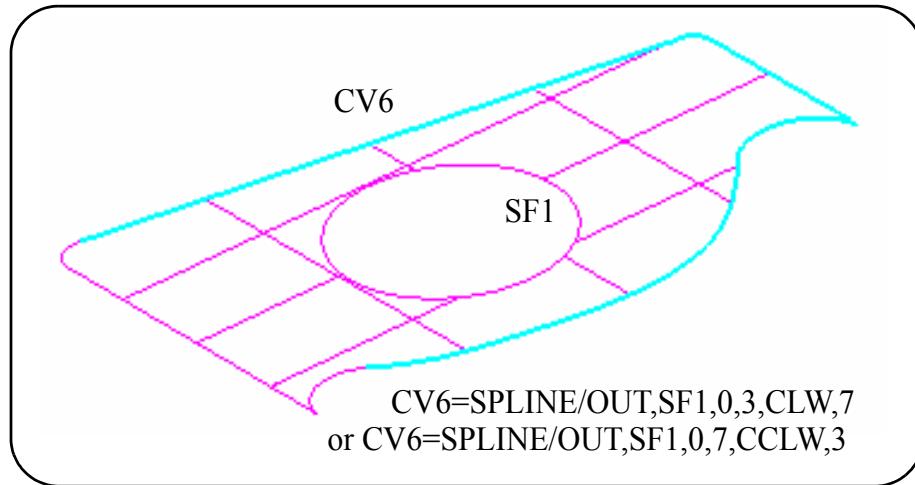
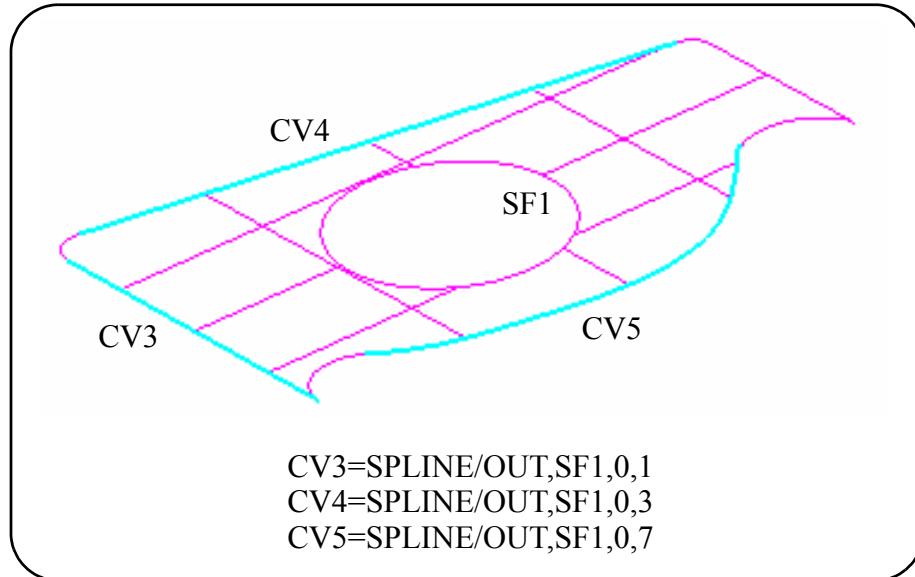
SSPLIN: > *Curve Menu* >



CV1 (Entire Outer Boundary)



CV1=SPLINE/OUT,SF1,0
CV2=SPLINE/OUT,SF1,1



3.2.15 A Spline Or S-spline Obtained By Projecting A Curve On A Surface At An Angle

Command Syntax:

```
SPLINE/PROJCT,curve ,ATANGL,start,[end,]           $
SSPLIN          spline
[surf1,]surface[[u,v]]
```

The inner square brackets must be specified as part of the syntax if the optional u,v parameters are used. “SPLINE” can be specified as “CURVE”, they are interchangeable here and only B-Splines will be created.

Where:

SPLINE	Specifies that a composite curve or B-Spline will be created.
SSPLIN	Specifies that a composite Surface-spline or single Surface-spline will be created.
curve	The curve to be projected onto the surface.
ATANGL	Specifies that the curve would be projected on the surface by rotating the normal vector to the interpolation surface at an angle around the forward sense of the curve.
start	Starting angle to use with interpolating surface.
end	Ending angle to use with interpolating surface.
surf1	Interpolation surface.
surface	The surface onto which the curve will be projected, no NET surface allowed.
[u,v]	Optional [u,v] values to tell NCL to “look” at the surface in area specified by the UV values. This is sometimes necessary on surfaces as there may be multiple projection possibilities.

Icon Menu Sequence:



> *Curve Menu* >



> *Curve Menu* >



Calculation Method:

- The specified curve will be broken into a series of points on the curve according to the current value of **TOLER**. Each of the individual points will be projected onto the specified surface. Finally, a B-Spline is generated from the projected points.
- An interpolation surface can be specified. If an interpolation surface is not specified, the surface onto which the curve is projected will be utilized as the interpolation surface.
- A vector normal to the interpolation surface is calculated at the given point on the curve.
- If the start angle and end angle are different, an *interpolated* angle will be calculated at the given point.
- The calculated normal vector rotates around the forward direction of the curve at the given point by the *interpolated or specified* angle utilizing the right hand rule.
- The given point is then projected onto the surface on which the curve is to be projected along this rotated vector.
- The extension of the specified surface or the interpolation surface will be used if required. The underlying surface will be used if the surface specified is of the type **TRIMMED**.

Error Conditions:

- A **NET** surface is specified as the surface of projection.
- A continuous section of the projection overlapped itself. Overlapping on different sections of the projection is allowed.

3.2.16 A Spline Or S-Spline Obtained By Projecting A Curve On A Surface

Command Syntax:

```
SPLINE/PROJECT,curve , [vector,]surface[[u,v]]      $
SSPLIN          spline

[,NOWRAP][,START     ][,near-point ]           $
WRAP            MIDDLE                pntvec
REVOLV          END
RADIAL          CENTER
AT,point
```

The inner square brackets must be specified as part of the syntax if the optional u,v parameters are used. “SPLINE” can be specified as “CURVE”, they are interchangeable here and only B-Splines will be created.

Where:

SPLINE	Specifies that a composite curve or B-Spline will be created.
SSPLIN	Specifies that a composite surface-spline or single surface-spline will be created.
curve	The curve to be projected onto the surface.
vector	Optional projection vector.
surface	The surface onto which the curve will be projected, no NET surface allowed.
[u,v]	Optional [u,v] values to tell NCL to “look” at the surface in area specified by the UV values. This is sometimes necessary on surfaces as there may be multiple projection possibilities.
NOWRAP	The specified curve will be projected onto the specified surface without any wrapping. This will generate a distorted curve.
WRAP	The specified curve will be projected onto the specified surface with wrapping. The size and the shape of the generated curve will be approximated maintained.

REVOLV This only applies when the specified surface is a cone, a cylinder, or a surface of revolution. The size of the generated curve will be approximately maintained, but not the shape.

RADIAL This only applies when the specified surface is a cone, a surface of revolution or a cylinder. The shape of the generated curve will be approximately maintained, but not the size.

START Specifies the base of projection is at the beginning of the curve.

MIDDLE Specifies the base of projection is at half way along the curve.

END Specifies the base of projection is at the end of the curve.

CENTER Specifies the base of projection is at the center of a bounding box formed by the curve.

AT Specifies the user explicitly specified a point as the base of projection.

point The user selected base point.

near-point or near-pntvec

Optional parameter (point or point-vector) given to clarify which of the several possible projections is desired.

Icon Menu Sequence:



> **Curve Menu** >

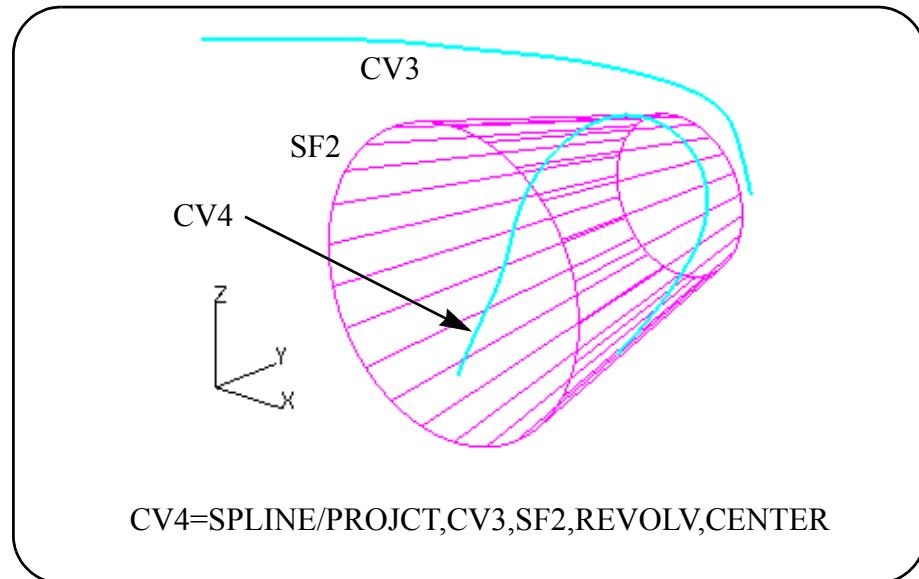
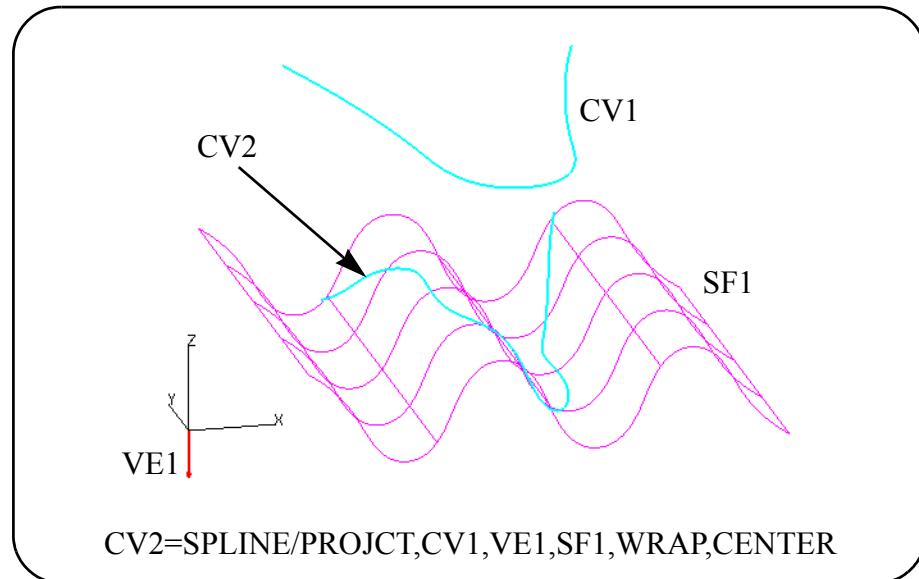


> **Curve Menu** >



>





Calculation Method:

- The specified curve will be broken into a series of points on the curve according to the current value of **TOLER**. Each of the individual points will be projected onto the specified surface according to the “Method” specified. Finally, a B-Spline is generated from the projected points.

- If the optional parameter “vector” is not specified, the projection will be along a vector normal to the specified surface.
- If the optional parameter “NOWRAP”, “WRAP”, “REVOLV” or “RADIAL” is not specified, the projection will default to “NOWRAP”.
- If “NOWRAP” is specified, the internal generated points on the specified curve will be projected individually onto the surface. This will create a distorted curve. The shape or the size of the curve will not be maintained.
- If “WRAP” is specified, the user can specify an optional “ATTACH” method, either START, MIDDLE, END, CENTER, or “AT,point”. Distances and angles from the attach point are maintained. This will approximately maintain the shape and size based at the attach point.
- If the optional parameter “START”, “MIDDLE”, “END”, “CENTER” or “AT,point” is not specified, the projection will default to “START”.
- “REVOLV” or “RADIAL” is only applicable for projecting onto cones, cylinders, or surface of revolution. Distances between the projected points are not preserved for “RADIAL”. This will approximately maintain the shape, but not the size. Distances between the projected points are preserved for “REVOLV”. This will approximately maintains the size, but not the shape. There will be no difference for “REVOLV” and “RADIAL” if the surface is of the type “cylinder”. See [REVOLV Projection Calculation Method](#) and [RADIAL Projection Calculation Method](#) for detail.
- The extension of the specified surface will be used if required. The underlying surface will be used if the surface specified is of the type [TRIMMED](#).

Error Conditions:

- If “REVOLV” or “RADIAL” is specified and the surface of projection is not a cone, a cylinder or a surface of revolution.
- The projection vector does not intersect the surface of projection when the vector is base at the projection point.
- The optional near-point or near-pntvec is not closed to the desired surface position.
- A [NET](#) surface is specified as the surface of projection.
- A continuous section of the projection overlapped itself. Overlapping on different sections of the projection is allowed.

RADIAL Projection Calculation Method:

1. The attached point “PT1” specified by “START, MIDDLE, END, CENTER, or At,point” projects onto the surface “SF1” according to the method specified, i.e. along a specific vector, or normal to the surface “SF1”.
2. This projected attached point “PT2” becomes the reference point on the surface “SF1”.
3. A reference circle normal to the revolving axis, with its center along the revolving axis and its circumference passing through the reference point “PT2” is created internally. The radius of this reference circle is “R1”.
4. A reference plane “PL1” which contains the revolving axis and the attach point projection vector is created internally.
5. The curve “CV1” which is going to be projected is broken into a series of points “PT(n)” internally.
6. The normal distance “L1” of a point “PT(n)” on the curve “CV1” to the reference plane “PL1” is calculated.
7. The point “PT(n)” is projected normal to the reference plane “PL1”. The distance “L2” between the point “PT(n)” and the attached point “PT1” is calculated.
8. A point “PT3” is created internally at a distance “L2” along the slope of the revolving surface “SF1” on the reference plane “PL1” in the same side as the point “PT(n)”.
9. A circle “CI1” normal to the revolving axis, with its center along the revolving axis and its circumference passing through the point “PT3” is created internally. The radius of this circle is “R2”.
10. A point “PT4” on the circle “CI1” along an arc distance away from the reference plane “PL1” in the same side of the point “PT(n)” on the curve “CV1” is created internally. The location of the point “PT4” is where the point “PT(n)” on the curve projected to the revolved surface “SF1”.
11. This arc distance is calculated according to the following equation.

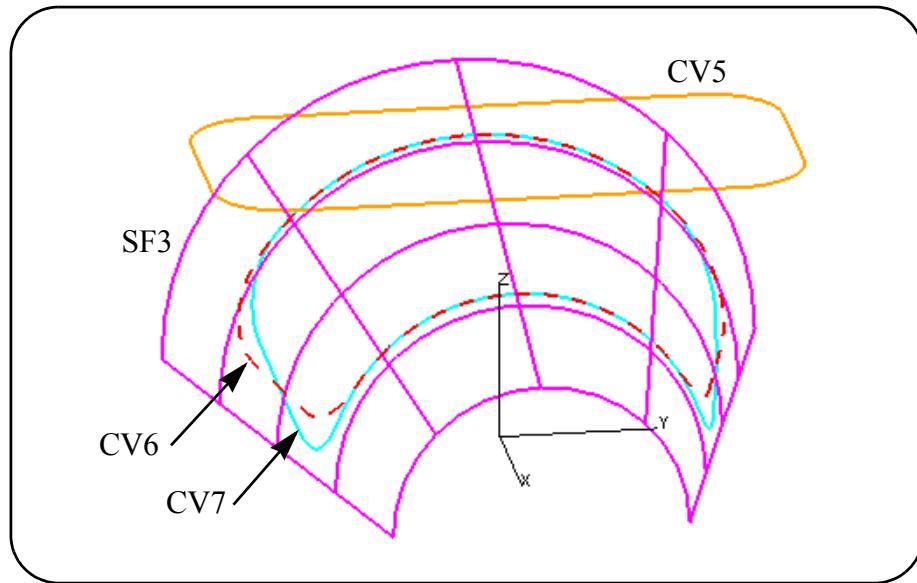
$$\text{arc distance} = L1 * R2 / R1$$

12. Repeat Step 6 through Step 11 for all the points on the curve “CV1” until done.
13. All the projected points created internally are joined together to form the resultant projected curve on the surface.

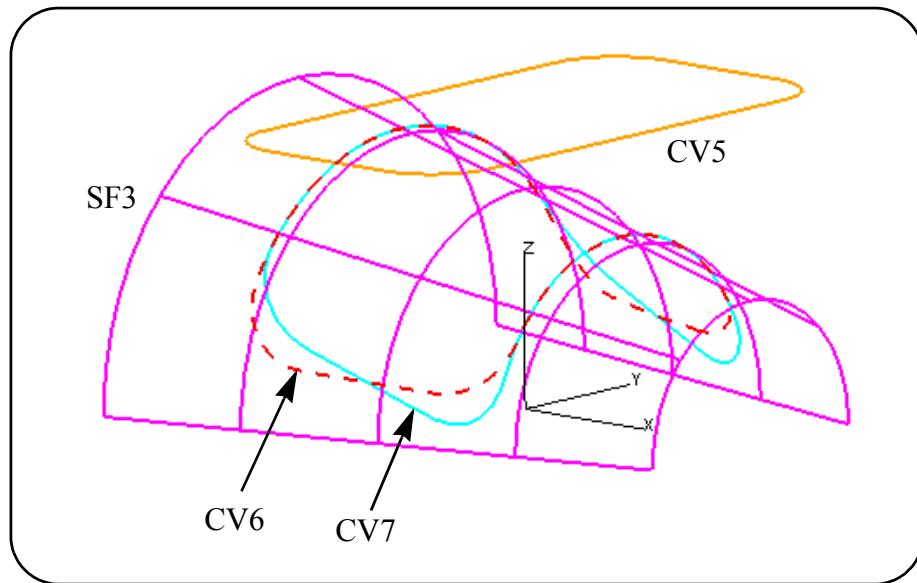
REVOLV Projection Calculation Method:

1. The attached point “PT1” specified by “START, MIDDLE, END, CENTER, or At,point” projects onto the surface “SF1” according to the method specified, i.e. along a specific vector, or normal to the surface “SF1”.
2. This projected attached point “PT2” becomes the reference point on the surface “SF1”.
3. A reference plane “PL1” which contains the revolving axis and the attach point projection vector is created internally.
4. The curve “CV1” which is going to be projected is broken into a series of points “PT(n)” internally.
5. The normal distance “L1” of a point “PT(n)” on the curve “CV1” to the reference plane “PL1” is calculated.
6. The point “PT(n)” is projected normal to the reference plane “PL1”. The distance “L2” between the point “PT(n)” and the attached point “PT1” is calculated.
7. A point “PT2” is created internally at a distance “L2” along the slope of the revolving surface “SF1” on the reference plane “PL1” in the same side as the point “PT(n)”.
8. A circle “CI1” normal to the revolving axis, with its center along the revolving axis and its circumference passing through the point “PT2” is created internally. The radius of this circle is “R2”.
9. A point “PT3” on the circle “CI1” along an arc distance away from the reference plane “PL1” in the same side of the point “PT(n)” on the curve “CV1” is created internally. The location of the point “PT3” is where the point “PT(n)” on the curve projected to the revolved surface “SF1”.
10. Repeat Step 5 through Step 9 for all the points on the curve “CV1” until done.
11. All the projected points created internally are joined together to form the resultant projected curve on the surface.

The figures on next page show the difference of projecting the same curve on a cone surface by RADIAL and REVOLV.



CV6=SPLINE/PROJCT,CV5,SF3,RADIAL,CENTER
CV7=SPLINE/PROJCT,CV5,SF3,REVOLV,CENTER



3.2.17 Isoparametric Surface-Spline On A Surface

Command Syntax:

```
SSPLIN/surface[ [,n],roff]
```

Where:

surface surface identifier (Surface types are **NCL** surfaces and Rational B-Spline surfaces. Trimmed surface is not allowed.)

n edge type:

1 = v curve at u = 0 (default condition)

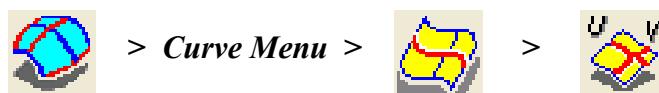
2 = v curve at u = 1

3 = u curve at v = 0

4 = u curve at v = 1

roff offset parameter (u or v) from the edge of the surface (defined by n) specified in the range of 0 - 1, where 0 applies to n edge and 1 applies to the opposite edge. If not specified, it will be defaulted to “0”.

Icon Menu Sequence:



NOTE: Surface-Splines can be used as a **Part Surface**. A pseudo ruled surface through the spline and normal to the **Drive Surface** will actually be driven when a Surface-Spline is used as the Part Surface. Use the ‘**PSIS/ssplin**’ command to specify a Surface-Spline as the part surface.

The following restrictions apply to SSPLIN and surfaces containing SSPLINS:

- A surface-spline cannot be copied, moved, or trimmed.
- A surface-spline cannot be used as a defining curve of a surface.

- A surface-spline is not supported in **GET** and **PUT** commands with a secondary unibase.
- A surface containing SSPLINs will be copied without the surface-splines.
- A surface containing SSPLINs will be transferred to or from a secondary unibase without the surface-splines.

3.2.18 Surface-Spline On A Surface Intersecting The Surface With Another Surface Or Plane With An Optional Near Point/Point-Vector

Command Syntax:

```
SSPLIN/INTOF, surface-1,surface-2[,near-point ]
                           plane           pntvec
```

Where:

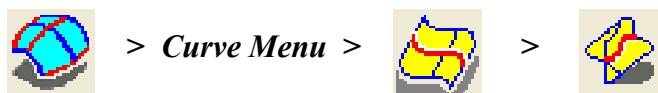
surface-1 identifier of surface for which the created spline will belong.
(Surface types are **NCL** surfaces and Rational B-Spline surfaces. Trimmed surface is not allowed.)

surface-2 identifier of surface or plane to intersect with surface-1.
(Surface types are **NCL** surfaces and Rational B-Spline surfaces. Trimmed surface is not allowed.)

near-point or pntvec

optional nearest point/point-vector to establish a spline when more than one spline can be created as the intersection of the two entities. The point-vector origin will be used if a point-vector is specified.

Icon Menu Sequence:



NOTE: Surface-Splines can be used as a **Part Surface**. A pseudo ruled surface through the spline and normal to the **Drive Surface** will actually be driven when a Surface-Spline is used as the Part Surface. Use the '**PSIS/ssplin**' command to specify a Surface-Spline as the part surface.

The following restrictions apply to SSPLINs and surfaces containing SSPLINs:

- A surface-spline cannot be copied, moved, or trimmed.
- A surface-spline cannot be used as a defining curve of a surface.

- A surface-spline is not supported in **GET** and **PUT** commands with a secondary unibase.
- A surface containing SSPLINs will be copied without the surface-splines.
- A surface containing SSPLINs will be transferred to or from a secondary unibase without the surface-splines.

3.2.19 A Surface-Spline As A Composite Collections Of Surface-Splines

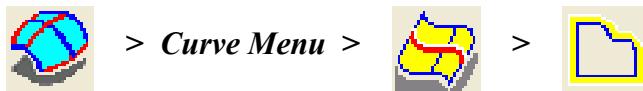
Command Syntax:

```
SSPLIN/COMPOS, ssplin[ [...] , ssplin]
```

Where:

ssplin Surface spline identifies which make up a composite curve. A surface spline composite curve can only be made up of surface splines. Regular 3-D curves, lines, and circles cannot be part of a surface spline composite curve.

Icon Menu Sequence:



NOTE: Surface-Splines can be used as a [Part Surface](#). A pseudo ruled surface through the spline and normal to the [Drive Surface](#) will actually be driven when a Surface-Spline is used as the Part Surface. Use the '[PSIS/ssplin](#)' command to specify a Surface-Spline as the part surface. When a composite Surface-Spline is used as the Part Surface, then **NCL** will treat each sub-components of the composite as a separate surface, similar to driving a [Net Surface](#).

The following restrictions apply to SSPLIN and surfaces containing SSPLINS:

- A surface-spline cannot be copied, moved, or trimmed.
- A surface-spline cannot be used as a defining curve of a surface.
- A surface-spline is not supported in [GET](#) and [PUT](#) commands with a secondary unibase.
- A surface containing SSPLINS will be copied without the surface-splines.
- A surface containing SSPLINS will be transferred to or from a secondary unibase without the surface-splines.

3.2.20 A Surface-Spline As An Offset Of A Surface-Spline

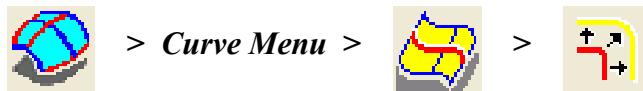
Command Syntax:

```
SSPLIN/OFFSET,ssplin,XSMALL,distance
          XLARGE
          YSMALL
          YLARGE
          ZSMALL
          ZLARGE
```

Where:

distance Offset distance measured along the base surface of the input surface-spline and not in 3-D space.

Icon Menu Sequence:



NOTE: Surface-Splines can be used as a [Part Surface](#). A pseudo ruled surface through the spline and normal to the [Drive Surface](#) will actually be driven when a Surface-Spline is used as the Part Surface. Use the '[PSIS/ssplin](#)' command to specify a Surface-Spline as the part surface.

The following restrictions apply to SSPLIN and surfaces containing SSPLINS:

- A surface-spline cannot be copied, moved, or trimmed.
- A surface-spline cannot be used as a defining curve of a surface.
- A surface-spline is not supported in [GET](#) and [PUT](#) commands with a secondary unibase.
- A surface containing SSPLINS will be copied without the surface-splines.
- A surface containing SSPLINS will be transferred to or from a secondary unibase without the surface-splines.

Error condition:

- The offset distance causes the generated sspline to go out of bounds of the base surface.

LINES

The following list gives an abbreviated notation of all the valid LINE definition formats. See the individual sections for a complete explanation of each format.

1. `LINE/x1,y1,x2,y2`
or `LINE/x1,y1,z1, x2,y2,z2`
2. `LINE/point ,point`
`point-vector point-vector`
3. `LINE/point-vector`
4. `LINE/point , PARREL,line`
`point-vector point-vector`
5. `LINE/point , PERPTO,line`
`point-vector point-vector`
6. `LINE/PARREL,line,direction,distance`
7. `LINE/FWD`
8. `LINE/point ,RIGHT,TANTO,circle`
`point-vector LEFT`
9. `LINE/RIGHT,TANTO,circle,RIGHT,TANTO,circle`
`LEFT LEFT`
10. `LINE/axis[,distance]`
11. `LINE/point ,ATANGL,angle[,XAXIS]`
`point-vector YAXIS`
`line`
`point-vector`
12. `LINE/point ,PERPTO,curve,near-point`
`point-vector TANTO point-vector`
13. `LINE/INTOF,plane,plane`
14. `LINE/direction,circle,ATANGL,angle,[XAXIS]`
`YAXIS`
`line`
`point-vector`

15. **LINE**/line [, Dx[, Dy[, Dz]]]
point-vector

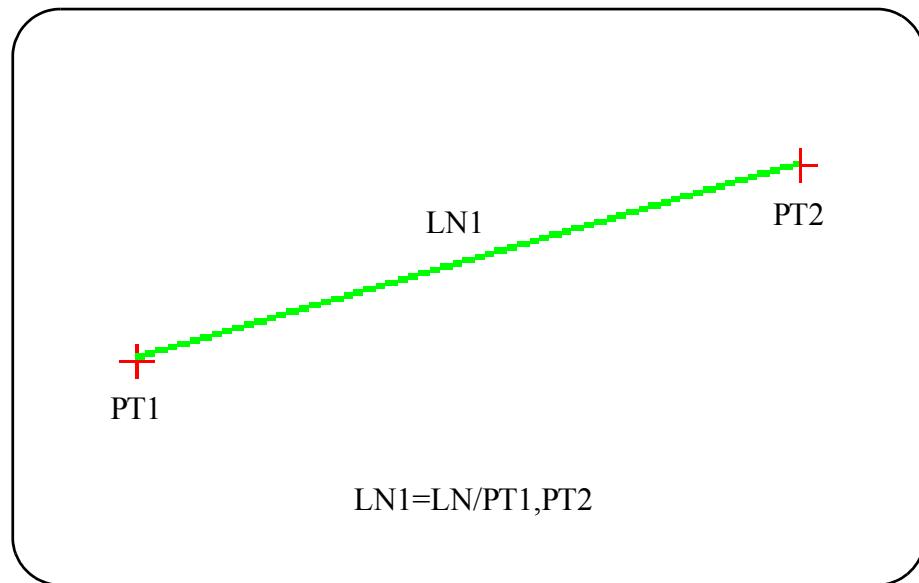
16. **LINE**/plane

Example Line Expressions:

```
LN/PTO(X),PTO(X+1)
LN/PV1
LN/PV1,PV2
LN/PT10,PARREL,LX
LN/PV1,PARREL,LN2
LN/PT10,PERPTO,LN4
LN/PV1,PERPTO,LN2
LN/PT1,PE,PV2
LINE/PARREL,LN1,XLARGE,2.3
LINE/FWD
LINE/PT1,RIGHT,TANTO,CI1
LINE/PV1,LEFT,TANTO,CI2
LINE/RIGHT,TANTO,CI1,LEFT,TANTO,CI2
LN/XAXIS
LINE/PT1,ATANGL,45,LN1
LN/PV1,AA,22,LN2
LINE/PT1,AA,30,PV2
LN/PT1,PA,PV2
LN/PT1,ATANGL,30
LN/2,3.5,1,5,2.25,2
LN/0,0,3,4
LN/PT1,TANTO,CV1,PT2
LN/PV1,TT,CV2,PV3
LN/PV1,PE,CV2,PV3
LINE/INTOF,PL3,PLZ8
LN/XL,CI22,AA,2.5,LN1
LINE/XS,CI1,AA,20,PV2
LN/LN1
LN/LN1,1,0,1
LN/PL1
LINE/PV1,x,y,z
```

3.3 LINE Expressions

A LINE is a geometric entity possessing three attributes: location, direction and length.



A line may be defined as bounded or unbounded. A bounded line defined by two points will only be displayed between those two points. However, for geometric definitions and tool motion, the bounded line extends to infinity. The bounded line has three attributes: location, direction and length. The "start point" of the line specifies its location in 3D space. The direction of the line is defined as being from the "start point" to the "end point." The length of the line is the distance between the "start point" and the "end point."

An unbounded line will be displayed as a line segment one (1) inch in length. For example: a line defined parallel to the x-axis will be displayed as a 1 inch line along the x-axis. However, this line extends to infinity.

Canonical Form:

LINE/X, Y, Z, DELTA-X, DELTA-Y, DELTA-Z

Special Conditions:

- **GREEN** is the **NCL** System DEFAULT color for CAM LINES.

3.3.1 A Line As Its Coordinates

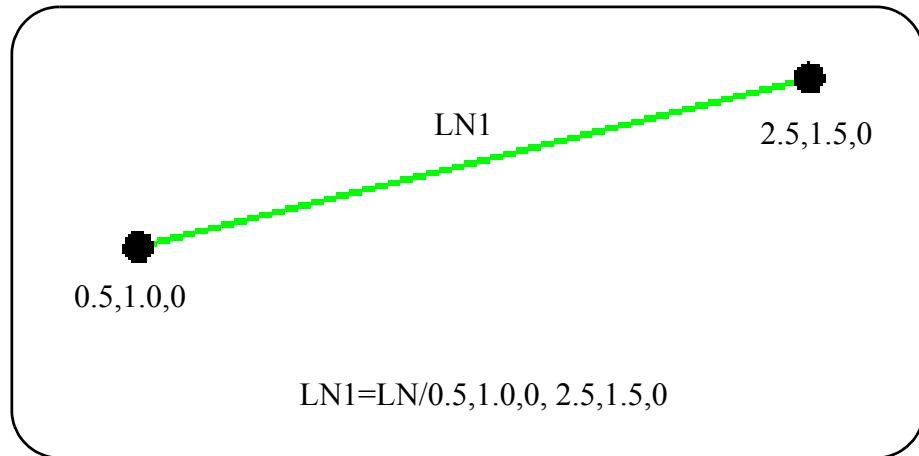
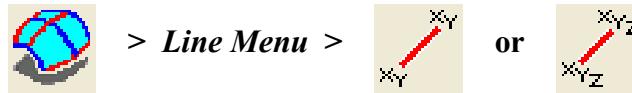
Command Syntax:

`LINE/x1,y1,x2,y2`

or

`LINE/x1,y1,z1, x2,y2,z2`

Icon Menu Sequence:



Calculation Method:

- The "implied" points will be calculated relative to the "current" REFERENCE SYStem. If the optional z-value is omitted, the z-value will be Z=0 of the "current" REFERENCE SYStem.
- The length of the line will be from the first "implied" point to the second "implied" point.
- The direction of the line will be from the first "implied" point to the second "implied" point.

Error Condition:

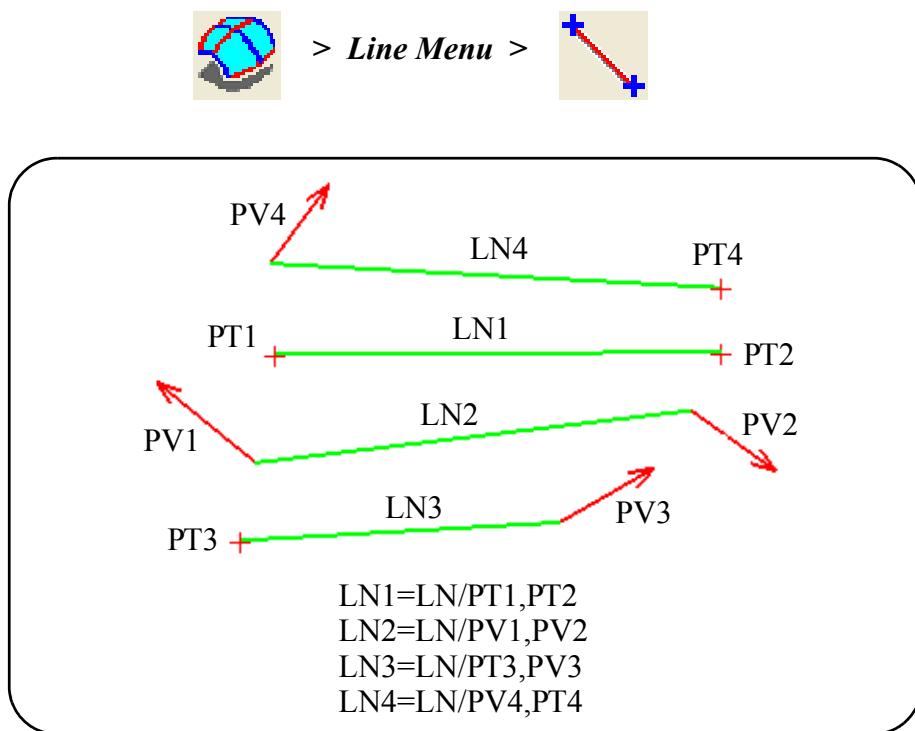
- The line may not be zero length.

3.3.2 A Line Between A Point/Point-Vector And A Point/Point-Vector

Command Syntax:

```
LINE/point      ,point
    point-vector point-vector
```

Icon Menu Sequence:



Calculation Method:

- The line is defined between the two (2) referenced points/point-vectors.
- The length of the line is calculated from the first referenced point or the origin of the first point-vector to the second referenced point or the origin of the second point-vector.
- The direction of the line is calculated from the first referenced point/point-vector to the second referenced point/point-vector.

Error Condition:

- The line may not be zero length (the points/point-vectors may not be coincident).

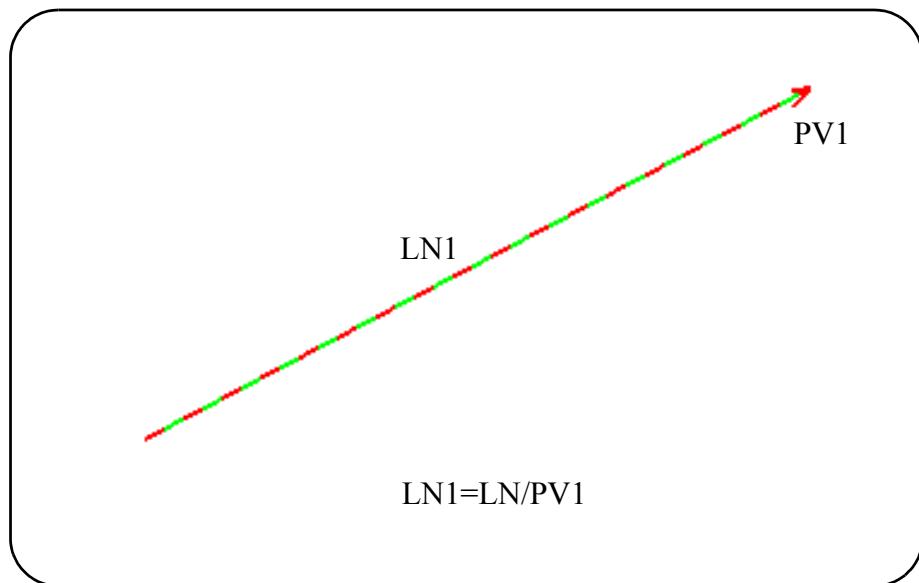
3.3.3 A Line From The Canonical Form Of A Point-Vector

Command Syntax:

LINE/point-vector

Icon Menu Sequence:

No corresponding Icon Menu.



Calculation Method:

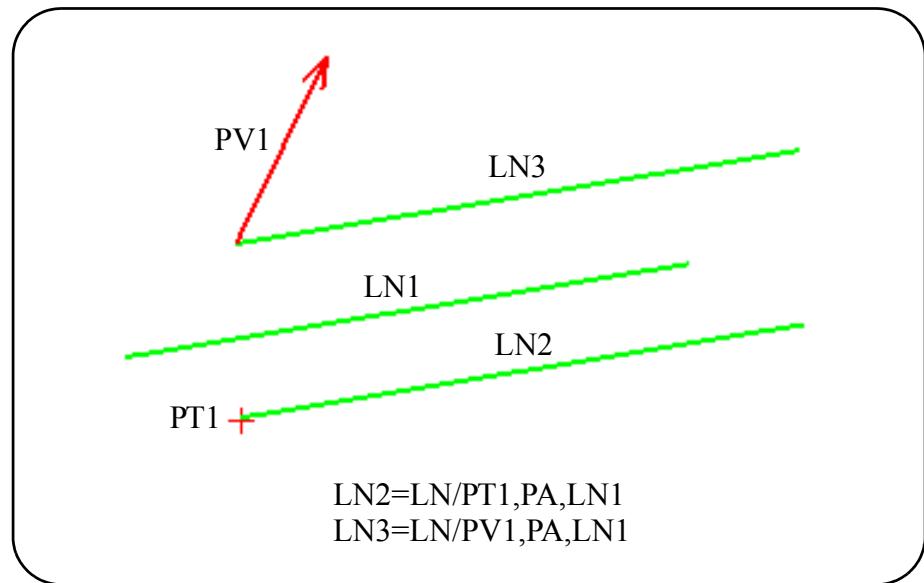
- The line is defined from the canonical form of the point-vector.
- The line originates at the origin of the point-vector, ends at the end of the vector and is the same length as the vector.

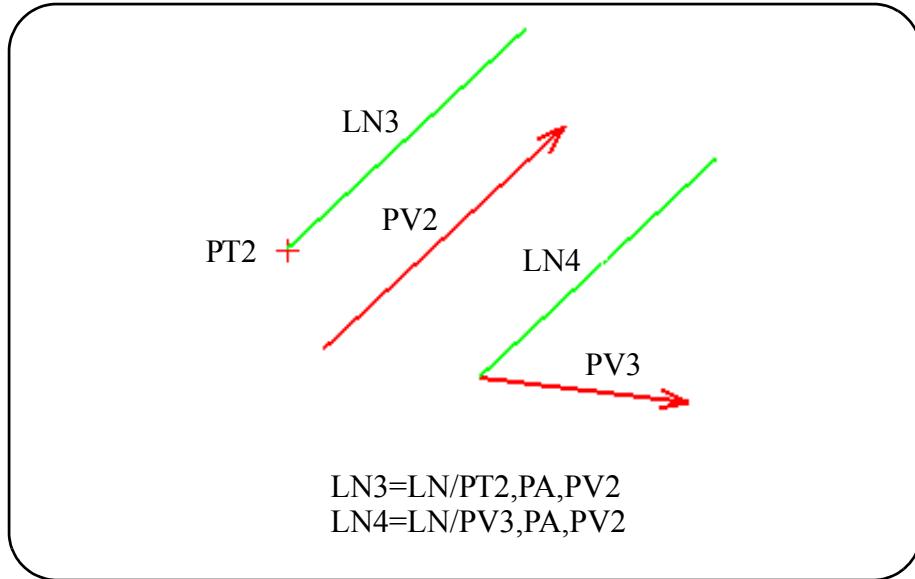
3.3.4 A Line From A Point/Point-Vector And Parallel To A Line/Point-Vector

Command Syntax:

```
LINE/point      , PARALLEL, line  
    point-vector   point-vector
```

Icon Menu Sequence:





Calculation Method:

- This format is a full three-dimensional definition that essentially makes a copy of the referenced line/point-vector with the referenced point or the origin of the first point-vector as the start point of the new line.
- The length of the line will be the same as the referenced line/point-vector.
- The direction of the line will be the same as the referenced line/point-vector.

Error Condition:

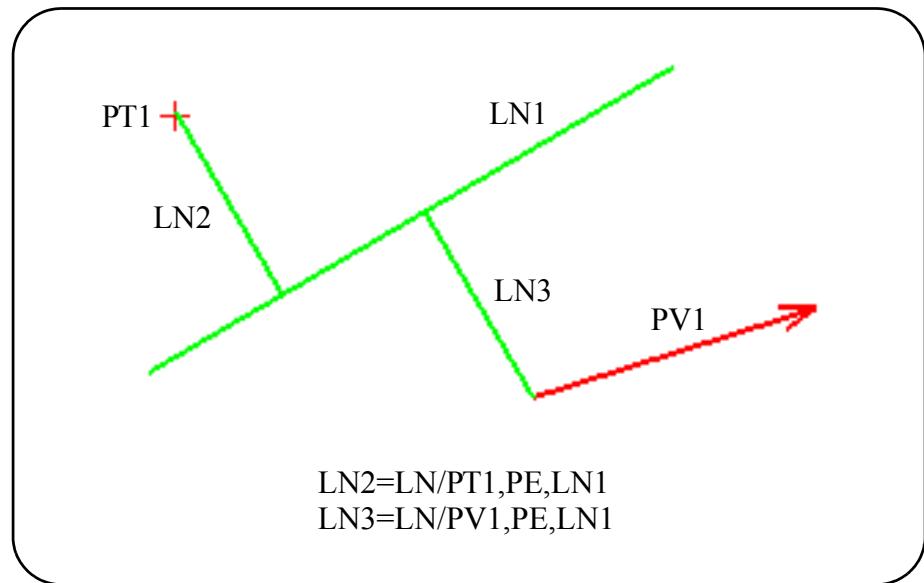
- The point/point-vector may lie on the referenced line.

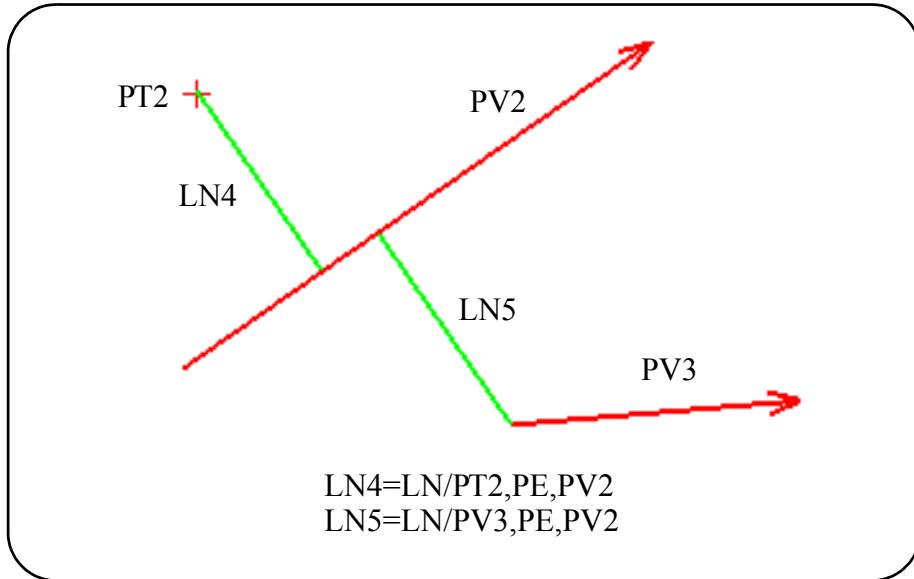
3.3.5 A Line From A Point/Point-Vector And Perpendicular To A Line/Point-Vector

Command Syntax:

```
LINE/point      , PERPTO, line  
    point-vector   point-vector
```

Icon Menu Sequence:





Calculation Method:

- The referenced point or the origin point of the first vector will be the start point of the new line.
- The line will lie in a Z-plane containing the referenced point.
- The length of the line will be calculated based on the length of the referenced line/point-vector using only the delta-x and delta-y components of the referenced line/point-vector (the delta-z component is not used). This means that the length of the new line is the "projected length" of the referenced line/point-vector as a result of being projected onto the XY-plane of the "current" REFERENCE SYStem.
- The direction of the line will be calculated as follows:
- The referenced line is projected onto the XY-plane containing the referenced point/point-vector. The defined line lies on that XY-plane perpendicular to the projected line and goes through the referenced point or the origin of the first referenced point-vector. The direction is from the referenced point or the origin of the first reference point-vector towards the point of intersection of the defined line and the projected line.

Error Conditions:

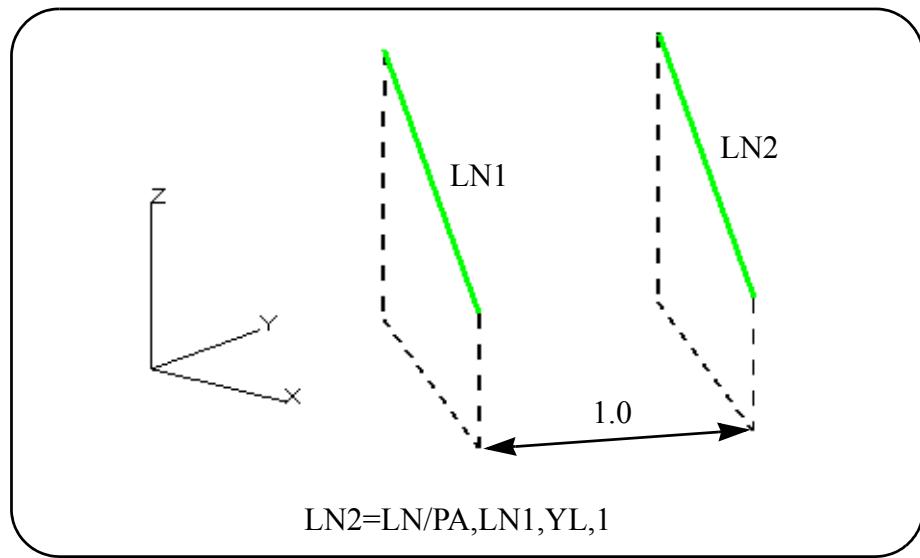
- The line may not be zero length when projected.
- The point/point-vector may lie on the referenced line/point-vector.

3.3.6 A Line Parallel To A Line At A Distance

Command Syntax:

```
XLARGE
XSMALL
LINE/PARREL, line, YLARGE, distance
YSMALL
```

Icon Menu Sequence:



NOTE: The modifiers **XLARGE**, **XSMALL**, **YLARGE** and **YSMALL** designate the position of the line being defined relative to the position of the referenced line.

Calculation Method:

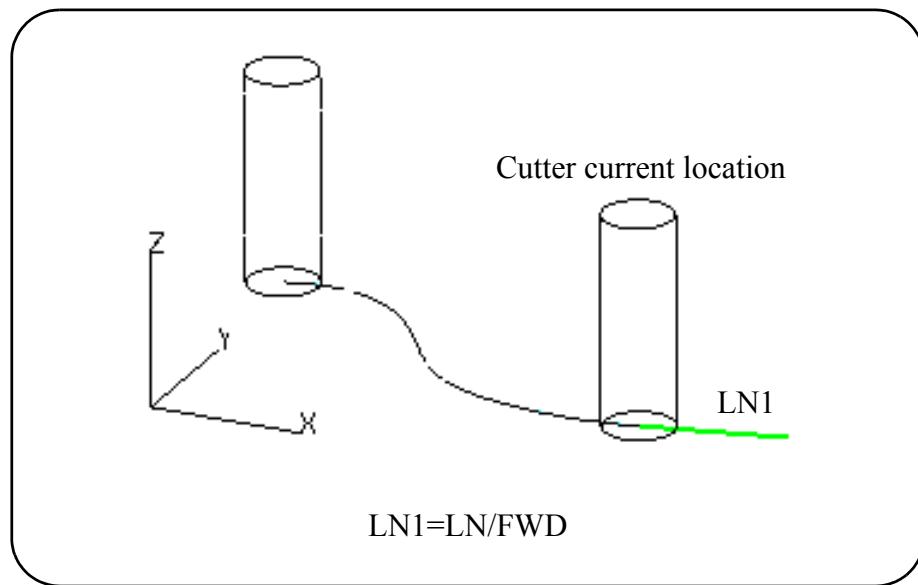
- The format is a full three-dimensional definition that essentially makes a copy of the referenced line with the new line located in a position which is parallel to the referenced line at the specified distance.
- The length of the line will be the same as the referenced line.
- The direction of the line will be the same as the referenced line.

3.3.7 A Line Defined By The Forward Sense Of The Current Tool Position

Command Syntax:

LINE/FWD

Icon Menu Sequence:



This format is used to define geometric entities that are difficult if not impossible to define using conventional techniques. This example illustrates the same concepts as those explained in the point definition format ([POINT/ TE](#)) with the exception that a line (LINE/ FWD) rather than a point is defined once the cutter has been "driven" to the desired position.

Calculation Method:

- The length of the line will be one (1) inch regardless of the system input units.

- The direction of the line will be from the current cutter position pointing along the forward sense of the current tool motion.

Error Conditions

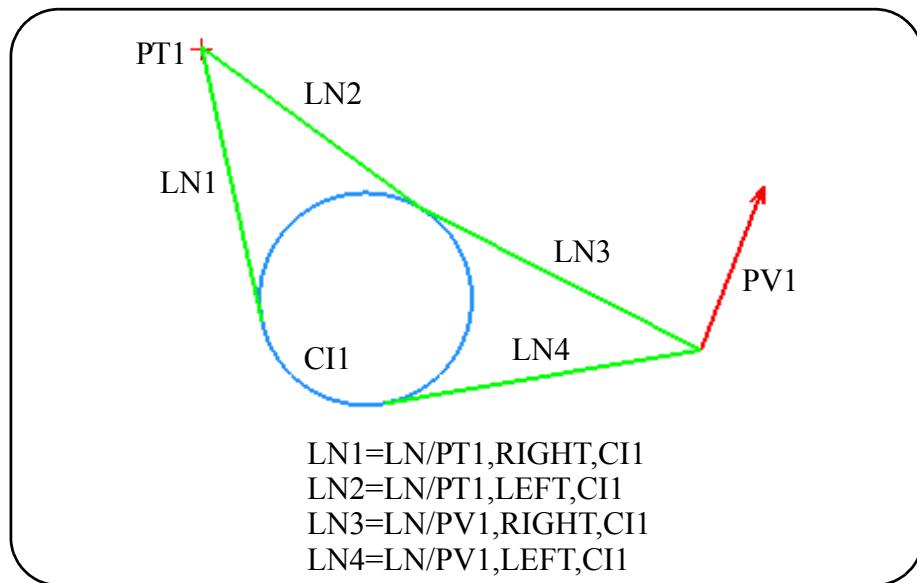
- An error will be issued if no motion has been generated prior to the use of this statement.

3.3.8 A Line Through A Point/Point-Vector And Tangent To A Circle

Command Syntax:

```
LINE/point      , LEFT , TANTO,circle
    point-vector RIGHT
```

Icon Menu Sequence:



NOTE: The modifiers **RIGHT** and **LEFT** are applied looking from the point/point-vector toward the circle.

Calculation Method:

- The circle is projected onto the Z-plane containing the referenced point/point-vector which is parallel to the xy-plane of the "current" REference SYstem, followed by;
- The calculation of the line which is through the referenced point or the origin of the referenced point-vector and tangent to the "projected circle."

- The length of the line will be calculated from the referenced point or the origin of the referenced point-vector to the tangency point of the line and the "projected circle."
- The direction of the line will be from the referenced point or the origin of the referenced point-vector to the tangency point of the line and the "projected circle."

Error Conditions:

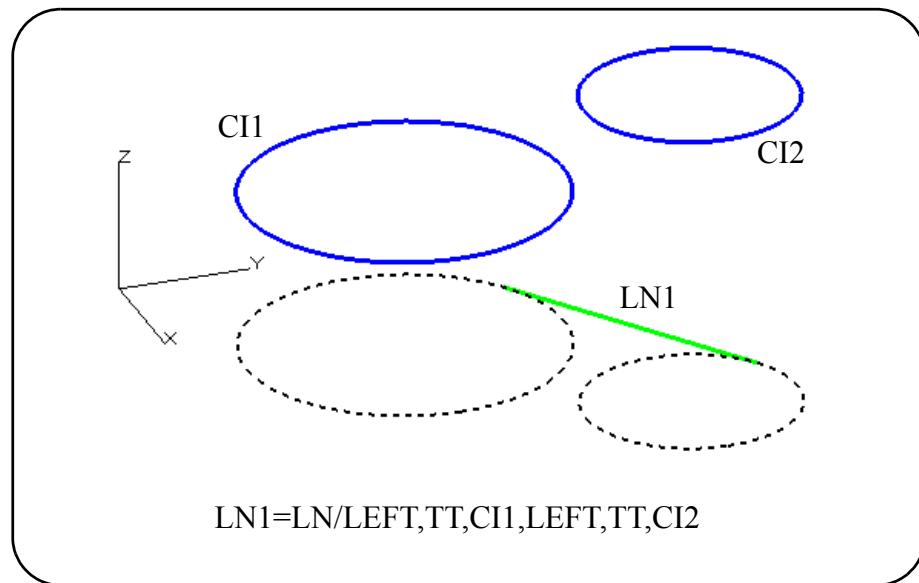
- The circle may not be "tipped" relative to the xy-plane of the "current" REReference SYStem.
- The calculated line may not be zero length (the point or the point-vector origin may not lie on the circle when projected).
- The point or the origin of the point-vector may not lie inside the projected circle.

3.3.9 A Line Tangent To Two Circles

Command Syntax:

```
LINE/LEFT ,TANTO,circle,LEFT ,TANTO,circle
      RIGHT           RIGHT
```

Icon Menu Sequence:



NOTE: The modifiers **RIGHT** and **LEFT** are applied looking from the first referenced circle to the second referenced circle.

Calculation Method:

- The second circle is projected onto the plane of the first circle, where both circles must be parallel to the XY-plane of the "current" **REFerence SYStem**, followed by;
- The calculation of the line which is tangent to the "projected circles" as specified by the modifiers **RIGHT** or **LEFT**.

- The length of the line will be calculated from the first line/referenced circle tangency point to the second line/"projected circle" tangency point.
- The direction of the line will be calculated from the first line/referenced circle tangency point to the second line/"projected circle" tangency point.

Error Conditions:

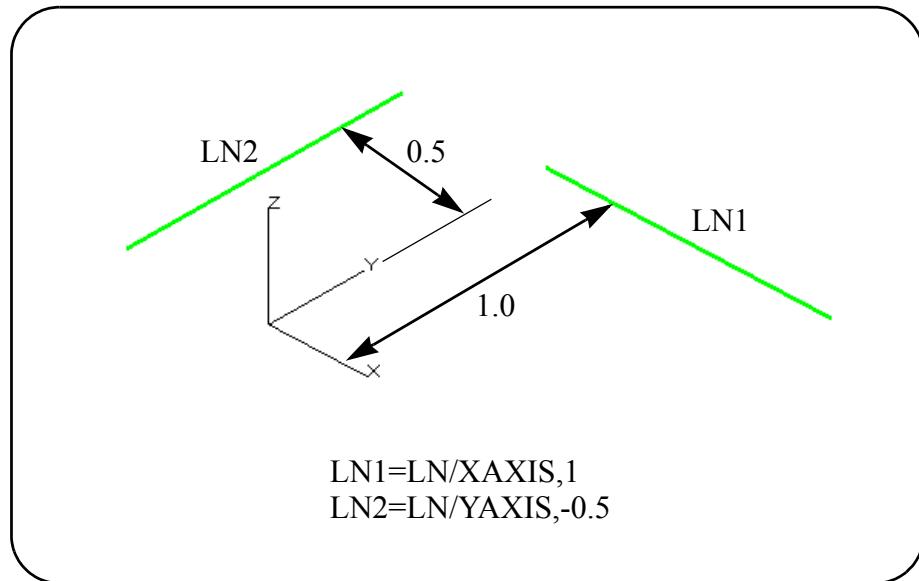
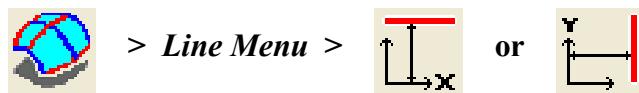
- The circles may not be "tipped" relative to the XY-plane of the "current" REference SYstem.
- The calculated line may not be zero length.

3.3.10 A Line Parallel To the X Or Y Axis At A Distance

Command Syntax:

```
LINE/XAXIS [,distance]
YAXIS
```

Icon Menu Sequence:



NOTE: The optional distance may be a positive (+) or negative (-) value. The sign of the number specified is used by **NCL** to designate which side of the specified axis the line will be positioned. A positive number specifies that the line be positioned on the positive (**XLarge** or **YLarge**) side of the X-axis or Y-axis. A negative number specifies that the line be positioned on the negative (**XSmall** or **YSmall**) side of the X-axis or Y-axis. If the optional distance is omitted, then the line will lie on the X or Y-axis.

Calculation Method:

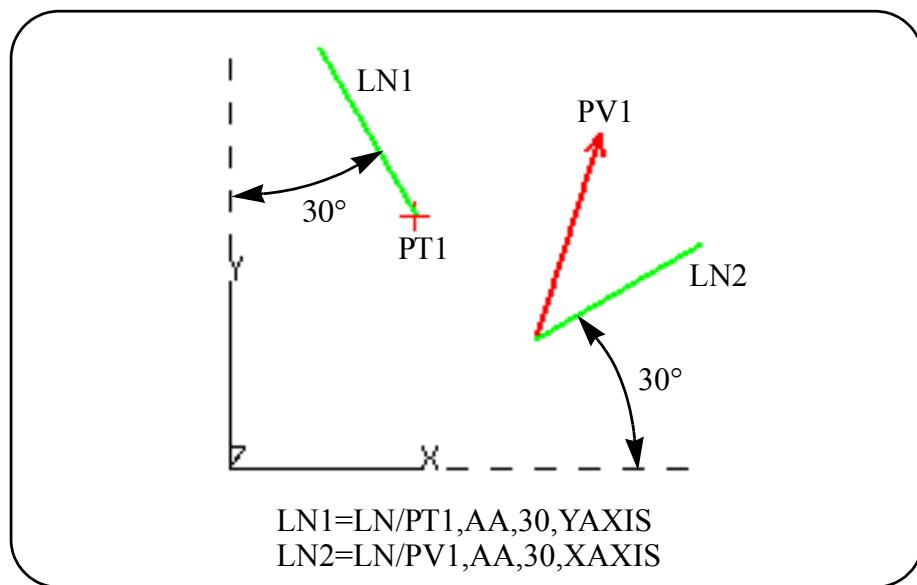
- The line that is defined will lie on the xy-plane of the "current" REFERENCE SYStem.
- The line will be parallel to the X-axis or Y-axis of the "current" REFERENCE SYStem.
- The length of the line will be one inch regardless of the system input units.
- The direction of the line will be from a point which is the specified distance and direction from the origin of the "current" REFERENCE SYStem to a point which is one inch along the positive direction of the line.

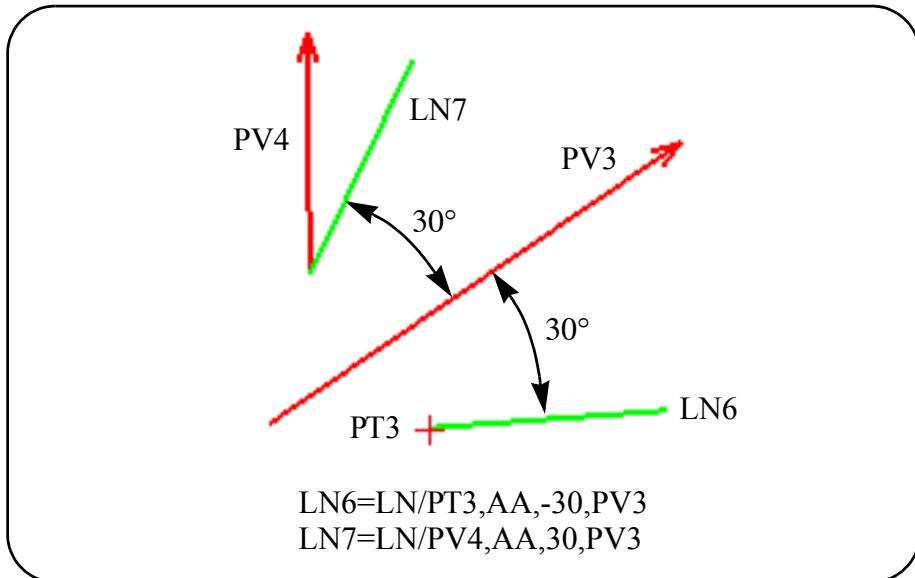
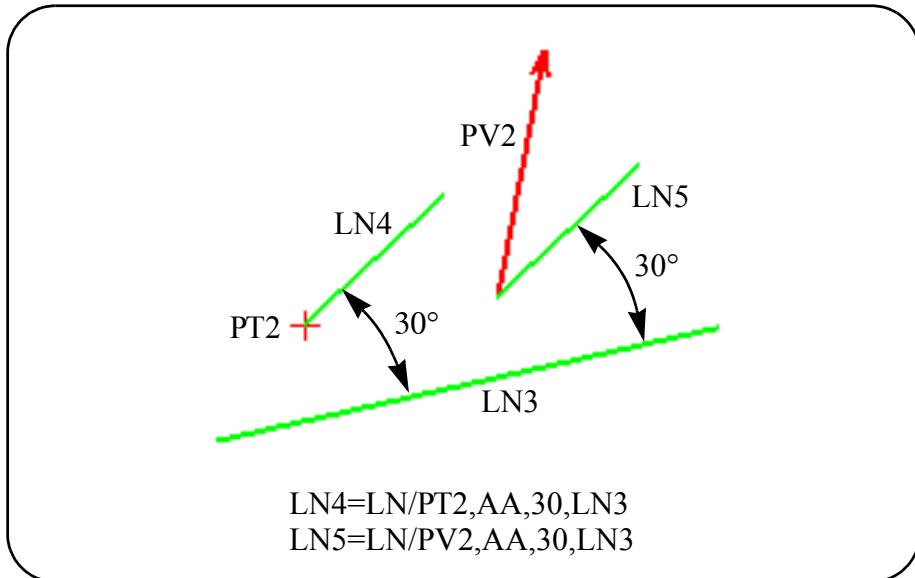
3.3.11 A Line Through A Point/Point-Vector At An Angle To The X-Axis Or the Y-Axis Or A Line Or A Point-Vector

Command Syntax:

```
LINE/point      ,ATANGL,angle[,XAXIS      ]
    point-vector          YAXIS
                           line
                           point-vector
```

Icon Menu Sequence:





NOTE: The angle, expressed in decimal degrees or degrees, minutes and seconds (deg'min^{sec}; e.g. 10 degrees, 2 minutes and 30 seconds will be entered as 10'2^30) is measured from the X-axis, Y-axis or the referenced line or the referenced point-vector. If omitted, **NCL** will use the X-axis. The angle may be a positive or negative number. A positive angle is measured in a counterclockwise direction. A negative angle is measured in a clockwise direction.

Calculation Method:

- The referenced point or the point-vector origin will be the start point of the new line.
- The length of the line will be one inch regardless of the system input units.
- The direction of the line will be defined by projecting the referenced line/point-vector onto an XY-plane containing the referenced point/point-vector. The defined line's end point is then calculated based on the given angle applied to the projected line/point-vector.

Error Conditions

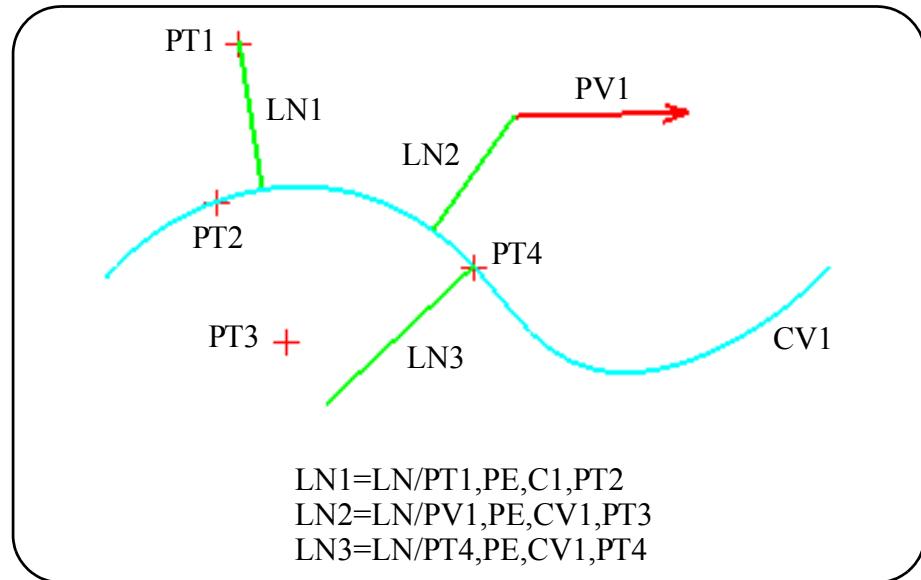
- The line/point-vector may not be zero length when projected.

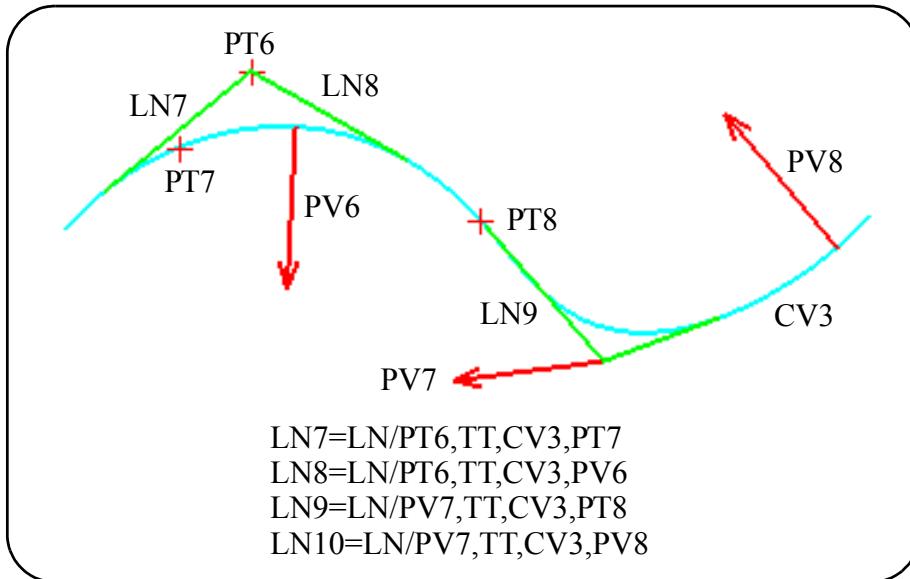
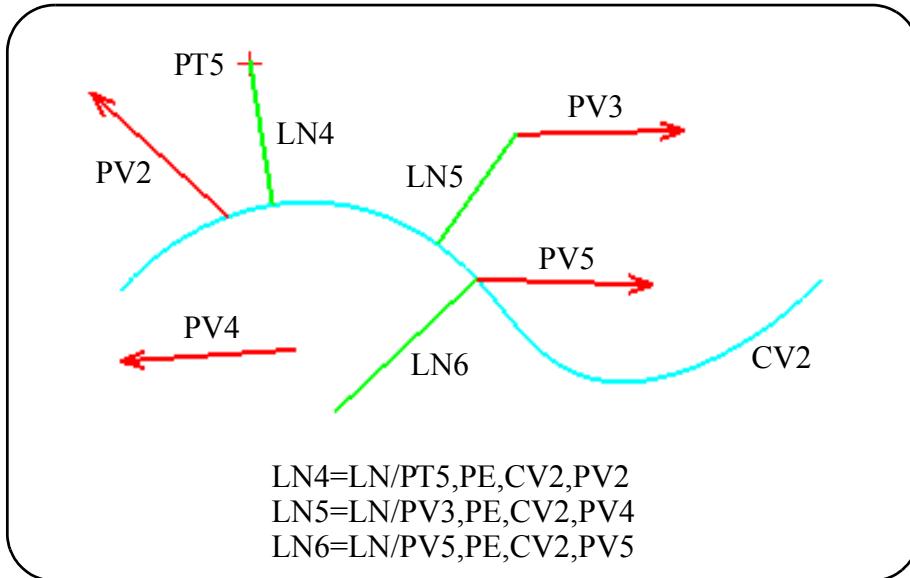
3.3.12 A Line From A Point/Point-Vector And Perpendicular Or Tangent To A Curve Near A Point/Point-Vector

Command Syntax:

```
LINE/point      , PERPTO, curve, near-point
    point-vector TANTO      point-vector
```

Icon Menu Sequence:





Calculation Method:

- The curve is projected onto a Z-plane containing the referenced point/vector.

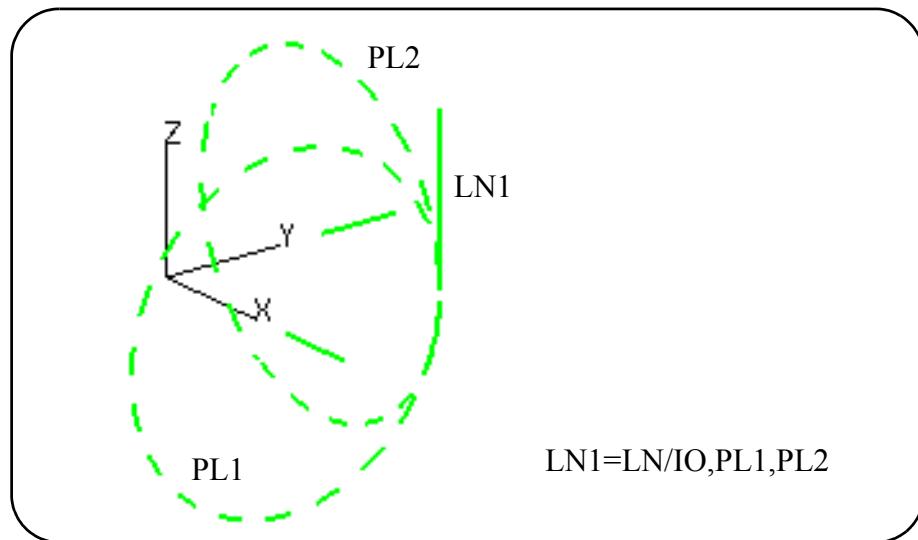
- The length of the line will be calculated from the point or the point-vector origin to the tangency or intersection point of the line and the projected curve. If the calculated length is less than 0.01 inch (0.254mm), it will be created to start at the origin point or point vector and have a length of 1.0 inch (25.4mm).
- The direction of the line will be from the point or the point-vector origin to the curve. If the point or the point-vector origin is on the line, then the direction of the line will be a function of the direction of the curve. The tangent line will be one inch along the forward sense vector, the perpendicular line will be one inch along the vector which is 90 degrees in the CLW direction to the forward sense vector.
- Both the point (or the point-vector origin) and the "near" point (or the point-vector origin) may lie on the curve, or the same location on the curve.

3.3.13 A Line As The Intersection Of Two Planes

Command Syntax:

```
LINE/INTOF,plane,plane
```

Icon Menu Sequence:



Calculation Method:

- The length of the line will be one inch regardless of the system input units.
- The direction of the line will be calculated as a vector that is the cross product of the reference plane normal vectors (first reference plane vector CROSS second reference plane vector).
- The start point of the line will be calculated as the shortest distance from the origin (of the current REference SYstem) to the line of intersection between the two planes.

Error Conditions

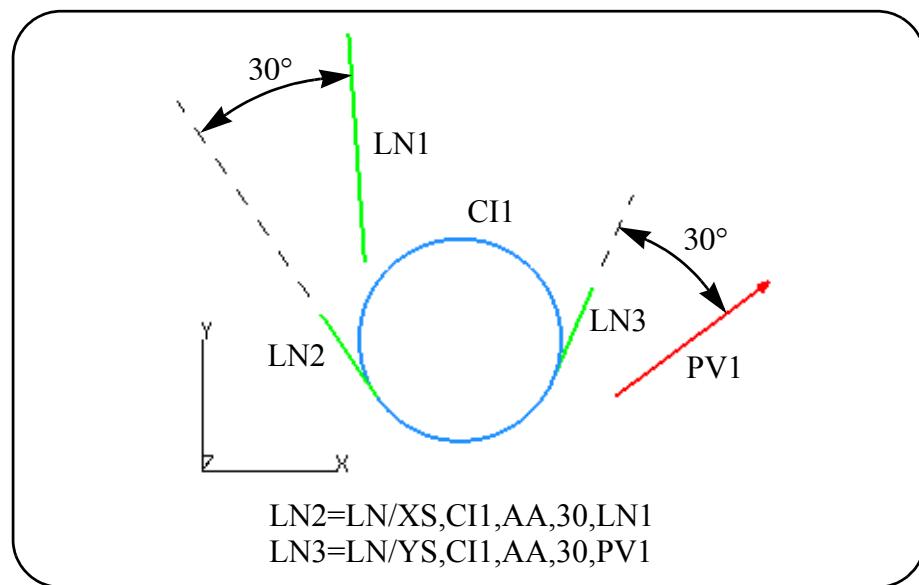
- The planes must intersect.

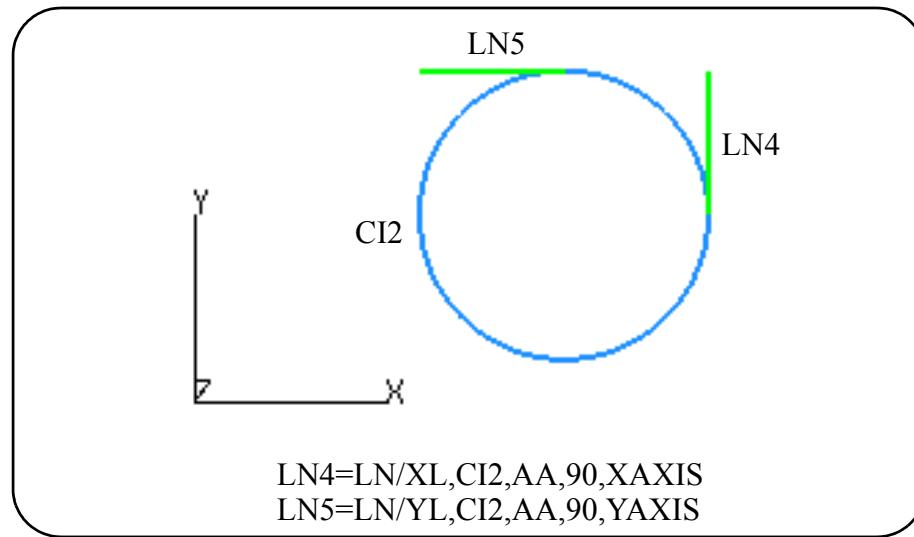
3.3.14 A Line Tangent To A Circle At An Angle To The X-Axis Or The Y-Axis Or A Line Or A Point-Vector

Command Syntax:

XLARGE	XAXIS
XSMALL	YAXIS
LINE/YLARGE,circle,ATANGL,angle[,line]	
YSMALL	point-vector

Icon Menu Sequence:





NOTE: The angle, specified in decimal degrees or degrees, minutes and seconds (deg'min^{sec}; e.g. 10 degrees, 2 minutes and 30 seconds will be entered as 10'2^30) is measured from the X-axis, Y-axis, the referenced line or the referenced point-vector. A positive angle is measured in the counterclockwise direction. A negative angle is measured in the clockwise direction.

Calculation Method:

- The referenced axis/line/point-vector is projected onto the Z-plane of the circle in the "current" REference SYStem, followed by;
- The calculation of the line which is tangent to the circle at the specified angle with respect to the X-axis, Y-axis, the "projected line" or the "projected point-vector" (X-axis is the default value).
- The start point of the line will be the tangency point of the line with the circle.
- The line will lie in the Z-plane containing the circle.
- The length of the line will be one inch regardless of the system input units.
- The direction of the line is defined from the tangency point end to the other end of the defined line.

Error Conditions:

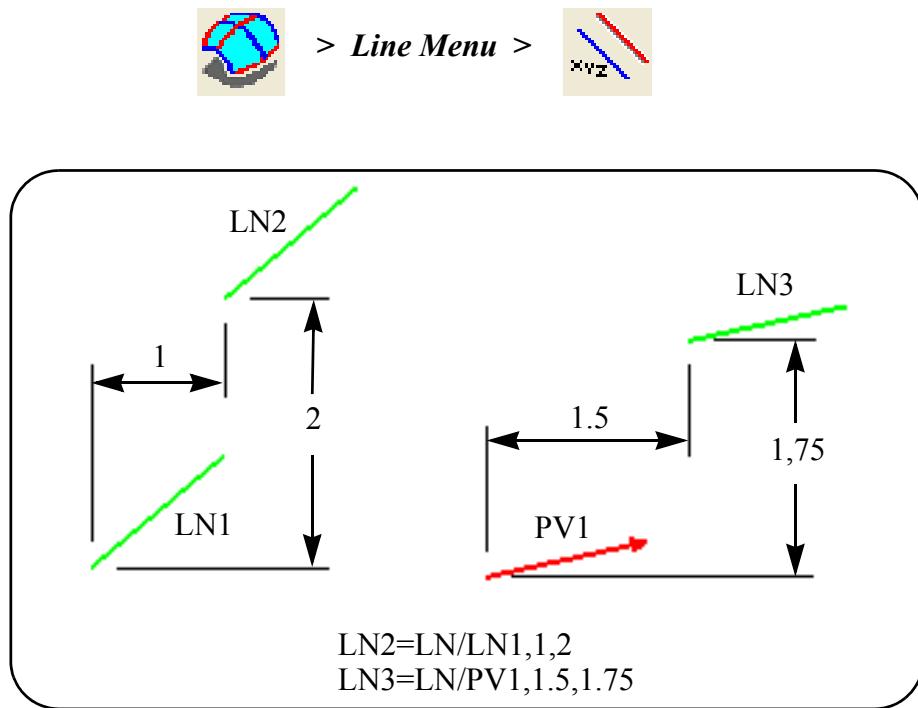
- The circle may not be "tipped" relative to the XY-plane of the "current" REference SYStem.
- The line/point-vector must not be zero length when projected.

3.3.15 A Line As The X, Y and Z Delta Distance From A Line Or A Point-Vector

Command Syntax:

```
LINE/line      [,delta-x[,delta-y[,delta-z]]]
    point-vector
```

Icon Menu Sequence:



NOTE: The line is defined by its delta distance offsets from the referenced line/point-vector. The delta values are optional. However, the fixed field format must be observed. Any delta offset may be omitted by specifying a zero (0) in the field.

Calculation Method:

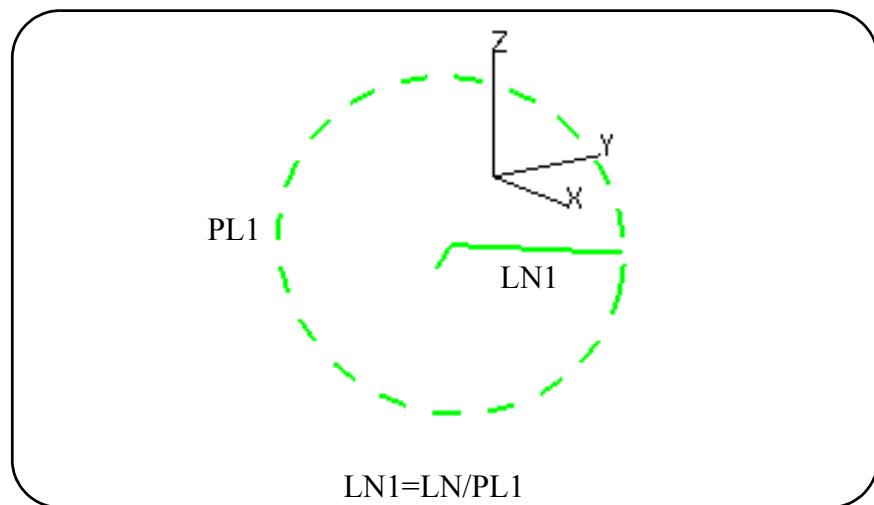
- The X, Y and Z delta values are added to the X, Y and Z values of the referenced line/point-vector's start point X, Y and Z values.
- The length of the line will be the same as the referenced line/point-vector.
- The direction of the line will be the same as the referenced line/point-vector.

3.3.16 A Line At The Intersection Of A Plane And The XY-Plane

Command Syntax:

LINE/plane

Icon Menu Sequence:



Calculation Method:

- The normal vector of the referenced plane will be projected onto the XY-plane.
- The line where the plane cuts the XY-plane is calculated.
- The intersection of a projected vector from the origin, normal to the plane and the calculated line, is the starting point of the line being defined.
- The length of the line will be one inch regardless of the system input units.
- The direction of the line will be calculated as ninety (90) degrees rotation, in the counterclockwise direction as viewed from +Z, from the direction of the projected vector.

Error Condition:

- The PLANE cannot be parallel to the xy-plane.

PATERN

The following list gives an abbreviated notation of all the valid PATERN definition formats. See the individual sections for a complete explanation of each format.

1. **PATERN/LINEAR**,point,point,number-of-points
or PATERN/LINEAR,pntvec,pntvec,number-of-pntvecs
2. **PATERN/LINEAR**,point ,vector,number-of-entity
pntvec
3. **PATERN/LINEAR**,point ,vector,INCR,dist1[...] \$
pntvec num1,AT,dist1
[[...],INCR,distn[...]]
numn,AT,distn
4. **PATERN/ARC**,circle,start-angle,end-angle,CLW ,pt-num
CCLW
5. **PATERN/ARC**,circle,start-ang,CLW ,INCR,deg1[...] \$
CCLW num1,AT,deg1
[[...],INCR,degn[...]]
numn,AT,degn
6. **PATERN/PARREL**,patern,vector,number-of-set
pntvec
7. **PATERN/PARREL**,patern,vector,INCR,dist1[...] \$
pntvec num1,AT,dist1
[[...],INCR,distn[...]]
numn,AT,distn
8. **PATERN/PROJCT**,patern[,vector],\$
ATANGL,start[,end]
surface[[u,v]][,NOWRAP][,START] \$
WRAP MIDDLE
REVOLV END
RADIAL CENTER
AT,point1
[,near-point]
pntvec

Example PATTERN Expressions:

```
PN/LINEAR, PT1, PT2, 7
PN/LINEAR, PV1, PV2, 7
PN/LINEAR, PV1, VE2, INCR, 1, 2, 3
PN/LINEAR, PT1, VE1, INCR, 4, AT, 1, INCR, .5, .75
PN/LINEAR, PV1, VE2, INCR, 4, AT, 3.5
PN/LINEAR, PV1, VE1, INCR, 4, AT, 1, INCR, .5, .75
PN/ARC, CI1, 15, 125, CLW, 6
PN/ARC, CI1, 90, CLW, INCR, 15, 10, 15, 30
PN/PARALLEL, PN1, VE1, 6
PN/PARALLEL, PN1, PV2, 3
PN/PARALLEL, PN1, VE1, INCR, 1.25, 2, 1, INCR, 7, AT, .25
PN/PARALLEL, PN1, PV2, INCR, 3.5, 1, 1, INCR, 2, AT, 3
PN/RANDOM, PT4, PT2, PN7, PT9
PN/RANDOM, PT1, THRU, PT4
PN/RANDOM, PV1, THRU, PV6
```

3.4

PATERN Expressions

The PATERN statement generates a series of X-Y-Z coordinates to direct the cutter motion. Patterns are arranged in a Linear, Circular, Parallel or Random order. The X-Y-Z coordinates generated will be displayed as plus (+) signs.

The PATERN statement may also be used to generate a series of point-vectors in place of x, y, z coordinate points. If a PATERN statement contains a point specifier, then a pattern of points will be created, if it contains a point-vector, then a pattern of point-vectors will be created. However, point-vector patterns are limited to Linear, Parallel or Random patterns.

Patterns may contain from 1 to 32,766 points (or point-vectors). Pattern points (or point-vectors) are assigned a number in the sequence of their generation. When used with motion, the cutter will move to the first point generated and proceed in sequential order to the last point.

Pattern points (or point-vectors) are NOT the same as normal **NCL** points. The coordinates of a specific pattern point can be assigned to an **NCL** point using point definition "PT/ PN, id-number." See Point Definition for further information.

A LINEAR pattern generates points (or point-vectors) in a straight line from the first point to the last point. The pattern can be defined using two points or from a point and vector. The points can be spaced equally or in varying increments.

ARC generates a series of points on a circle. The points can be clockwise or counterclockwise.

PARREL generates a parallel grid pattern in a zig-zag or lace cut motion. The points (or point-vectors) are generated from a pattern and vector.

RANDOM combines points (or point-vectors) and patterns to form one complex pattern. The points are numbered in the sequence that they were originally defined.

All z-coordinates assigned to the points in a PATERN will be computed based on the ZSURF value in effect at the time the PATERN expression is processed. If ZSURF/NOMORE is in effect at the time the PATERN expression is processed, the z-coordinates will be based on the defining geometry's z-coordinate value.

Special Conditions:

- **YELLOW** (pen 7) is the **NCL** System DEFAULT color for CAM PATERNs.

Note: The color **ORANGE** will be used in examples of this chapter for better contrast with a white background.

3.4.1 A Linear Pattern Between Two Points Or Two Point-Vectors

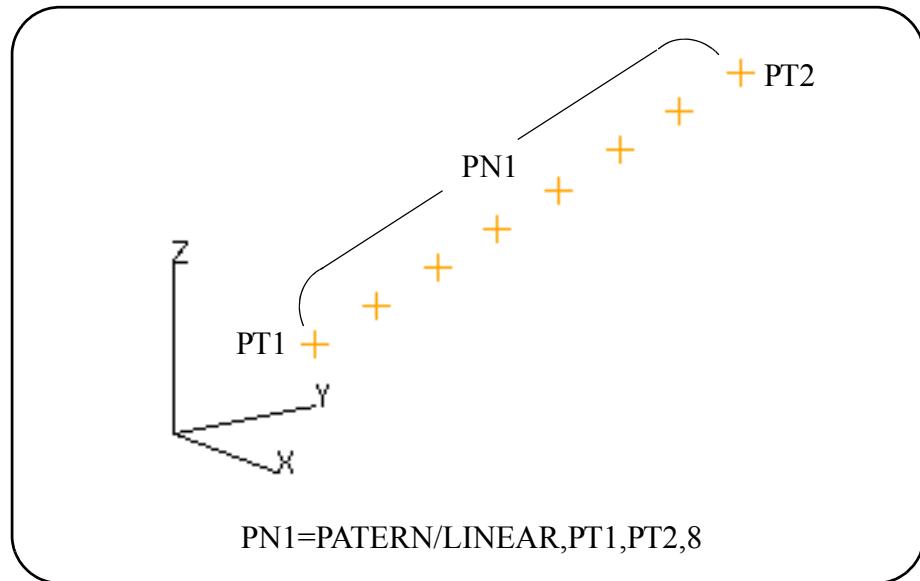
Command Syntax:

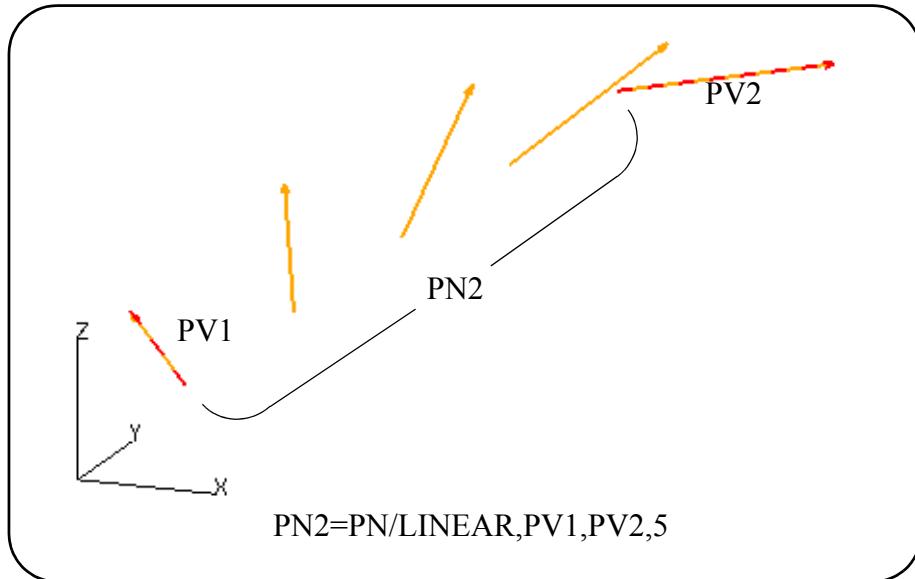
PATERN/LINEAR,point,point,number-of-points

or

PATERN/LINEAR,pntvec,pntvec,number_of_pntvecs

Icon Menu Sequence:





Calculation Method:

- A line is constructed in 3-D space between the first specified point (or the origin of the first specified point-vector) and the second specified point (or the origin of the second specified point-vector).
- The line is broken into a number of equal length segments. The number of segments is one less than the number of points/point-vectors specified in the PATERN.
- If the two specified end points (or the origin of the two specified point-vectors) have differing z-coordinate values, the resulting intermediate points (or the point-vectors) in the PATERN will have z-coordinate values that vary linearly from the z-coordinate value of the first point/vector to the z-coordinate value of the second point/vector.
- If point-vectors are specified, the resulting point-vectors will have gradually changing directions and magnitudes from the first point-vector to the last one.

Error Condition:

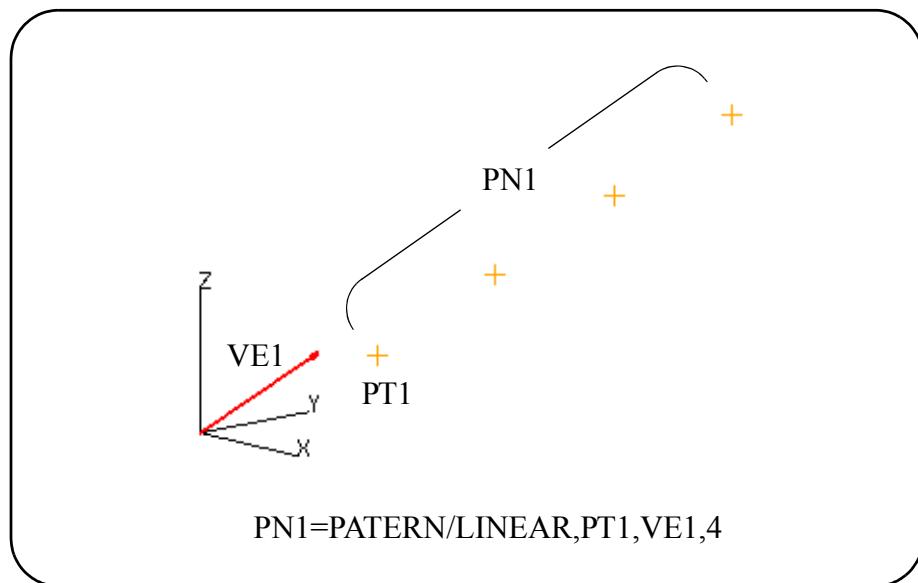
- A minimum of two points or two point-vectors are required in this PATERN definition.

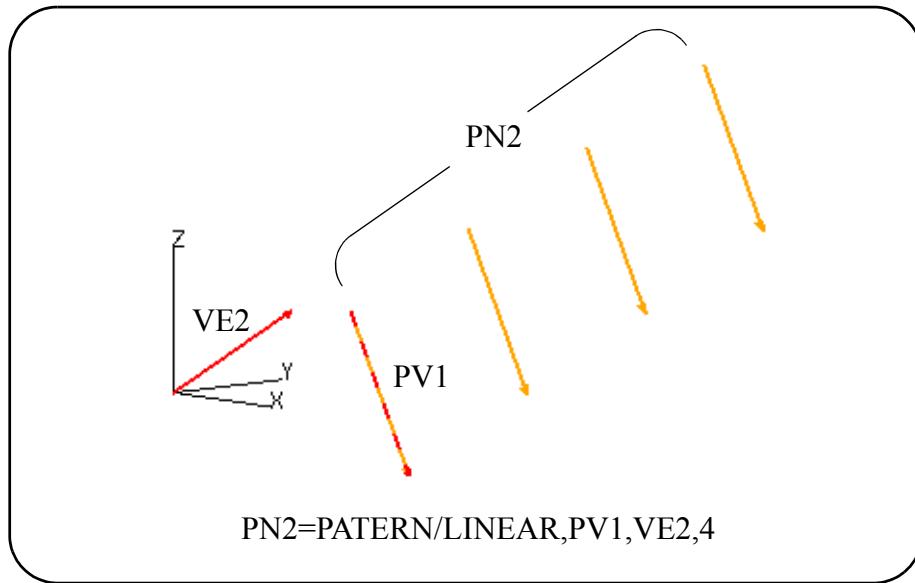
3.4.2 A Linear Pattern Along A Vector At A Start Point/Point-Vector With Distance Between Individual Elements Equal To The Magnitude Of The Vector

Command Syntax:

```
PATERN/LINEAR,point      ,vector,number-of-entity  
          point-vector
```

Icon Menu Sequence:





Calculation Method:

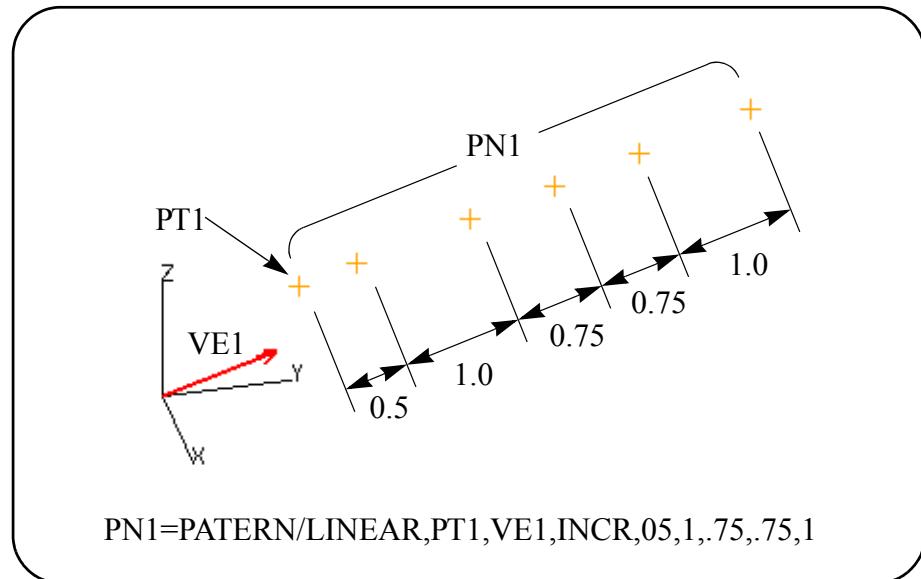
- The specified point or the specified point-vector is the first entity of the PATERN.
- The second entity of the PATERN is calculated from the first entity along the specified vector at a distance equal to the length of the vector.
- Each subsequent entity in the PATERN is calculated from the previous one in the same manner as the second entity is calculated from the first entity.

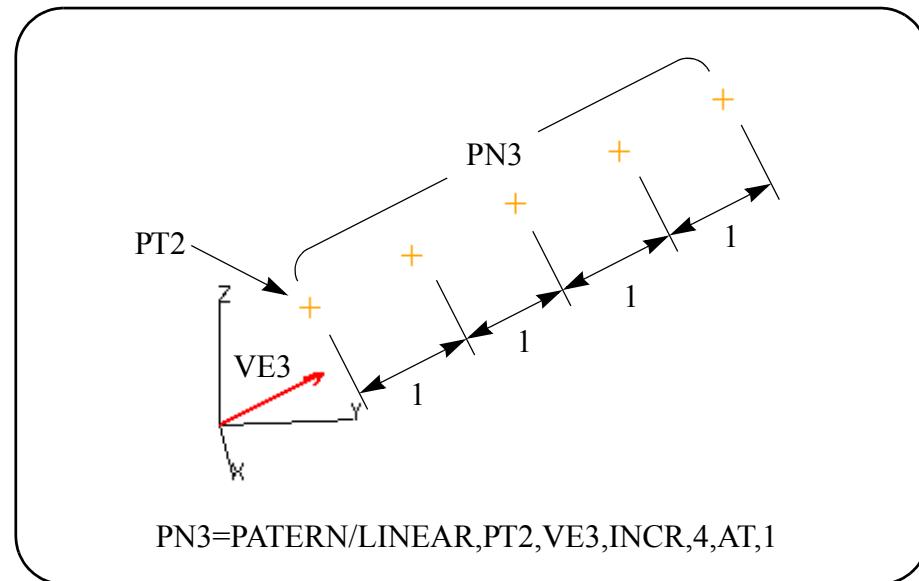
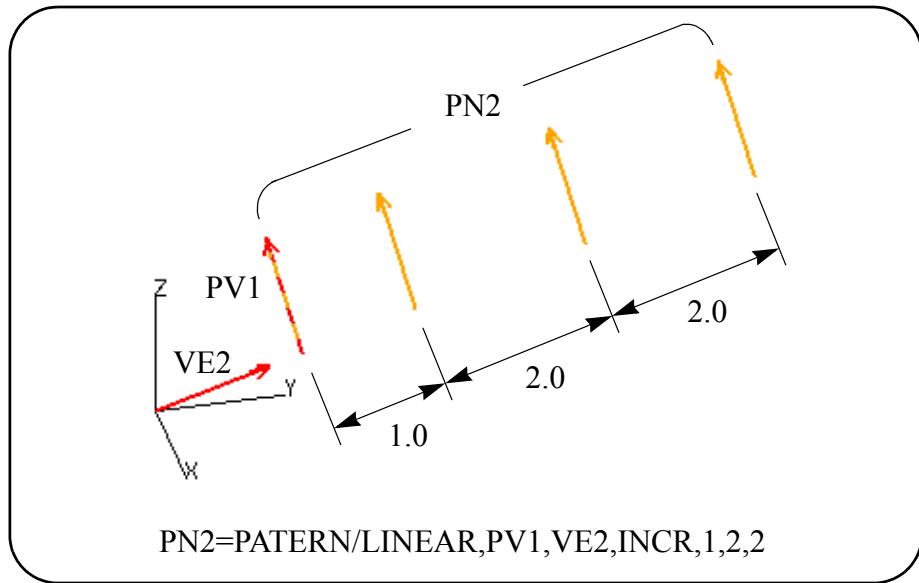
3.4.3 A Linear Pattern Along A Vector With A Start Point/Point-Vector And Specific INCR Clauses

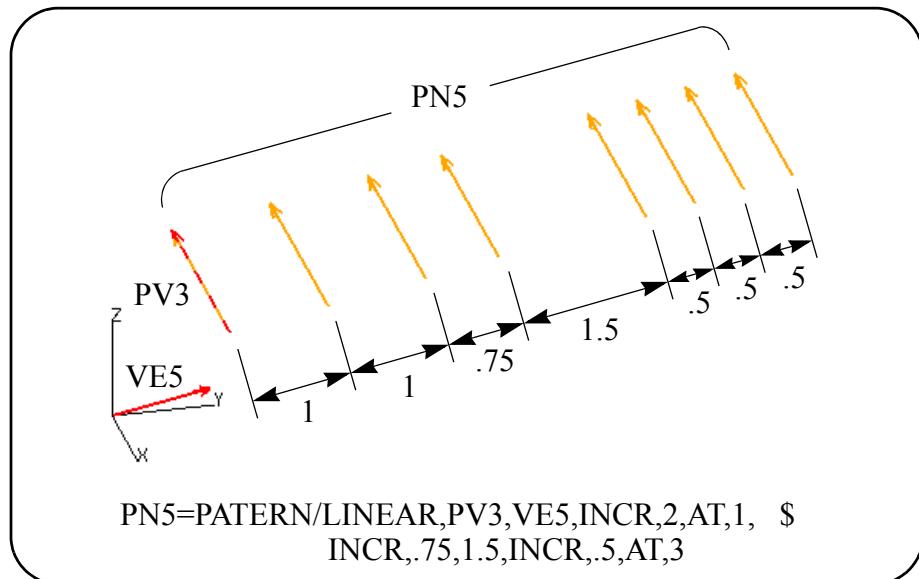
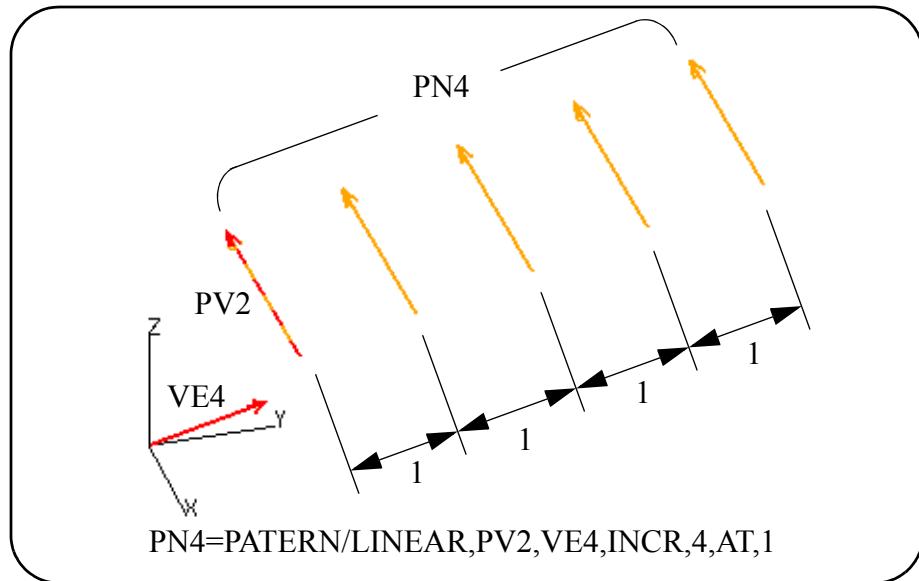
Command Syntax:

```
PATERN/LINEAR,point ,vector,INCR,dist1[...]      $  
          pntvec           num1,AT,dist1  
          [...] ,INCR,distn[...] ]  
          numn,AT,distn
```

Icon Menu Sequence:







Calculation Method:

- The specified point or point-vector is the first entity of the PATERN.

Case A: The following applies to the situation where the keyword “AT” is not specified.

- The second entity of the PATERN is calculated from the first specified entity along the specified vector at a distance specified by the first value given after the INCR key word.
- Each subsequent entity in the PATERN is calculated from the previous one along the specified vector at a distance specified by the next value given in the INCR portion of the statement.
- The total number of entity in the PATERN is determined by the number of incremental distances given in the expression, plus one for the specified starting entity.

Case B: The following applies to the situation where the keyword “AT” is specified.

- The second entity of the PATERN is calculated from the first specified entity along the specified vector at a distance specified after the keyword AT.
- Each subsequent entity in the PATERN is calculated from the previous one along the specified vector at the same distance specified after the keyword AT.
- The total number of entity in the PATERN is determined by “num”, plus one for the specified starting entity.

Case C: The following applies to the situation where the INCR clause is a combination of “Case A” and “Case B”.

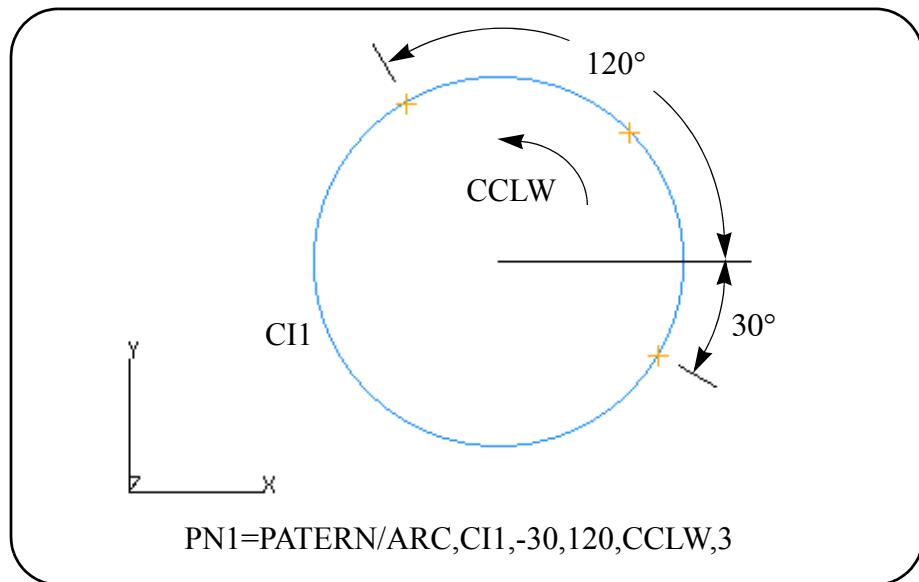
- Everything specified for “Case A” and “Case B” are true with the exception of the following:
- The last entity of the previous INCR clause becomes the first entity of the next INCR clause.
- The total number of entity in the PATERN will be the summation of all the entities for “Case A”’s and “Case B”’s, plus one for the specified starting entity and minus one for each appearance of the INCR clause.

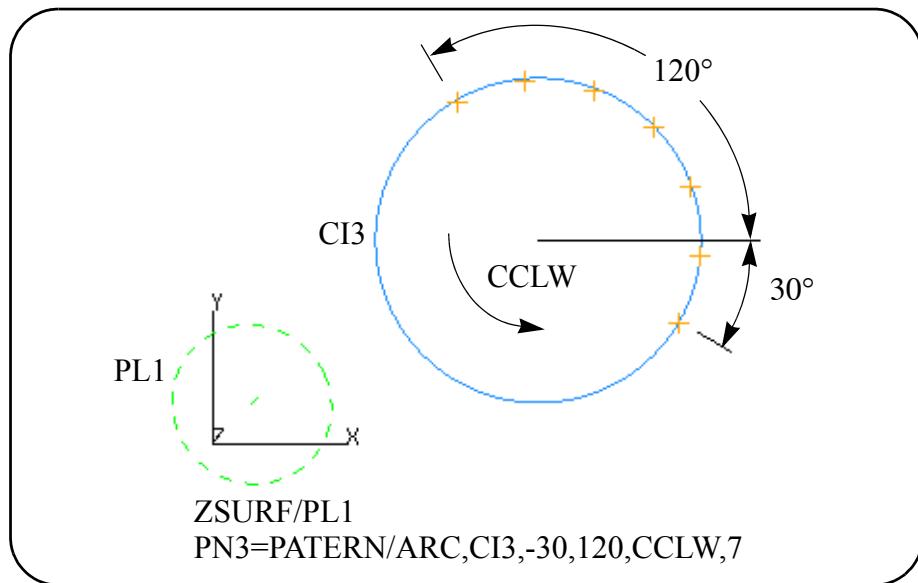
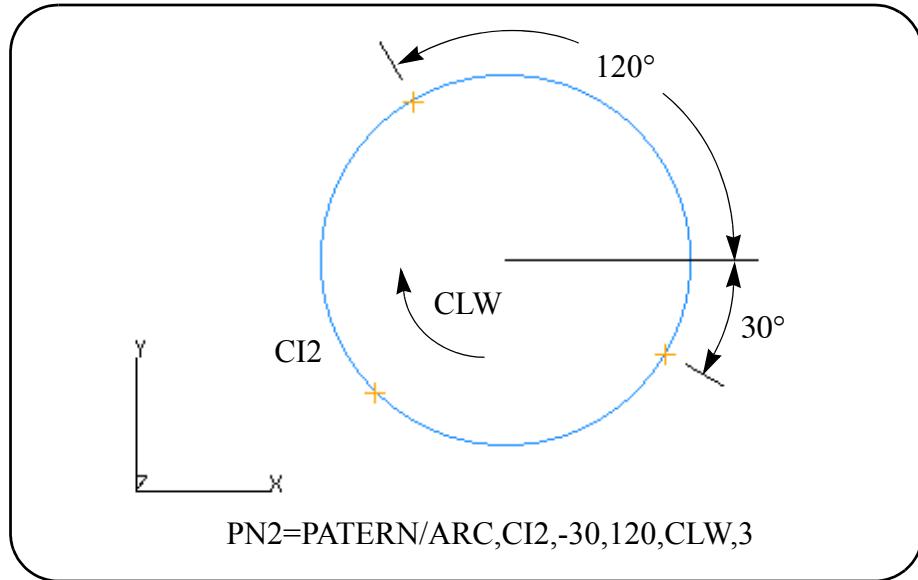
3.4.4 A Pattern Of Points Equally Spaced Around A Circle Between Two Specified Angles

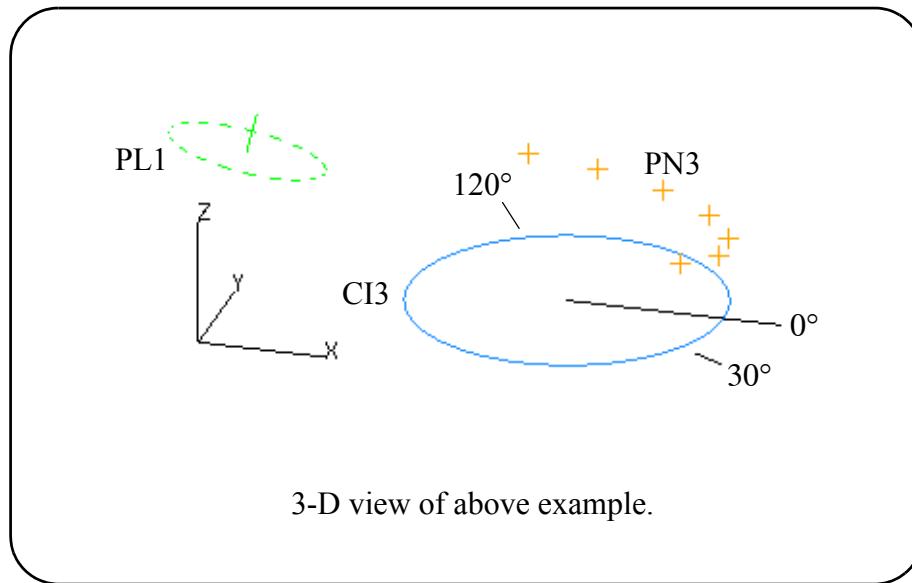
Command Syntax:

```
PATTERN/ARC,circle,start-angl,end-angl,CLW ,pt-num  
CCLW
```

Icon Menu Sequence:







This definition generates a pattern of points at the current **ZSURF** setting which are equally spaced around the circumference of a circle by specifying the start angle, end angle, the rotation direction and the total number of points desired. The angles may be expressed in decimal degrees or degrees, minutes and seconds (deg' min" sec).

Calculation Method:

- An internal Polar Coordinate System is set up in the center of the specified circle. This internal system is used for the generation of the PATTERN around the specified circle.
- The first point of the PATTERN is given in polar coordinate (radius, angle) as measured from the standard position at 0 degrees (3 o'clock). The “start-angle” specifies the first point angular coordinate.
- The last point of the PATTERN is given by the polar coordinate measured from the standard position. The “end-angle” specifies the last point angular coordinate.
- Positive angles specified for the start-angle and the end-angle indicate a counterclockwise direction and negative angles indicate a clockwise direction.
- “CLW/CCLW” specifies which rotational direction the pattern of points will be generated.

- The pt-num specifies the total number of points in this PATTERN which are equally spaced around the circumference of the circle.
- The z-coordinates of the individual points in the generated pattern will be affected by the current ZSURF setting, the generated points might not be on the circle. The points of the generated pattern will only be exactly on the circle if ZSURF is off.

Error Conditions

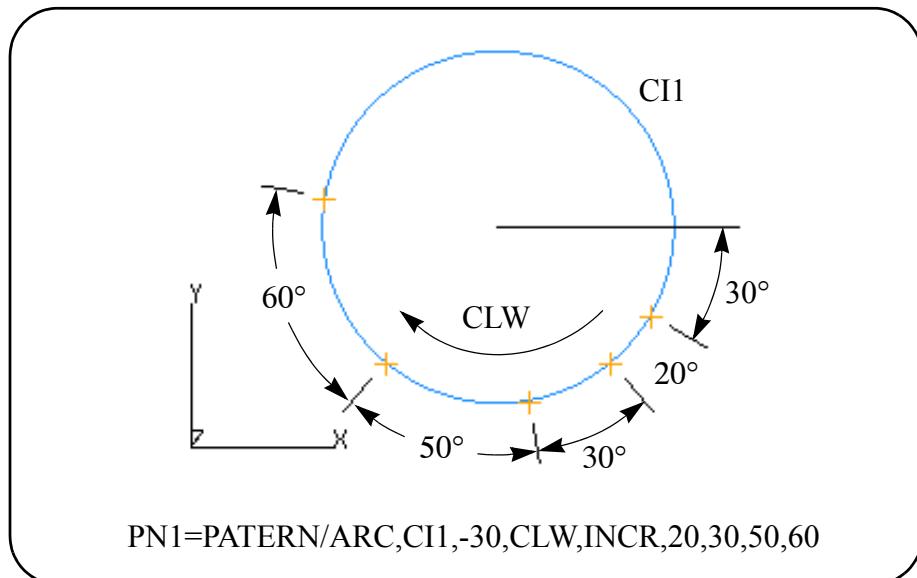
- The plane of the specified circle must be parallel to the XY-plane of the current [REFerenced SYStem](#).

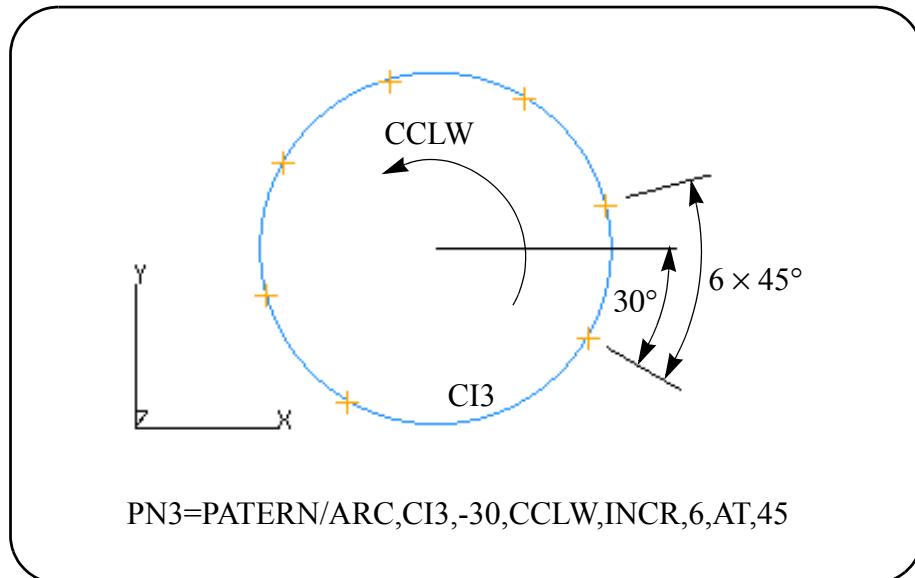
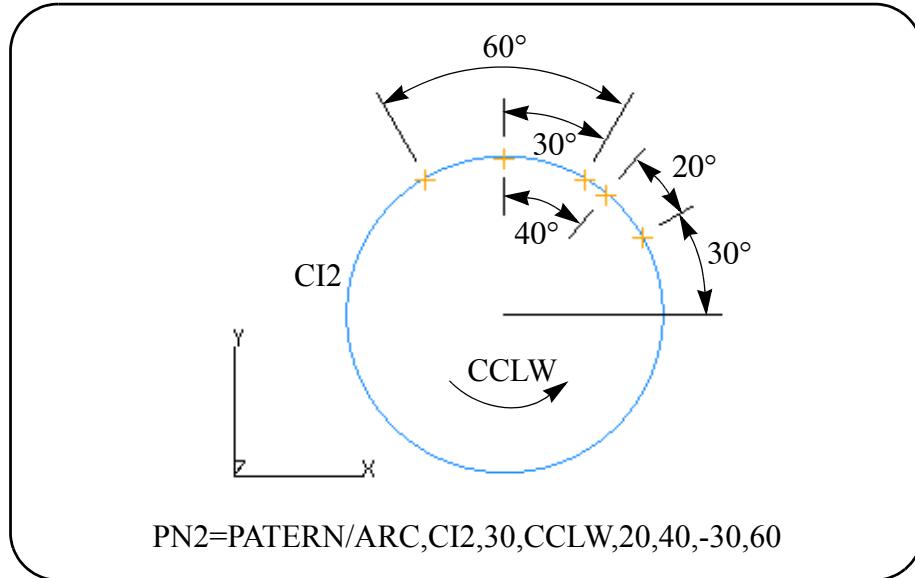
3.4.5 A Pattern Of Points Around A Circle With A Start Angle And Specific INCR Clauses

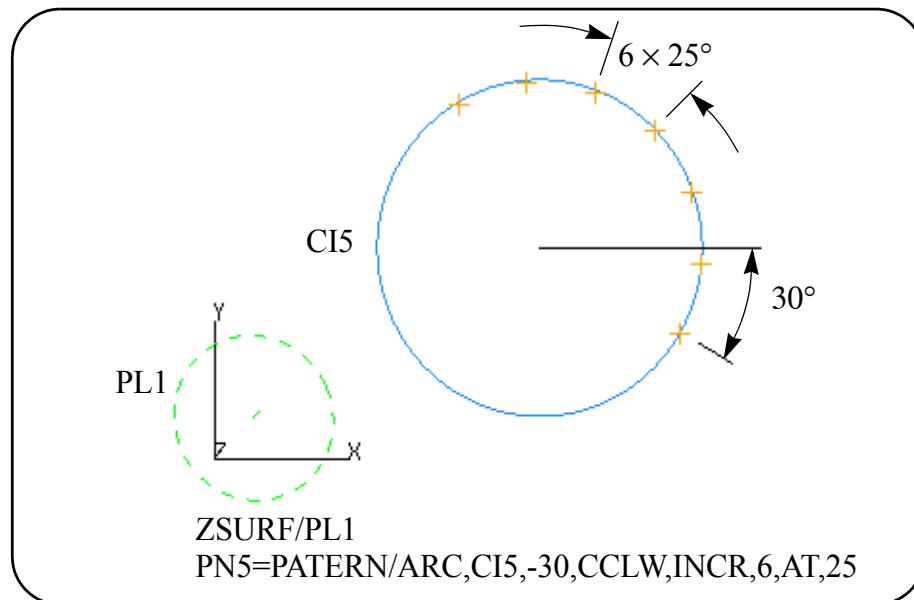
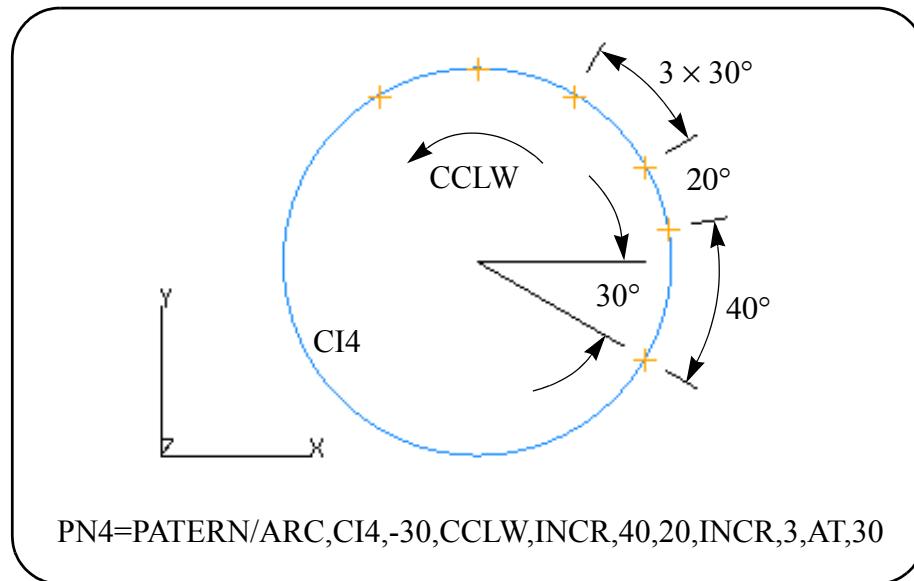
Command Syntax:

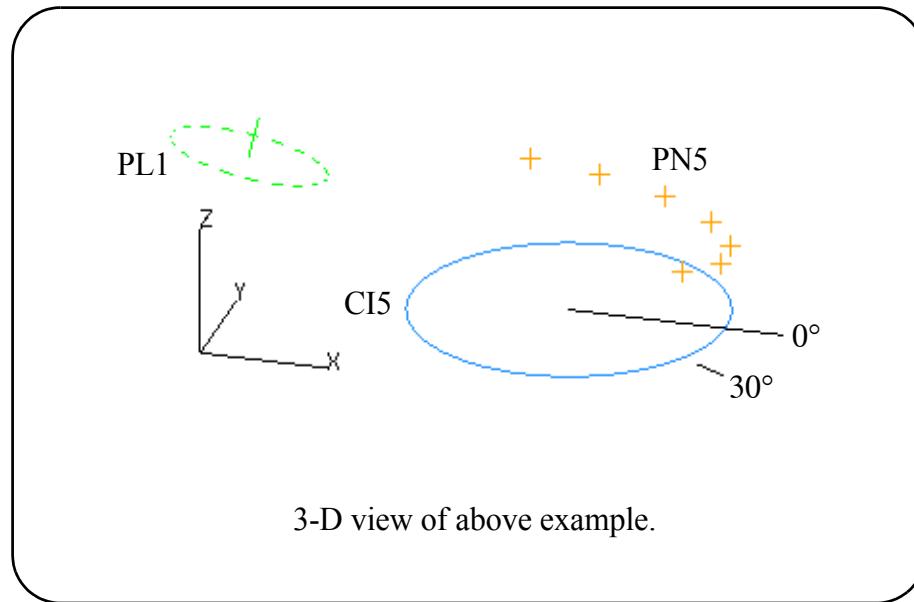
```
PATERN/ARC,circle,start-angle,CLW , $  
CCLW  
INCR,deg1[...] [...] ,INCR,degn[...] ]  
num1,AT,deg1 numn,AT,degn
```

Icon Menu Sequence:









This definition generates a pattern of points at the current **ZSURF** setting around the circumference of a circle by specifying the start angle, rotation direction and the desired INCR clauses. The angles may be expressed in decimal degrees or degrees, minutes and seconds (deg' min[^] sec).

Calculation Method:

- An internal Polar Coordinate System is set up in the center of the specified circle. This internal system is used for the generation of the PATTERN around the specified circle.
- The first point of the PATTERN is given in polar coordinate (radius, angle) as measured from the standard position at 0 degrees (3 o'clock). The “start-angle” specifies the first point angular coordinate.
- Positive angle specifies for the start-angle indicates a counterclockwise direction and negative angles indicate a clockwise direction.
- “CLW/CCLW” specifies which rotational direction the pattern of points will be generated.
- The z-coordinates of the individual points in the generated pattern will be affected by the current ZSURF setting, the generated points might not be on the circle. The points of the generated pattern will only be exactly on the circle if ZSURF is off.

Case A: The following applies to the situation where the keyword “AT” is not specified.

- The second point of the PATTERN is given by the polar coordinate measured from the start position along the circumference at an angular value given after the INCR clause.
- Each subsequent point in the PATTERN is calculated from the previous one along the circumference at an incremental angular value specified by the next value given in the INCR portion of the statement.
- The number of points in the PATTERN is determined by the number of incremental angular values given in the expression after the INCR clause, plus one for the starting point.

Case B: The following applies to the situation where the keyword “AT” is specified.

- The second point of the PATTERN is calculated from the first specified point along the circumference at the angular value specified after the keyword AT.
- Each subsequent point in the PATTERN is calculated from the previous one along the circumference at the angular value specified after the keyword AT.
- The total number of points in the PATTERN is determined by “num”, plus one for the specified starting point.

Case C: The following applies to the situation where the INCR clause is a combination of “Case A” and “Case B”.

- Everything specified for “Case A” and “Case B” are true with the exception of the following:
- The last entity of the previous INCR clause becomes the first entity of the next INCR clause.
- The total number of points in the PATTERN will be the summation of all the points for “Case A”’s and “Case B”’s, plus one for the specified starting point and minus one for each appearance of the INCR clause.

Error Condition:

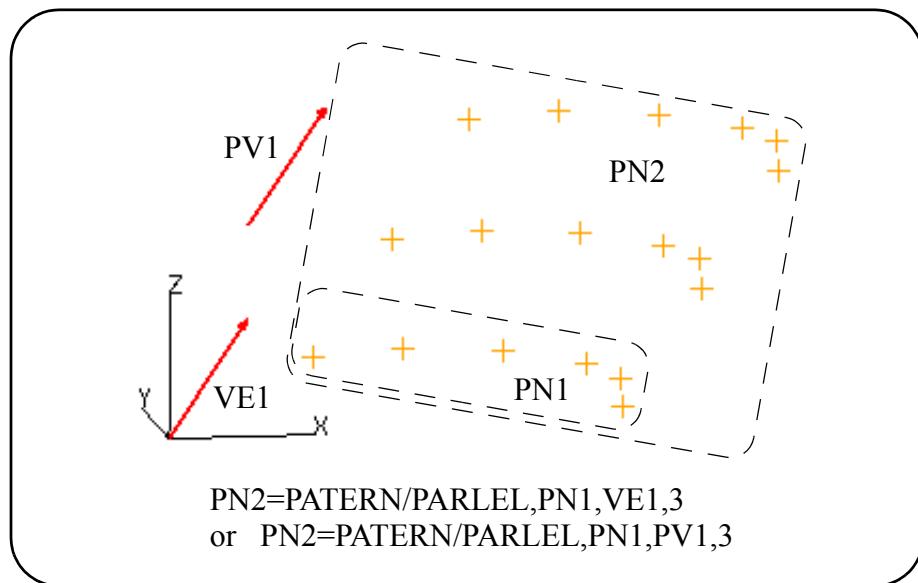
- The plane of the specified circle must be parallel to the XY-plane of the current REFerenced SYstem.

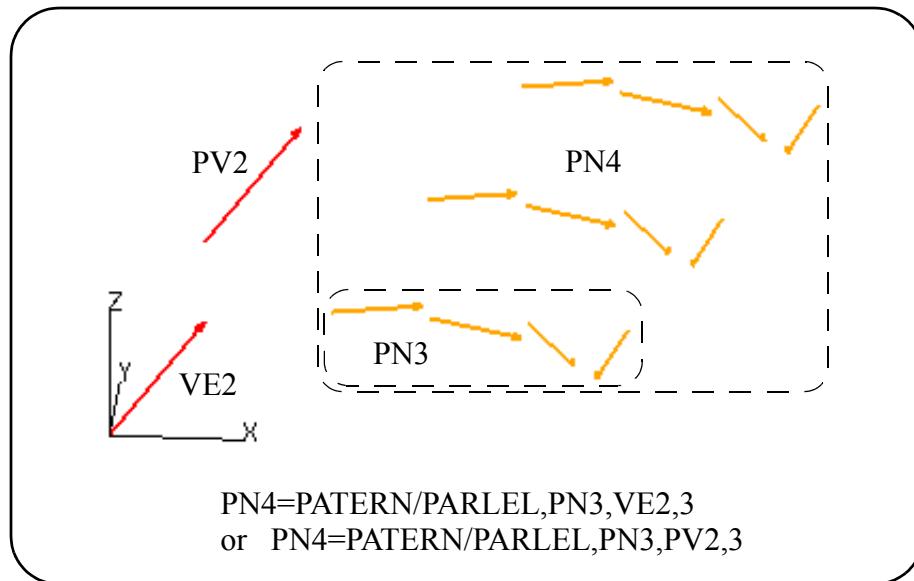
3.4.6 A Pattern Parallel To A Pattern In The Direction Of A Vector/Point-Vector

Command Syntax:

```
PATERN/PARREL,pattern,vector      ,number-of-sets  
                                point-vector
```

Icon Menu Sequence:





Calculation Method:

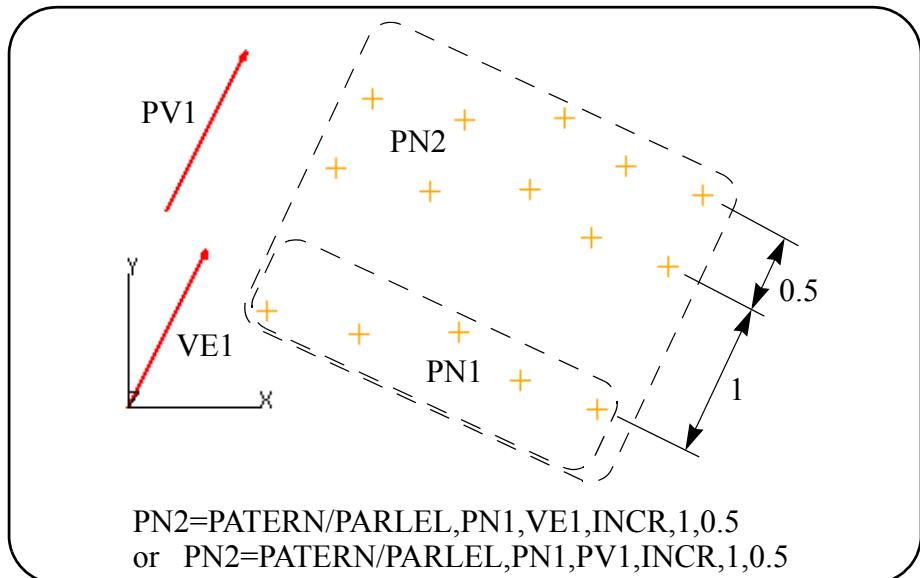
- The first set of pattern is same as the previously defined specified PATERN.
- The second set of pattern is calculated from the first set of pattern along the specified vector/point-vector at a distance equal to the length of the vector/point-vector.
- Each subsequent set of pattern is calculated from the previous one in the same manner as the second set of pattern is calculated from the first set of pattern.
- The total number of entities in the PATERN is equal to the total number of entities of the specified PATERN times the number of sets specified.

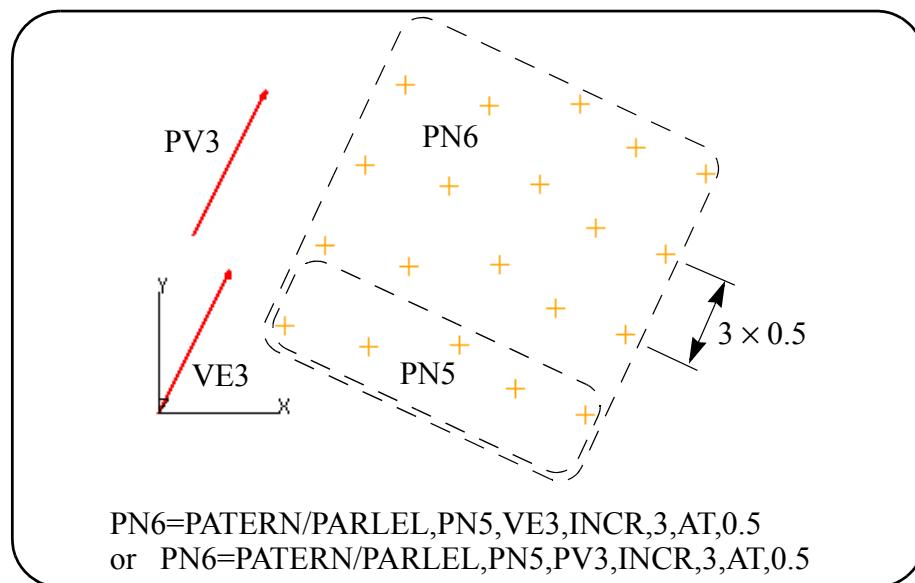
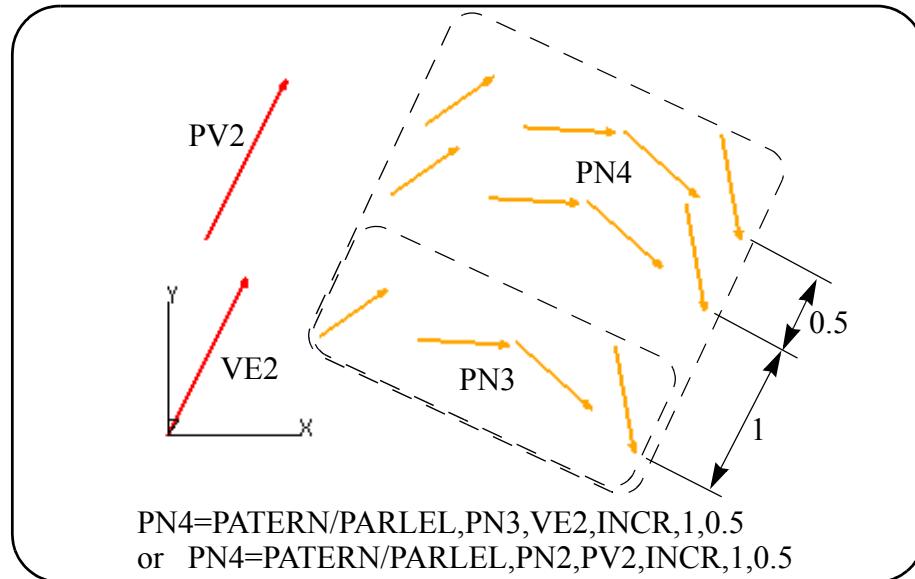
3.4.7 A Pattern That Is Parallel To A Pattern In the Direction Of A Vector/Point-Vector With Specified INCR Clauses

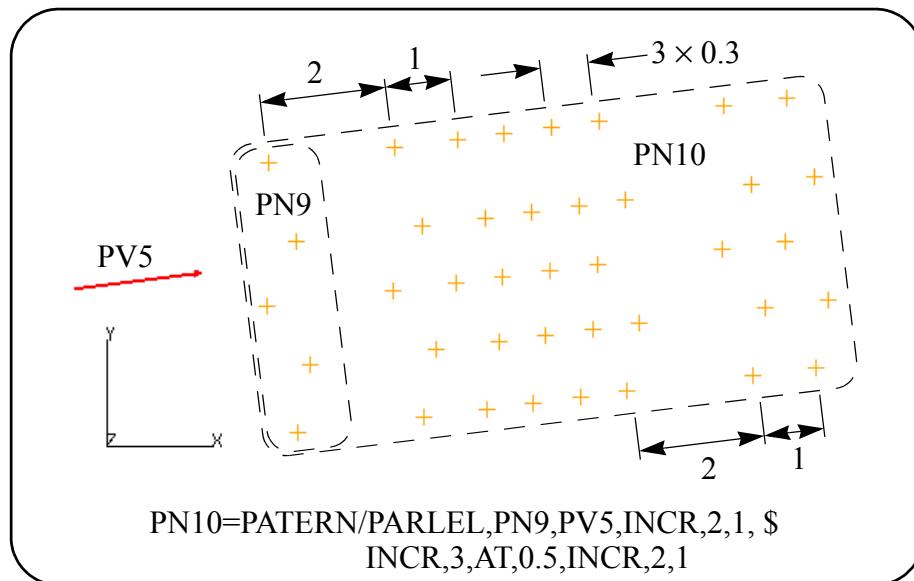
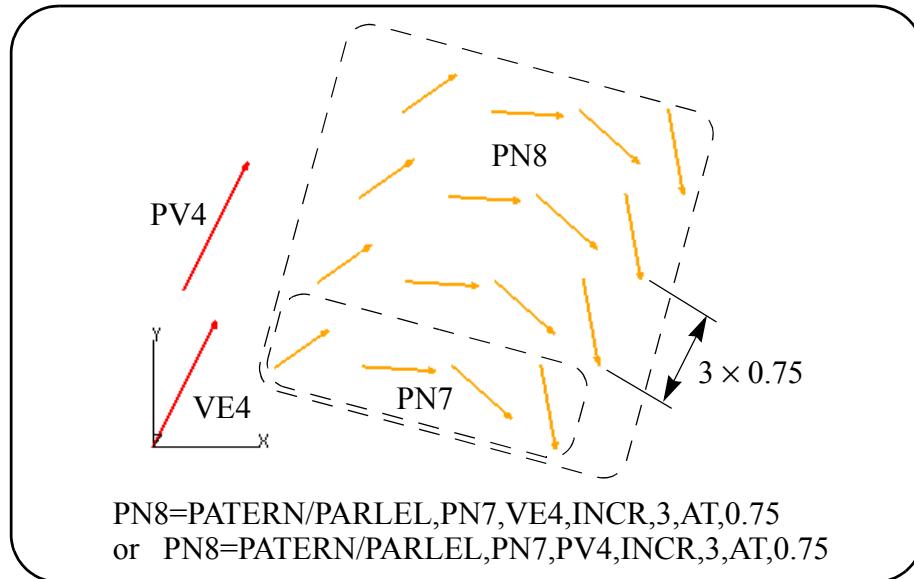
Command Syntax:

```
PATERN/PARALLEL,pattern,vector,INCR,dist1[,...]      $
          pntvec      num1,AT,dist1
[ [...],INCR,distn[...]      ]
          numn,AT,distn
```

Icon Menu Sequence:







Calculation Method:

- The specified pattern is the first entity set of the PATERN.
- The total number of individual entities in the PATERN is determined by the number of individual entities in the specified pattern set times the total number of sets in the PATERN.

Case A: The following applies to the situation where the keyword “AT” is not specified.

- The second entity set of the PATERN is calculated from the first specified entity set along the specified vector/point-vector at a distance specified by the first value given after the INCR key word.
- Each subsequent entity set in the PATERN is calculated from the previous one along the specified vector/point-vector at a distance specified by the next value given in the INCR portion of the statement.
- The total number of entity sets in the PATERN is determined by the number of incremental distances given in the expression, plus one for the specified starting entity set.

Case B: The following applies to the situation where the keyword “AT” is specified.

- The second entity set of the PATERN is calculated from the first specified entity set along the specified vector at a distance specified after the keyword AT.
- Each subsequent entity set of the PATERN is calculated from the previous one along the specified vector at the same distance specified after the keyword AT.
- The total number of entity sets in the PATERN is determined by “num”, plus one for the specified starting entity set.

Case C: The following applies to the situation where the INCR clause is a combination of “Case A” and “Case B”.

- Everything specified for “Case A” and “Case B” are true with the exception of the following:
- The last entity set of the previous INCR clause becomes the first entity set of the next INCR clause.
- The total number of entity set in the PATERN will be the summation of all the entities set for “Case A”s and “Case B”s, plus one for the specified starting entity set and minus one for each appearance of the INCR clause.

3.4.8 A Pattern Obtained By Projecting A Pattern On A Surface

Command Syntax:

```
PATTERN/PROJCT,pattern[,vector]$,  
ATANGL,start[,end]  
  
surface[[u,v]][,NOWRAP][,START]$  
WRAP      MIDDLE  
REVOLV   END  
RADIAL    CENTER  
AT,point  
  
,near-point]  
pntvec
```

The inner square brackets must be specified as part of the syntax if the optional u,v parameters are used.

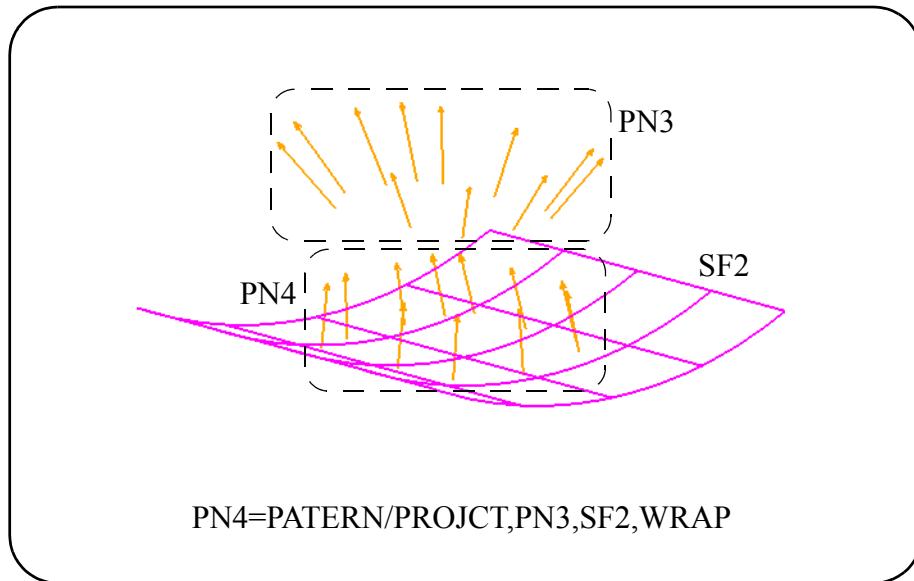
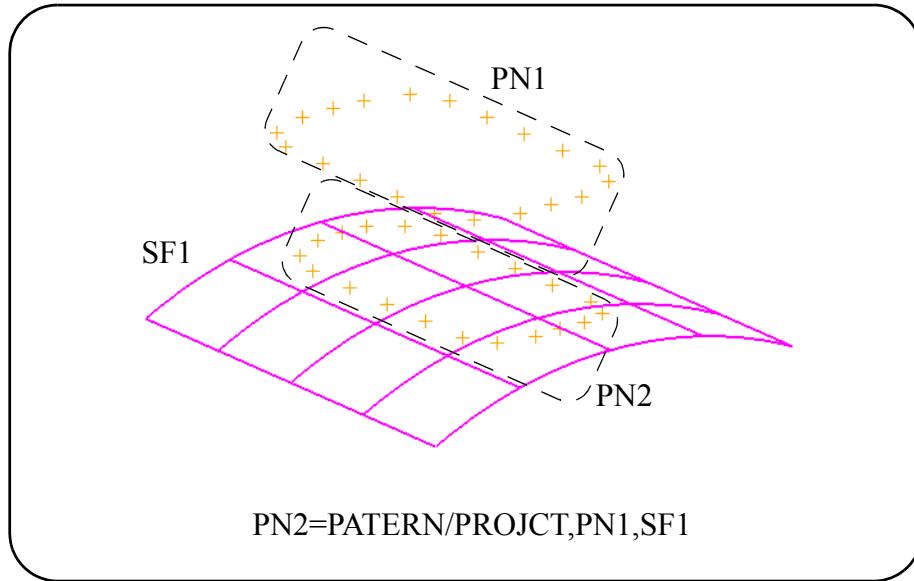
Where:

pattern	The PATTERN to be projected onto the surface.
surface	The surface onto which the PATTERN will be projected, no NET surface allowed.
[u,v]	Optional [u,v] values to tell NCL to “look” at the surface in area specified by the UV values. This is sometimes necessary on surfaces as there may be multiple projection possibilities.
vector	Optional projection vector.
ATANGL	Optional parameter specifies that an interpolation surface will be used and the curve will be projected at an angle to the forward sense of the specified curve.
start	Starting angle to use with interpolating surface.
end	Ending angle to use with interpolating surface.
NOWRAP	The specified PATTERN will be projected onto the specified surface without any wrapping. This will generate a distorted pattern.

WRAP	The specified PATTERN will be projected onto the specified surface with wrapping. The size and the shape of the generated pattern will be approximately maintained.
REVOLV	This only applies when the specified surface is a cone, a cylinder, a surface of revolution or a ruled surface. The size of the generated pattern will be approximately maintained, but not the shape.
RADIAL	This only applies when the specified surface is a cone or a surface of revolution. The shape of the generated pattern will be approximately maintained, but not the size.
START	Specifies the base of projection is at the beginning of the PATTERN.
MIDDLE	Specifies the base of projection is at half way along the PATTERN.
END	Specifies the base of projection is at the end of the PATTERN.
CENTER	Specifies the base of projection is at the center of a bounding box formed by the PATTERN.
AT	Specifies the user explicitly specified a point as the base of projection.
point	The user selected base point.
near-point or near-pntvec	Optional parameter (point or point-vector) given to clarify which of the several possible projections is desired.

Icon Menu Sequence:





Calculation Method:

- Each of the individual entities of the specified PATERN will be projected onto the specified surface according to the “Method” specified. If the individual entity of the specified PATERN is of the type point-vector, only the origin of the point-vector will be used for projection. The vector portion

of the original point-vector will not be used. The vector portion will be always normal to the projected surface after projection.

- If the optional parameter “vector” or “ATANGL,start,(end),surf” is not specified, the projection will be along a vector normal to the specified surface.
- If the optional parameter “NOWRAP”, “WRAP” or “REVOLV” is specified, the default is “NOWRAP”.
- If the optional parameter “START”, “MIDDLE”, “END”, “CENTER” or “AT,point” is not specified, the default is “START”.
- If “NOWRAP” is specified, the internal generated points on the specified PATTERN will be projected individually onto the surface. This will create a distorted pattern. The shape or the size of the pattern will not be maintained.
- If “WRAP” is specified, the user can specify an optional “ATTACH” method, either START, MIDDLE, END, CENTER, or “AT,point”. Distances and angles from the attach point are maintained. This will approximately maintain the shape and size based at the attach point.
- “REVOLV” and “RADIAL” are only applicable for projecting onto cones, cylinders, or surface of revolution. Distances between the projected points are not preserved for “RADIAL”. This will approximately maintain the shape, but not the size. Distances between the projected points are preserved for “REVOLV”. This will approximately maintains the size, but not the shape. There will be no difference for “REVOLV” and “RADIAL” if the surface is of the type “cylinder”. See REVOLV Projection Calculation Method and RADIAL Projection Calculation Method for detail.
- An interpolation surface can be used instead of a vector. This will smoothly interpolate the projecting vector away (perpendicular to the direction of travel through the generated points) from the normal connecting the original curve and the surface. The interpolation surface acts like a weighting function, causing more or less rotation away from the direction of travel.
- The extension of the specified surface will be used if required. The underlying surface will be used if the surface specified is of the type **TRIMMED**.

Error Conditions:

- If “REVOLV” or “RADIAL” is specified and the surface of projection is not a cone, a cylinder or a surface of revolution.

- The projection vector does not intersect the surface of projection when the vector is base at the projection point.
- The optional near-point or near-pntvec is not closed to the desired surface position.
- If a **NET surface** is specified as the surface of projection.

RADIAL Projection Calculation Method:

1. The attached point “PT1” specified by “START, MIDDLE, END, CENTER, or At,point” projects onto the surface “SF1” according to the method specified, i.e. along a specific vector, at an angle or normal to the surface “SF1”.
2. This projected attached point “PT2” becomes the reference point on the surface “SF1”.
3. A reference circle normal to the revolving axis, with its center along the revolving axis and its circumference passing through the reference point “PT2” is created internally. The radius of this reference circle is “R1”.
4. A reference plane “PL1” which contains the revolving axis and the attach point projection vector is created internally.
5. The normal distance “L1” of a point “PT(n)” which is an individual entity on the pattern “PN1” to the reference plane “PL1” is calculated.
6. The point “PT(n)” is projected normal to the reference plane “PL1”. The distance “L2” between the point “PT(n)” and the attached point “PT1” is calculated.
7. A point “PT3” is created internally at a distance “L2” along the slope of the revolving surface “SF1” on the reference plane “PL1” in the same side as the point “PT(n)”.
8. A circle “CI1” normal to the revolving axis, with its center along the revolving axis and its circumference passing through the point “PT3” is created internally. The radius of this circle is “R2”.
9. A point “PT4” on the circle “CI1” along an arc distance away from the reference plane “PL1” in the same side of the point “PT(n)” on the pattern “PN1” is created internally. The location of the point “PT4” is where the point “PT(n)” on the curve projected to the revolved surface “SF1”.
10. This arc distance is calculated according to the following equation.

$$\text{arc distance} = L1 * R2 / R1$$

11. Repeat Step 5 through Step 10 for all the points on the pattern “PN1” until done.

12. All the projected points created internally are joined together to form the resultant projected pattern on the surface.
13. If the original PATTERN is of the type POINT-VECTOR, a vector normal to the projected surface for each of the projected points will be added to the projected point to create a projected point-vector pattern.

REVOLV Projection Calculation Method:

1. The attached point "PT1" specified by "START, MIDDLE, END, CENTER, or At,point" projects onto the surface "SF1" according to the method specified, i.e. along a specific vector, at an angle or normal to the surface "SF1".
2. This projected attached point "PT2" becomes the reference point on the surface "SF1".
3. A reference plane "PL1" which contains the revolving axis and the attach point projection vector is created internally.
4. The normal distance "L1" of a point "PT(n)" which is an individual entity on the pattern "PN1" to the reference plane "PL1" is calculated.
5. The point "PT(n)" is projected normal to the reference plane "PL1". The distance "L2" between the point "PT(n)" and the attached point "PT1" is calculated.
6. A point "PT2" is created internally at a distance "L2" along the slope of the revolving surface "SF1" on the reference plane "PL1" in the same side as the point "PT(n)".
7. A circle "CI1" normal to the revolving axis, with its center along the revolving axis and its circumference passing through the point "PT2" is created internally. The radius of this circle is "R2".
8. A point "PT3" on the circle "CI1" along an arc distance away from the reference plane "PL1" in the same side of the point "PT(n)" on the pattern "PN1" is created internally. The location of the point "PT3" is where the point "PT(n)" on the curve projected to the revolved surface "SF1".
9. Repeat Step 4 through Step 8 for all the points on the pattern "PN1" until done.
10. All the projected points created internally are joined together to form the resultant projected curve on the surface.
 - If the original PATTERN is of the type POINT-VECTOR, a vector normal to the projected surface for each of the projected points will be added to the projected point to create a projected point-vector pattern.

3.4.9 A Pattern That Randomly Combines Either Points And Point-Patterns, Or Point-vectors And Point-Vector-Patterns

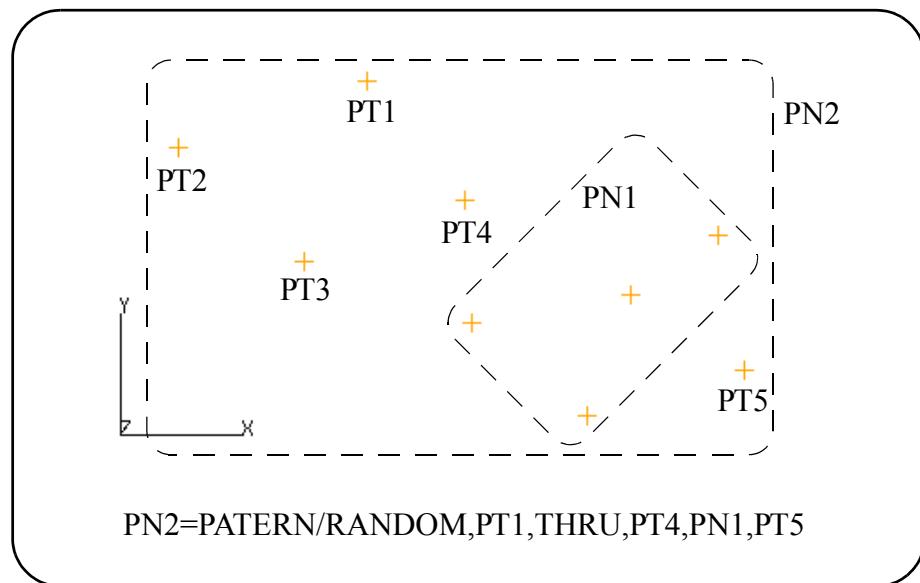
Command Syntax:

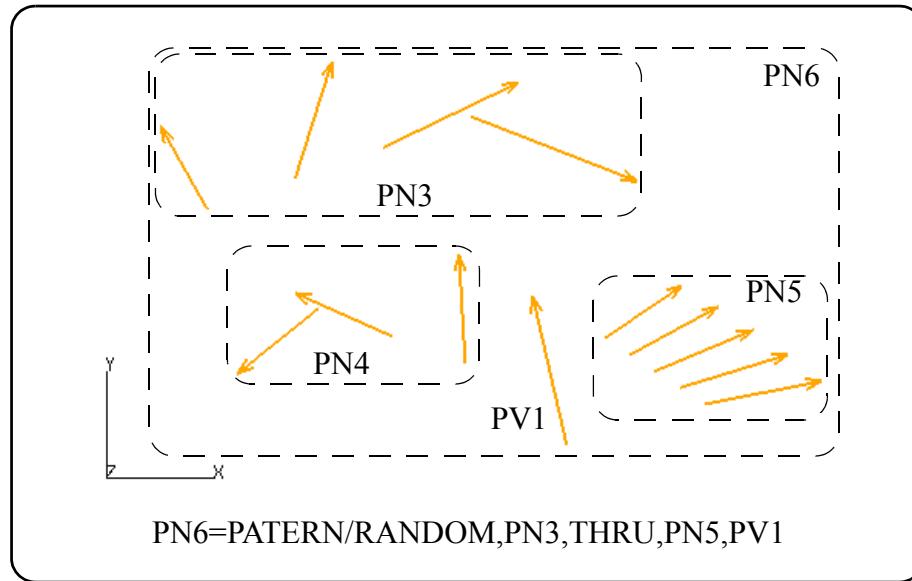
```
PATERN/RANDOM,point    [ [,THRU],point ] [...]
                           pt-patern           pt-patern
```

or

```
PATERN/RANDOM,pntvec   [ [,THRU],pntvec ] [...]
                           pv-patern           pv-patern
```

Icon Menu Sequence:





This definition generate a pattern which compose of all the individual entities and/or patterns of the same geometry types. The order of the individual entities in the generated pattern follows the original order of how they were specified in the definition.

Error Condition:

- The individual entity and the pattern must be of the same geometry type. No point and point-vector pattern, or point-vector and point pattern are allowed to specify in the same definition.

PLANES

The following list gives an abbreviated notation of all the valid PLANE definition formats. See the individual sections for a complete explanation of each format.

The syntax of all PLANE definitions allow the appending of an optional "display-point." This POINT will be used to define the center point of the dashed circle that is displayed for a PLANE. A nested POINT is acceptable. If the POINT does not lie on the PLANE, it will be projected onto the PLANE.

Some Example Plane Expressions:

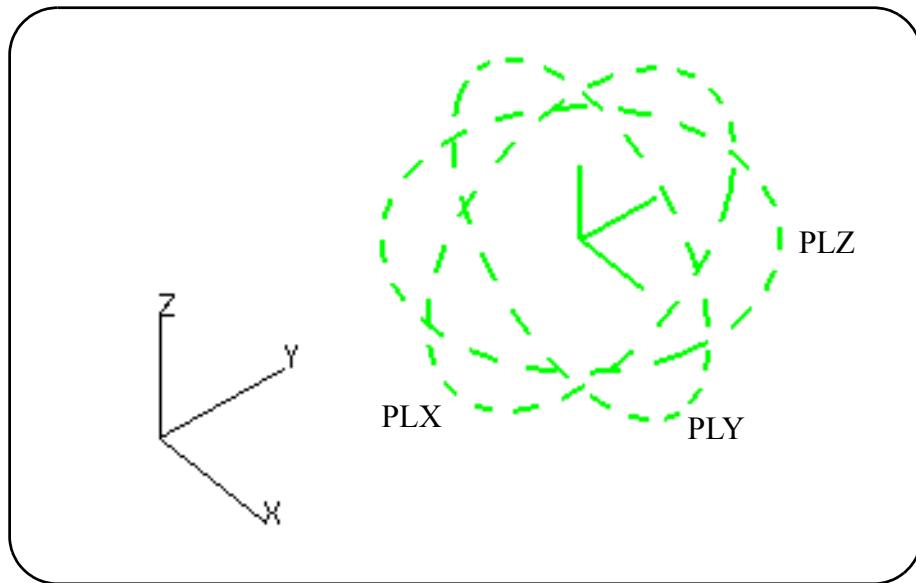
```
PLANE/0,0,1,16.345,PT3
PL/PTX(I),PTX(I+1),PTX(I+2)
PL/PV1,PV2,PV3,PV4
PLANE/PT1,PARREL,PLZ
PL/PV1,PA,PL2,PV2
PL/PARREL,PL4,ZSMALL,.5
PL/PT5,PERPTO,VE1
PL/PV1,PE,VE2,PV2
PLANE/PT1,PT2,PERPTO,PL1,PT1
PL/PV1,PV2,PE,PL3,PV4
PLANE/PT1,PERPTO,PL1,PL2
PL/PV1,PE,PL2,PL3,PV2
PLANE/LN1,PERPTO,PL1
PLANE/LN1,PT2
PLANE/SF2
```

3.5 PLANE Expressions

A plane is a geometric entity which is determined by three points in space, which do not lie in a straight line. An **NCL** plane is an unbounded analytical surface represented by the equation:

$$Ax + By + Cz = D$$

A, B and C are the I, J, K components of a unit vector which is perpendicular to the plane and D represents the directed distance, along the vector, from the coordinate system origin to the plane. A negative D value (-D) indicates that the distance is measured in the opposite direction along the vector.



Canonical Form:

`PLANE/I,J,K,D`

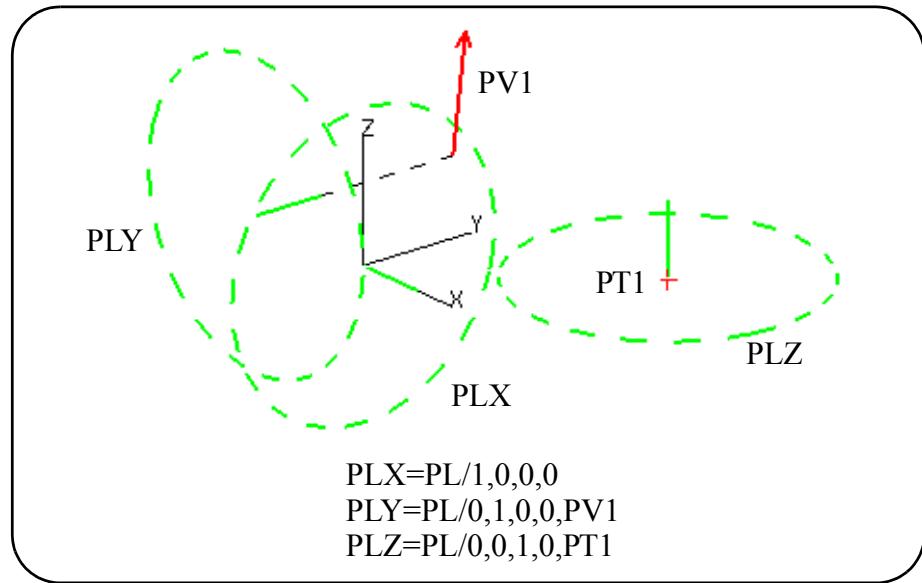
NOTE: A PLANE will be displayed as a dashed circle with a perpendicular vector. An optional near point or a point-vector can be included to place the displayed plane. If this near point or the point-vector origin is not on the plane, the plane will display in such a way that the near point or the point-vector origin will be along the normal vector emanating from the center of the plane symbol. **Green** is the **NCL** System default color for planes.

3.5.1 A Plane By The Normal Vector Components And An Offset Value With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/i,j,k,d[,display-point ]
    pntvec
```

Icon Menu Sequence:



Calculation Method:

- This format defines a plane to be normal to the vector (i, j, k) at a distance from the origin. The distance is along the i, j, k vector with an amount calculated as:

$$\text{distance} = d/\sqrt{i^2 + j^2 + k^2}$$

- If the optional display point/point-vector is not specified, the plane symbol display will be centered at the calculated distance from the origin of the

“current” REFERENCE SYSTEM along a direction specified by the “i, j, k” vector.

Error Condition:

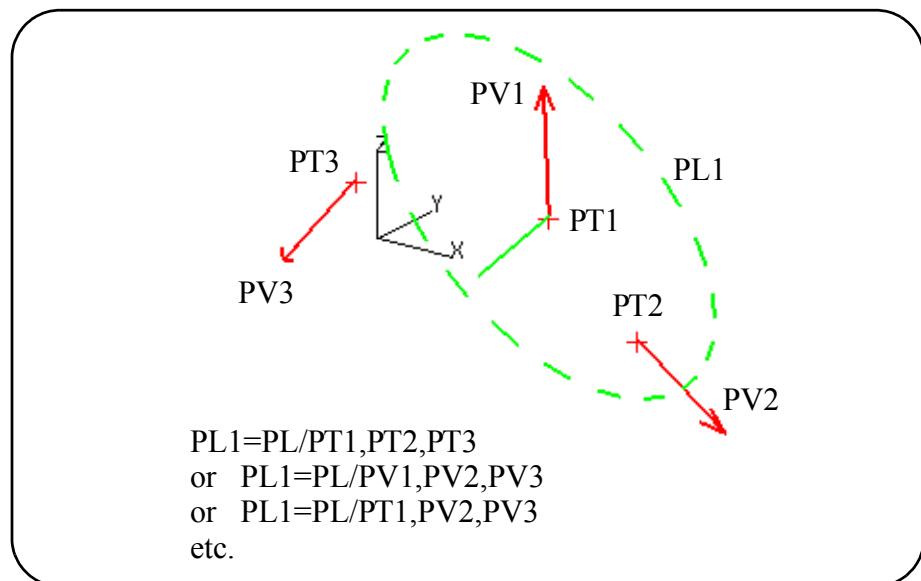
- The magnitude of the vector normal to the plane may not be zero.

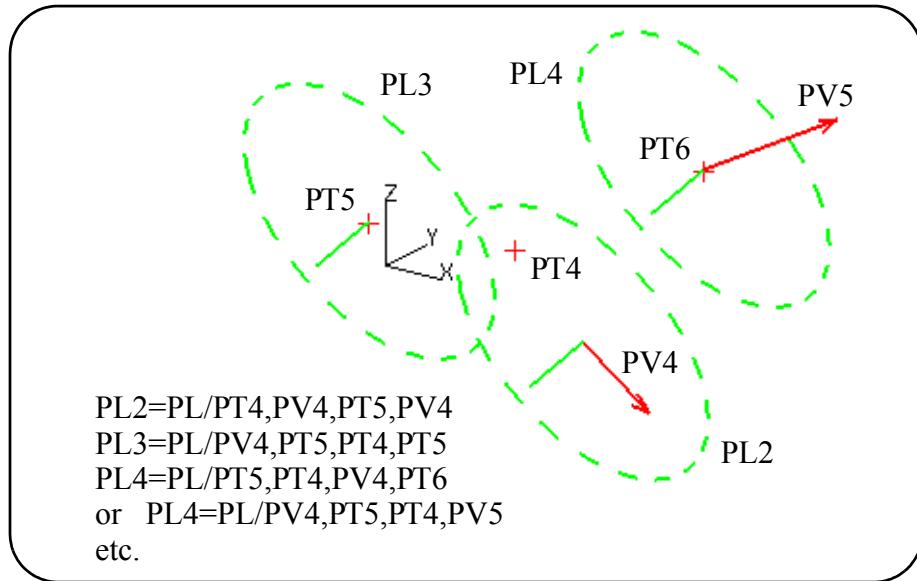
3.5.2 A Plane Through Three Geometric Entities In Any Combination Of Points Or Point-Vectors With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/point ,point ,point [,display-point ]
          pntvec pntvec pntvec
```

Icon Menu Sequence:





NOTE: All the planes in above two figures have the same physical properties even they are defined differently.

Calculation Method:

- The plane will pass through the three referenced points and/or point-vector origins.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at the first of the three given points or the point-vector origins.

Error Condition:

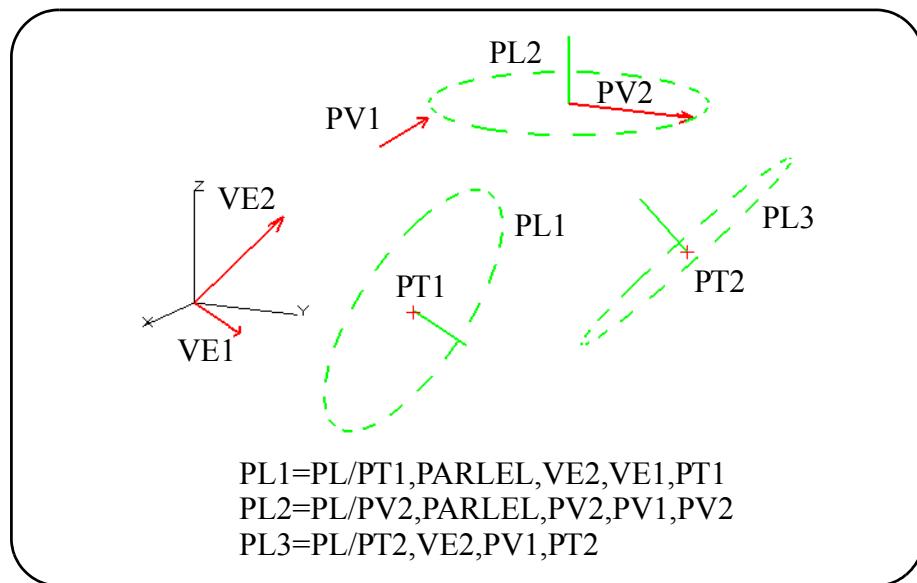
- The three referenced points and/or point-vectors origin must not be colinear.

3.5.3 A Plane Through A Point/Point-Vector And Parallel To Two Geometric Entities In Any Combination Of Points Or Point-Vectors With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/point , PARALLEL,vector,vector[,display-point ]
          pntvec           pntvec           pntvec
```

Icon Menu Sequence:



Calculation Method:

- The pair of vectors (or the vector portions of the pair of point-vectors) will be used to create a temporary plane on which the pairs of vectors (or the vector portions of the pair of point-vectors) lie. The plane will be defined parallel to this temporary plane through the referenced point or the point-vector origin.

- The normal direction of the created plane will be according to the right hand rule, i.e. from the first vector/point-vector specified to the second vector/point-vector specified.

Error Conditions:

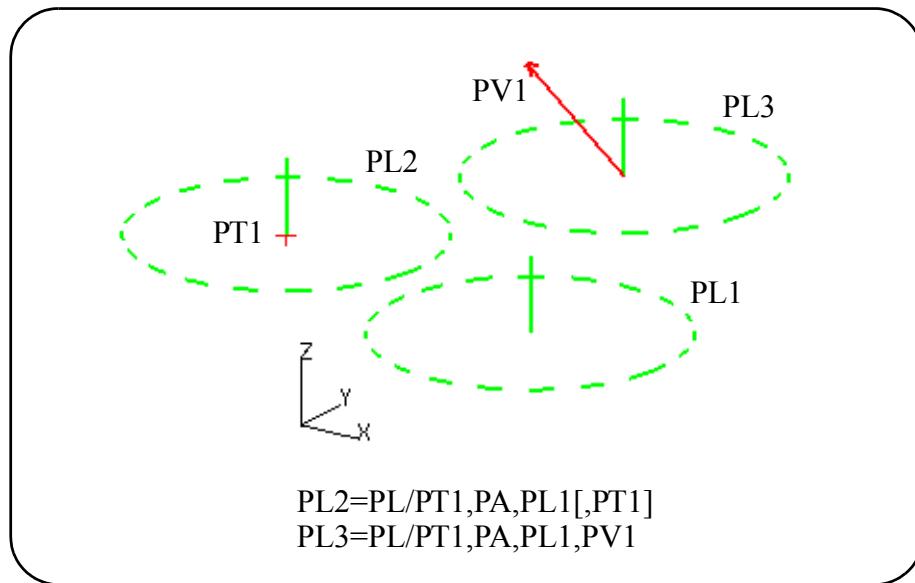
- The pair of vectors (or the pair of point-vectors) cannot be paralleled to each other.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at the calculated distance from the origin of the “current” REFERENCE SYStem along a direction specified by the “i, j, k” vector of the created plane.

3.5.4 A Plane Through A Point/Point-Vector And Parallel To A Plane With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/point , PARREL,plane[,display-point ]
          pntvec                      pntvec
```

Icon Menu Sequence:



NOTE: PL2 and PL3 physically are the same plane but displayed at different location.

Calculation Method:

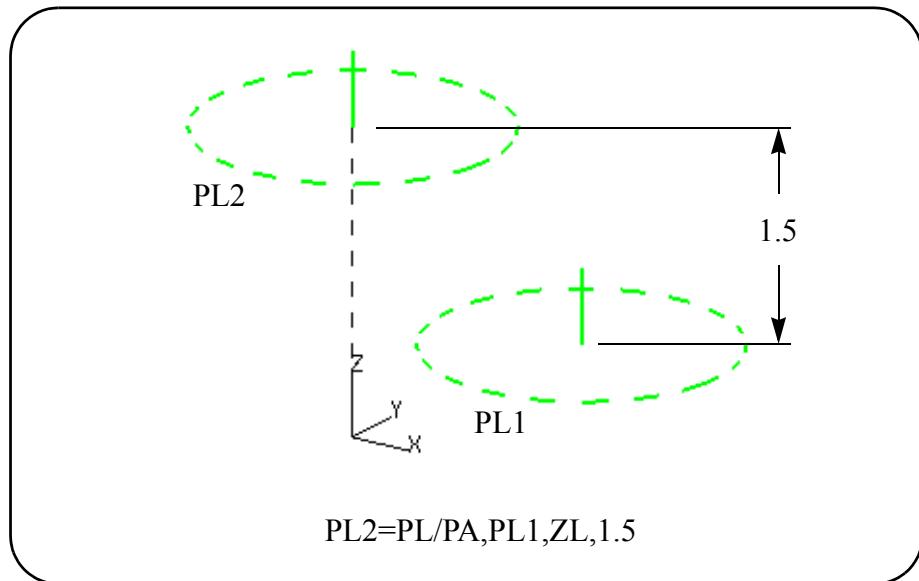
- The plane will be defined parallel to the referenced plane through the referenced point or the point-vector origin.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at the defining point/point-vector origin.

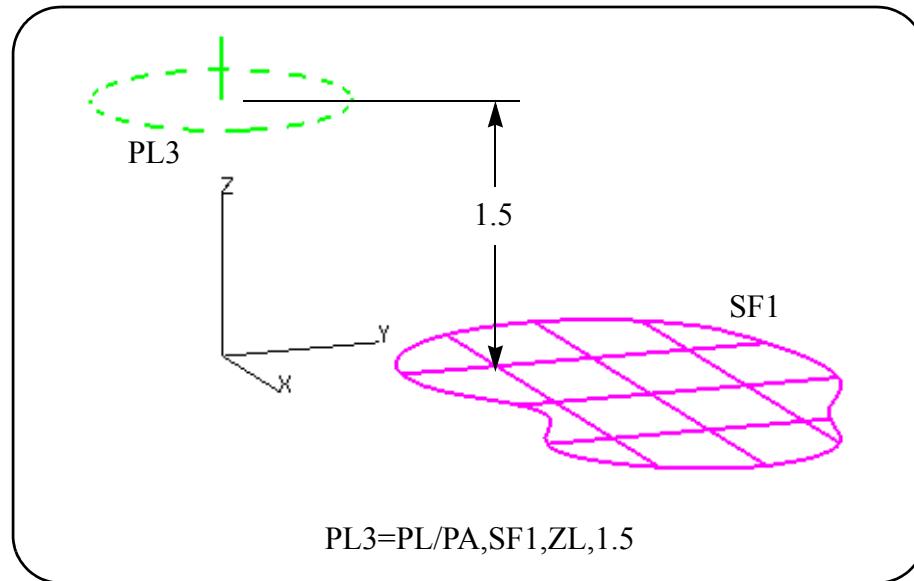
3.5.5 A Plane Parallel To A Plane/Planar-Surface At A Distance With An Optional Display Point/Point-Vector

Command Syntax:

```
XLARGE  
XSMALL  
YLARGE  
PLANE/PARREL,plane,YSMALL,dist[,display-point ]  
surf ZLARGE pntvec  
ZSMALL
```

Icon Menu Sequence:





Calculation Method:

- The new plane will be defined parallel to the specified plane or the planar surface at the specified distance in the direction as specified by the modifiers **XLARGE**, **XSMALL**, **YLARGE**, **YSMALL**, **ZLARGE** or **ZSMALL**.
- The planar surface can be either of the type **NCL** or trimmed.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at a location along the vector normal to the plane (i.e. the vector generated by the i, j, k components of the plane) at a total distance of “D” from the origin of the “current” **REFerence SYStem**. Here “D” is defined as:

$$D = \text{specified distance} + \\ \text{the unitized “d” canonical component of the specified plane}$$

Error Condition:

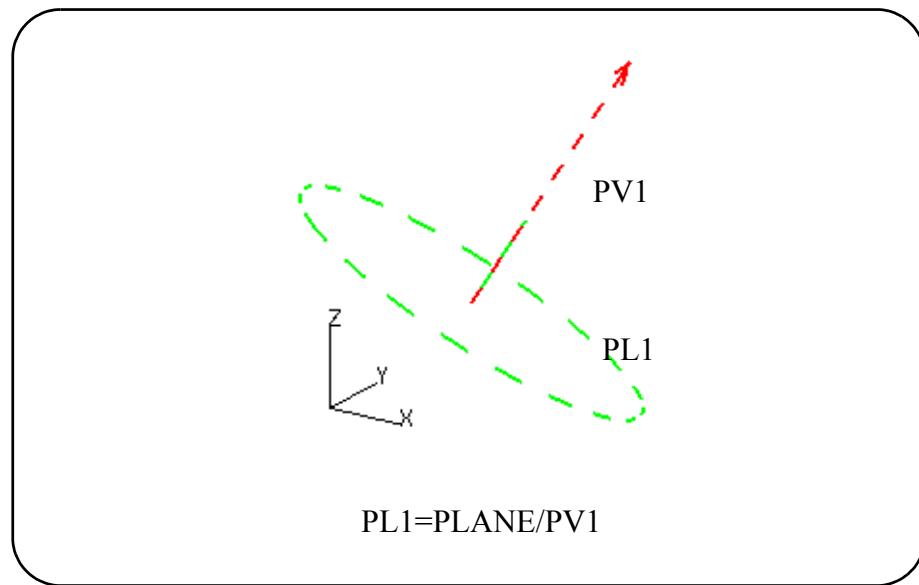
- The referenced surface must be of the primitive type: “PLANAR”.

3.5.6 A Plane Perpendicular To A Point Vector And Through Its Origin With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/pntvec[,display-point ]
           pntvec
```

Icon Menu Sequence:



Calculation Method:

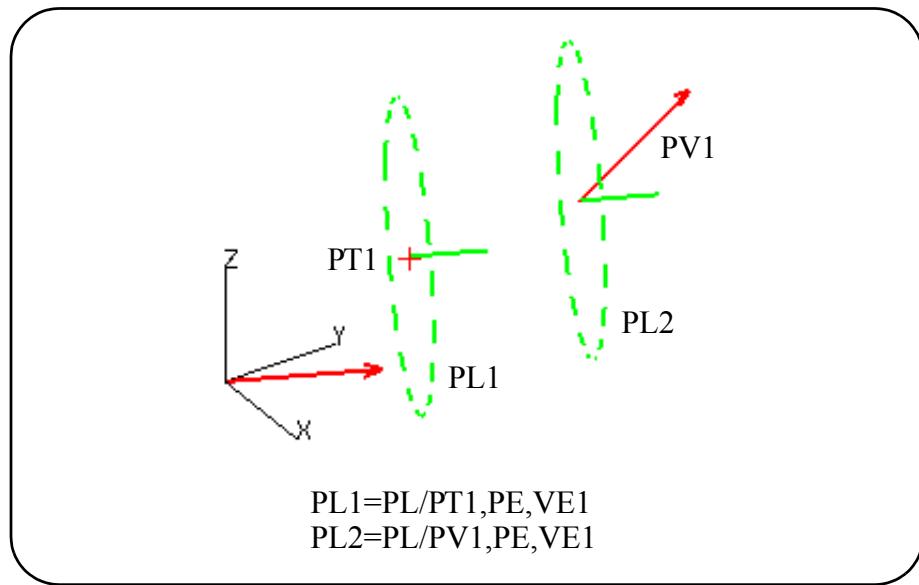
- The plane will be defined perpendicular to the referenced point-vector through the referenced point-vector origin.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at the point-vector origin.

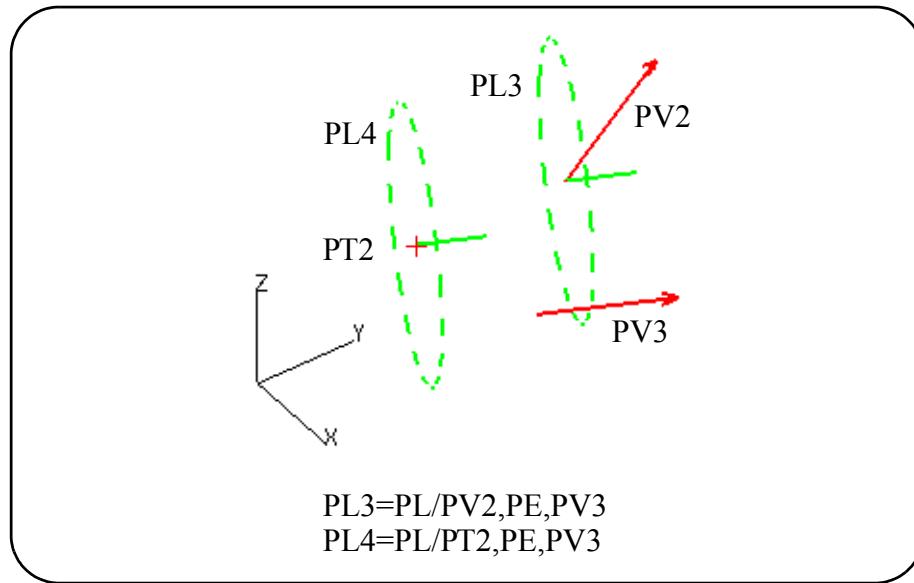
3.5.7 A Plane Through A Point/Point-Vector And Perpendicular To A Vector/Point-Vector With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/point , PERPTO, vector[,display-point ]  
          pntvec           pntvec           pntvec
```

Icon Menu Sequence:





Calculation Method:

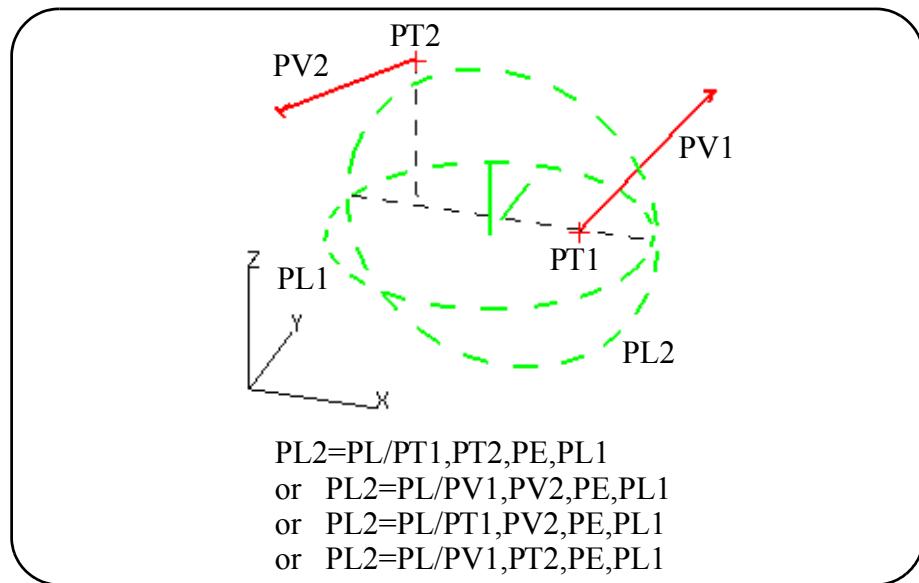
- The plane will be defined passing through the referenced point/point-vector origin and perpendicular to the referenced vector/point-vector.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at the end of the defining point/point-vector origin.

3.5.8 A Plane Perpendicular To A Plane And Through Two Entities Which Can Be Points And/Or Point-Vectors With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/point ,point ,PERP TO,plane[,display-point ]
          pntvec pntvec
```

Icon Menu Sequence:



Calculation Method:

- The plane will pass through the two referenced points or the point-vectors origins and will be perpendicular to the referenced plane.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at the first of the two given points.

Error Condition:

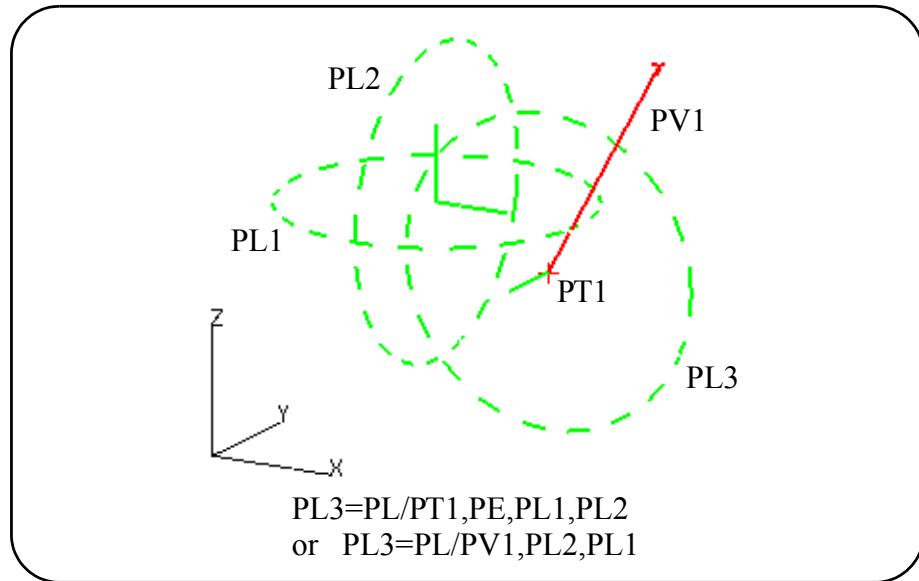
- The referenced points/point-vectors origins may not be coincident.

3.5.9 A Plane Through A Point Or Point-Vector And Perpendicular To Two Intersecting Planes With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/point , PERPTO,plane,plane[,display-point ]
          pntvec
```

Icon Menu Sequence:



Calculation Method:

- The plane will be calculated which contains the referenced point or the point-vector origin and is perpendicular to the two referenced planes.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at the first given point.

Error Condition:

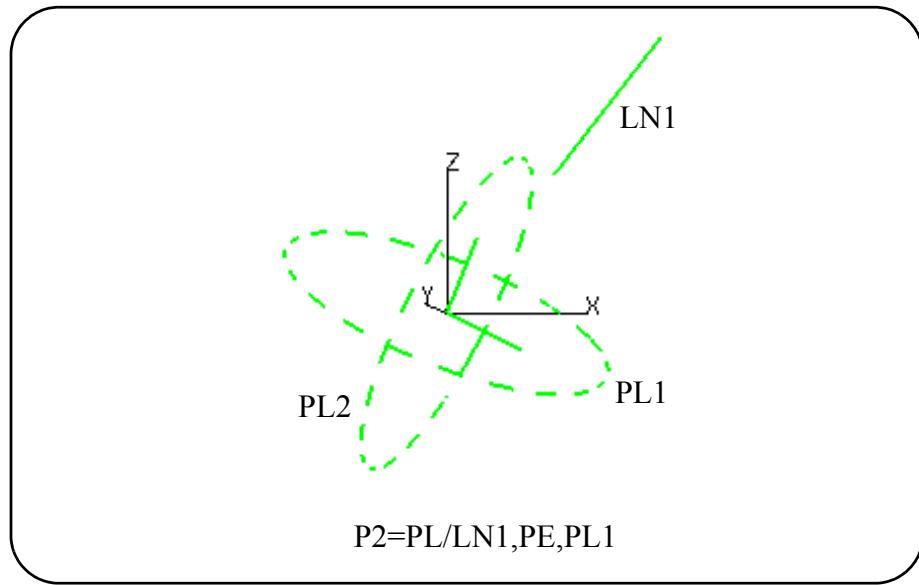
- The two referenced planes must intersect as an unbounded straight line.

3.5.10 A Plane Which Contains A Line And Perpendicular To A Plane With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/line [, PERPTO, plane] [, display-point ]
            pntvec
```

Icon Menu Sequence:



NOTE: If the optional [, PERPTO, plane] couplet is not specified, then the system will use the DEFAULT which is the xy-plane of the "current" REFERENCE SYStem.

Calculation Method:

- The plane will contain the start point and end point of the referenced line and will be perpendicular to the referenced plane.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at a location along the vector normal to the

calculated plane (i.e. the vector generated by the i, j, k components of the plane) at a distance of the unitized “d” canonical component of the calculated plane from the origin of the “current” REference SYStem.

Error Condition:

- The referenced line cannot be perpendicular to the referenced plane.

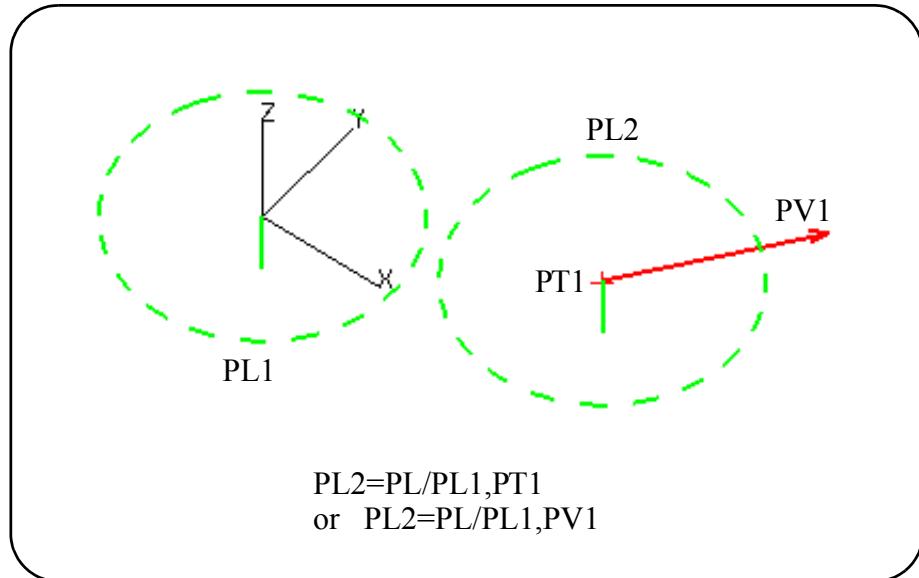
3.5.11 A Plane By Cloning Itself With An Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/plane[,display-point ]
            pntvec
```

Icon Menu Sequence:

No corresponding Icon Menu.



Calculation Method:

- The plane created will be the same as the original plane.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at a location along the vector normal to the calculated plane (i.e. the vector generated by the i, j, k components of the plane) at a distance of the unitized “d” canonical component of the calculated plane from the origin of the “current” REFERENCE SYStem.

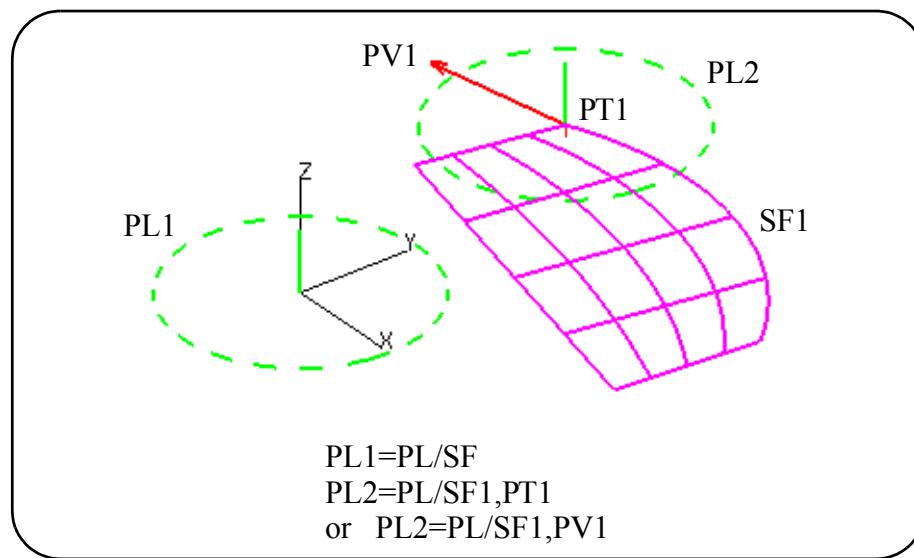
3.5.12 A Plane From A Planar Surface With Optional Display Point/Point-Vector

Command Syntax:

```
PLANE/surface[,display-point ]
           pntvec
```

Icon Menu Sequence:

No corresponding Icon Menu



Calculation Method:

- The plane will contain the referenced surface.
- The surface can be either of the type **NCL** or trimmed.
- If the optional display point/point-vector is not specified, the plane symbol display will be centered at a location along the vector normal to the calculated plane (i.e. the vector generated by the i, j, k components of the plane) at a distance of the unitized “d” canonical component of the calculated plane from the origin of the “current” **REFerence SYStem**.

Error Condition:

- The referenced surface must be of the primitive type: “PLANAR”

3.5.13 .An Optimal Plane From A Curve or A Surface

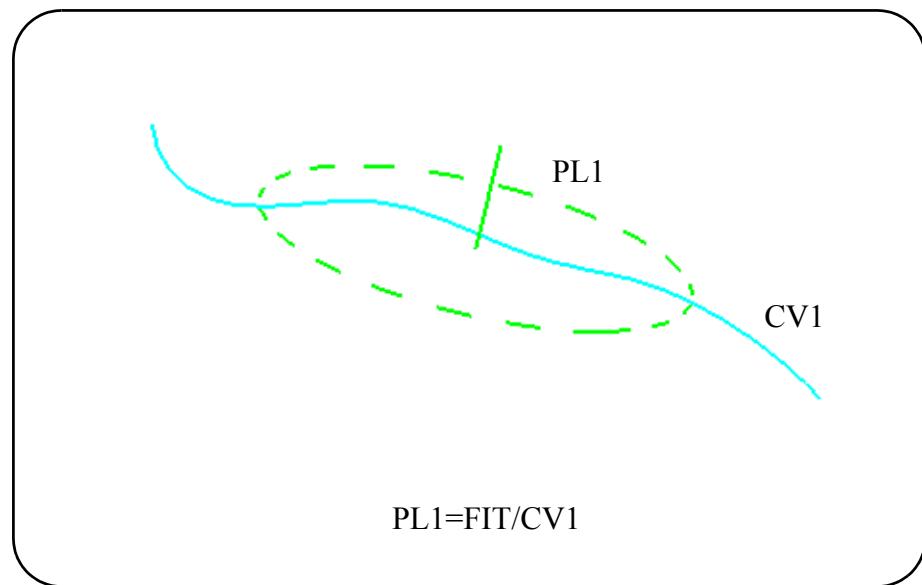
Command Syntax:

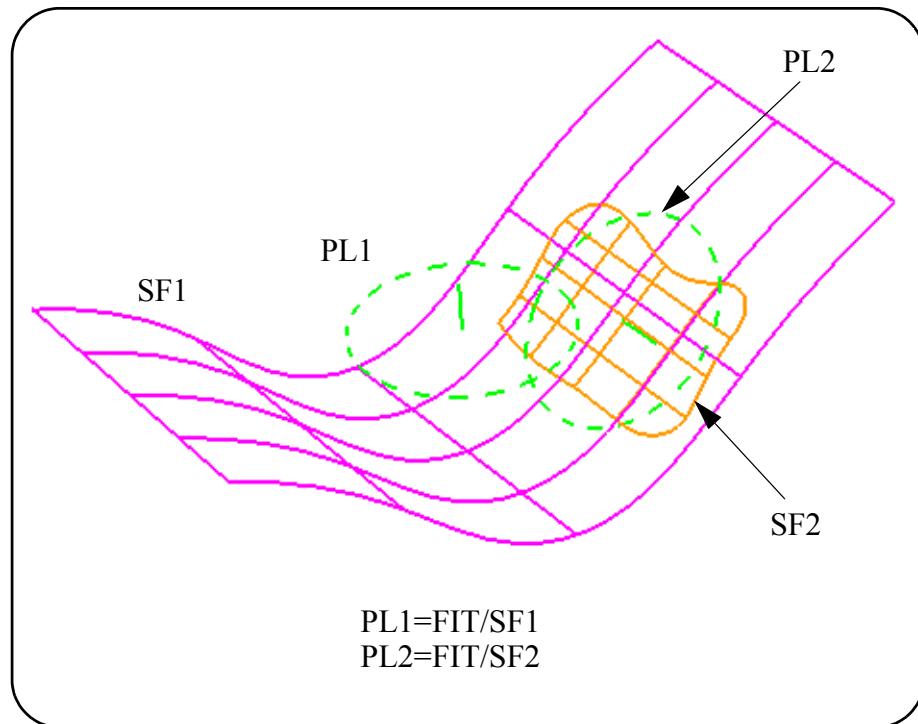
```
PLANE/FIT,circle [,planar]  
          curve  
          surface
```

where:

planar An optional scalar holds a value of 1 if the curve or surface is planar, otherwise it holds a value of 0.

Icon Menu Sequence:





Calculation Method:

- If a trimmed surface is specified, only the trimmed surface itself will be considered to define the plane. The underlying parent surface will not be considered.

Error Condition:

- A straight curve is not allowed.

POINTS

The following list gives an abbreviated notation of all the valid POINT definition formats. See the individual sections for a complete explanation of each format.

1. `POINT/x,y,[z]`
2. `POINT/pntvec`
3. `POINT/TE`
4. `POINT/CENTER,circle`
5. `POINT/CENTER,curve
surface`
6. `POINT/point [,Dx[,Dy[,Dz]]]
pntvec`
7. `POINT/point ,DELTA,scale1,TIMES,vector
pntvec`
8. `POINT/INTOF,line ,line
pntvec pntvec`
9. `POINT/INTOF,line,plane
or POINT/INTOF,plane,line`
10. `POINT/INTOF,pntvec,plane`
11. `POINT/INTOF,plane,plane,plane`
12. `POINT/INTOF,circle,curve,near-point
plane pntvec`
13. `POINT/INTOF,curve,curve ,ALL
line near-point
pntvec pntvec
circle
plane`
14. `POINT/INTOF3,curve ,curve [,ALL]
line line near-point
circle circle pntvec`

15. **POINT**/direction,ENDPT,circle
curve
line
pntvec
16. **POINT**/direction,INTOF,circle,circle
line
pntvec
17. **POINT**/circle,ATANGL,angle
18. **POINT**/point ,circle,angle
pntvec
19. **POINT**/line ,distance[,start-pt]
circle
curve
20. **POINT**/ON,circle, per
line
curve [PERCNT,]
21. **POINT**/ON,surf,[PERCNT,]uper,vper
22. **POINT**/patern,id-number
23. **POINT**/PROJCT,point [,vector],surface[**[u,v]**] \$
pntvec pntvec solid
[,near-point]
pntvec

Example Point Expressions:

```
POINT/0,0
PT/X,Y,Z
PT/PV1
POINT/TE
POINT/CENTER,CI1
POINT/INTOF,PL1,PL2,PL3
PT/YSMALL,INTOF,LN1,CI3
PT/YSMALL,INTOF,PV1,CI3
POINT/XLARGE,INTOF,CI2,CI15
POINT/INTOF,LN2,LN3
POINT/INTOF,PV1,PV2
PT/INTOF,PL9,CV33,PT5
POINT/CI3,ATANGL,35
PT/YL,INTOF,CI2,CI5
PT/INTOF,CI3,CV12,PT14
PT/PT4,CI22,35
PT/PT1,1.5,2,0
PT/XSMALL,ENDPT,CI1
PT/XS,ENDPT,PV1
PT/INTOF,LN1,PL1
PT/IO,PV1,PL1
PT/INTOF,PL1,CV1,PV3
PT/INTOF3,CV1,LN1
PT/IO,CI1,CV2,PV1
PT/PV1,CI2,22
PT/PV1,3,4,2.5
PT/CV1,1
PT/CV1,-3
PT/CV1,1,PT1
PT/ON,CV1,.5
PT/ON,SF1,.3,.6
PT/PN1,4
```

3.6 POINT Expressions

A point describes a unique position in three-dimensional space. It has no length, width or depth.

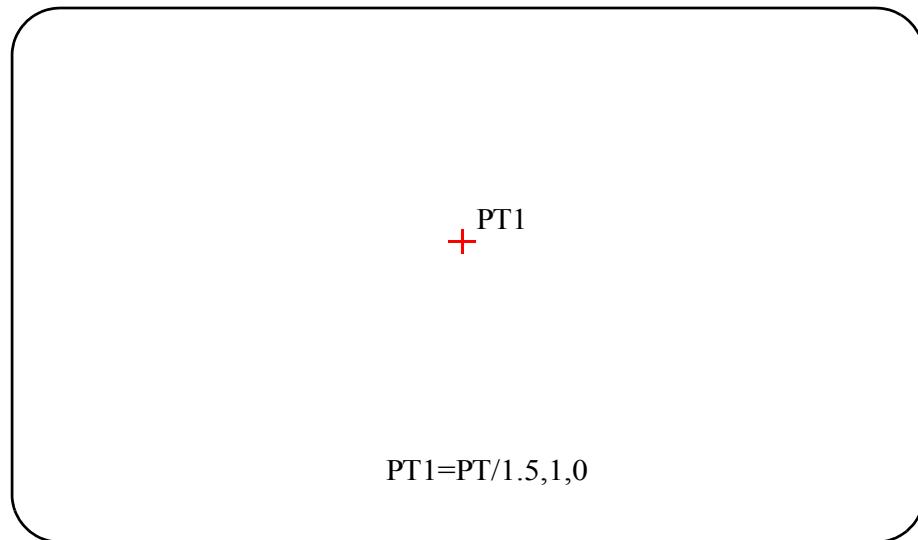
By Default, a point in **NCL** is displayed as a plus sign (+) in three-dimensional space. The current point display can be modified by using the following commands.

DRAFT/MODIFY=POINT,MARKER=marker_type

or

DRAFT/MODIFY=point_label_list,MARKER=marker_type

Valid marker types are: DOT, PLUS, STAR, CIRCLE, CROSS, TRIAN, DIMOND, SQUARE, DBLCIR, LRGDOT and CUBE.



Canonical Form:

POINT/X, Y, Z

The ZSURF statement applies to the point definition formats where a z-coordinate is not explicitly designated. In these cases, the z-coordinate is designated by the “ZSURF/plane” or “ZSURF/z-value” statement. If no **ZSURF** statement is in effect, **NCL** will use the XY-plane of the current **REFERENCE SYSTEM**.

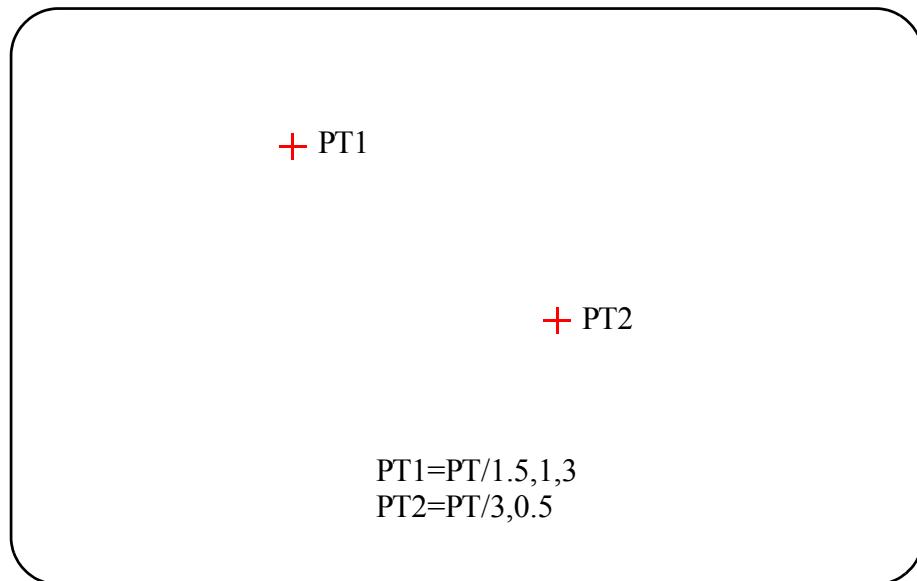
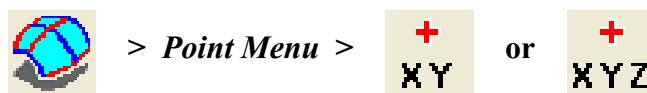
RED is the **NCL** System DEFAULT color for CAM POINTS.

3.6.1 A Point By Its Rectangular Coordinates

Command Syntax:

```
POINT/x-coordinate,y-coordinate,[z-coordinate]
```

Icon Menu Sequence:



Calculation Method:

- A unique point in three-dimensional space will be defined if the z-coordinate is specified.
- The z-coordinate is calculated by projecting the xy-coordinates onto the "current plane or implied plane" as specified by the **NCL** statements "ZSURF/plane" or "ZSURF/z-value" if z-coordinate is not specified.
- If no **ZSURF** statement is in effect, **NCL** will use the System DEFAULT z-value which is Z=0 (the xy-plane of the current **REFERencE SYStem**) if z-coordinate is not specified.

Error Conditions:

- The minimum input is x and y-coordinates where any or all coordinate values may be positive (+), negative (-) or zero.
- If z-coordinate is not specified, the ZSURF plane must not be a plane which is parallel to the z-axis of the current REference SYstem since a z-value cannot be determined.

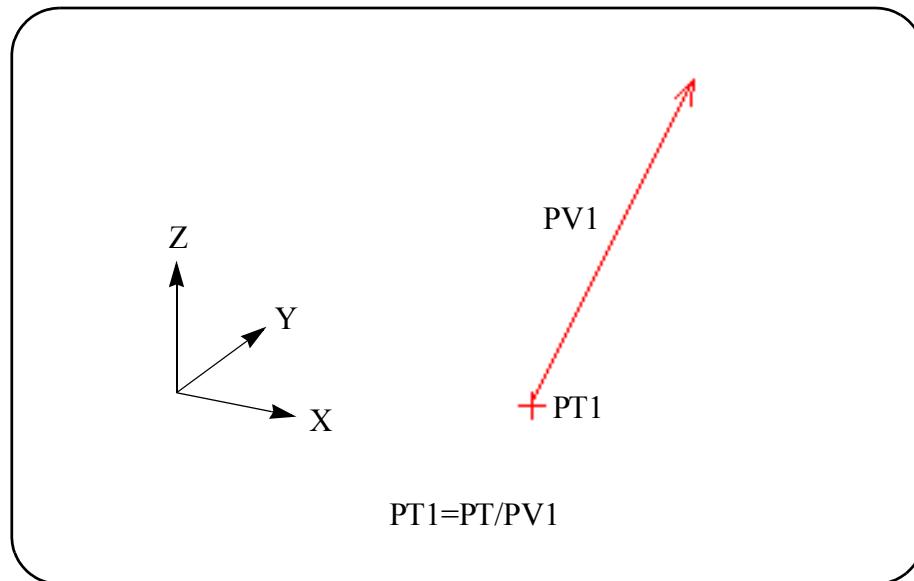
3.6.2 A Point At The Origin Of A Point-Vector In Three-Dimensional Space

Command Syntax:

POINT/point-vector

Icon Menu Sequence:

No corresponding Icon Menu.



Calculation Method:

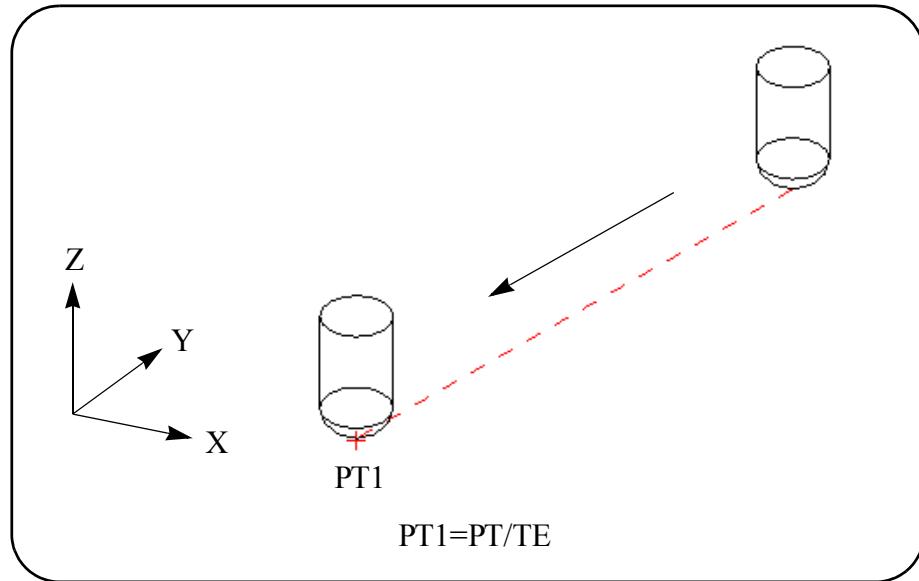
- This point definition format uses full three-dimensional rectangular coordinate data obtained from the origin of a point-vector to define a unique position in three-dimensional space.
- If “ZSURF/z-value” or “ZSURF/plane” is in effect, the z-coordinate of the new point will be determined by projecting the xy-coordinates of the new point onto the “current ZSURF plane or implied plane” as specified by the [ZSURF](#) statement.

3.6.3 A Point At The Current Cutter Location Or Tool End

Command Syntax:

POINT/TE

Icon Menu Sequence:



Calculation Method:

- This format defines a point at the current tool location using the 3D tool motion capabilities of **NCL**.

Error Condition:

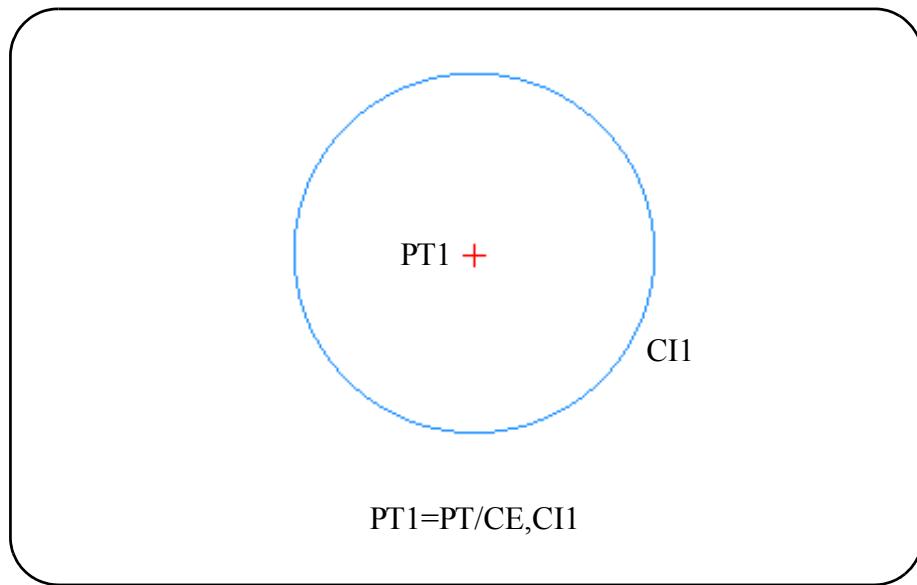
- If no tool motion has been specified, prior to the use of this statement, then **NCL** will use the System DEFAULT tool position which is X=0, Y=0, Z=0.

3.6.4 A Point At the Center Of A Circle

Command Syntax:

POINT/CENTER,circle

Icon Menu Sequence:



Calculation Method:

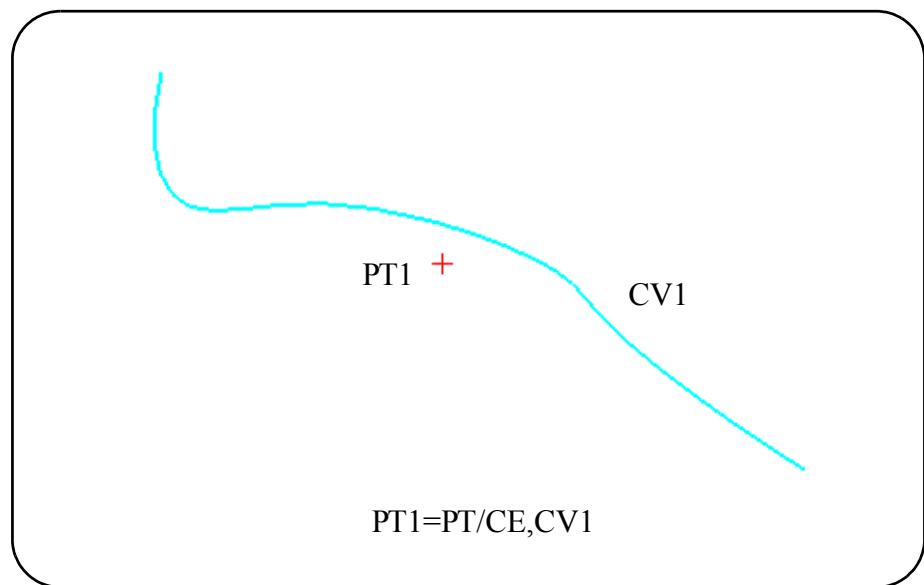
- **NCL** obtains the center location of the referenced circle from the canonical form of the referenced circle.
- The z-value of the point being defined will be the same as the z-value of the center point of the referenced circle, if **ZSURF** is not in effect. If ZSURF is in effect, the point's z-value will be determined by projecting the point's x and y-coordinates to the plane in a line perpendicular to the XY-plane and using the z-value of the point where the line pierces the ZSURF plane.

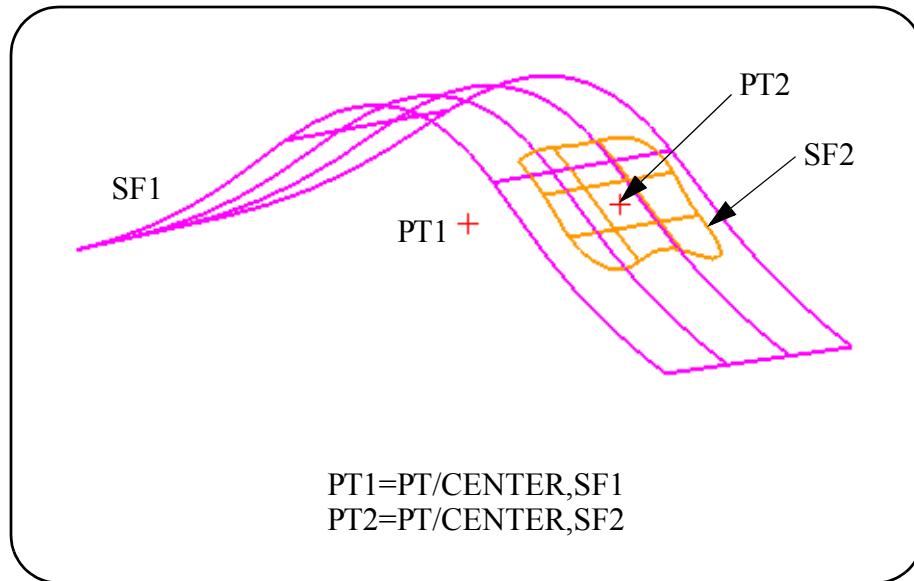
3.6.5 A Point At the Center Of Gravity Of A Curve Or Surface

Command Syntax:

```
POINT/CENTER,curve  
surface
```

Icon Menu Sequence:





Calculation Method:

- The point is at the center of gravity of the curve or surface.
- The trimmed surface itself will be considered for the center of gravity. The underlying parent surface will not be considered.

Error Condition:

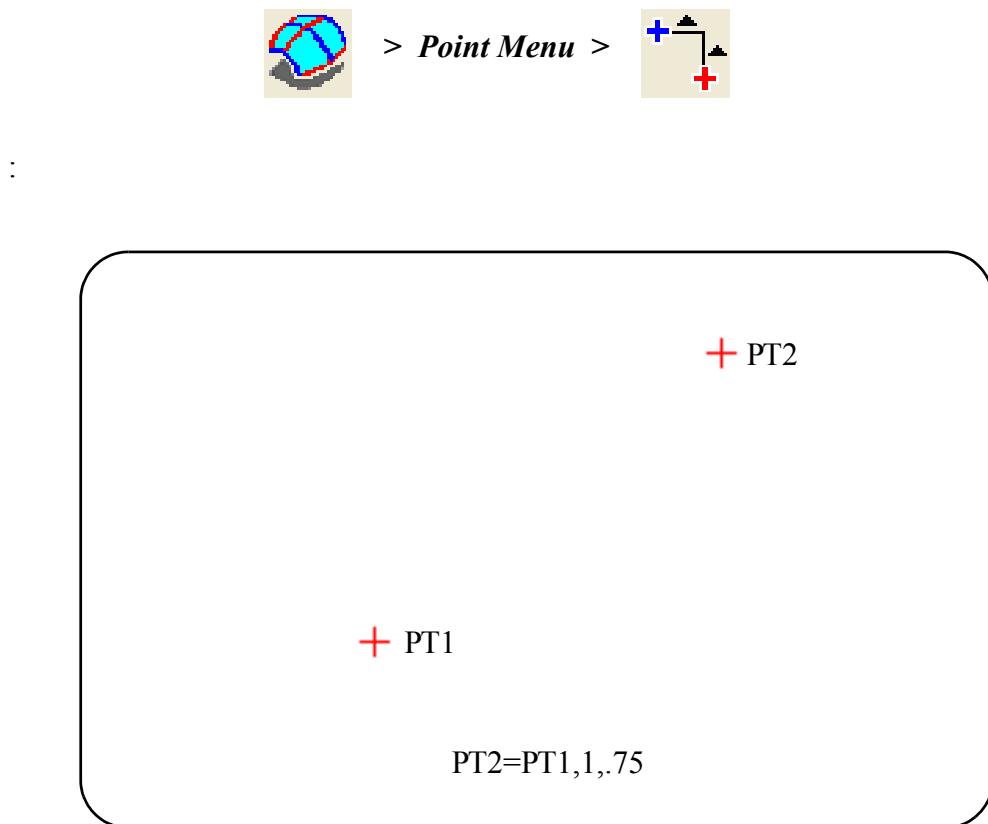
- A straight curve is specified.

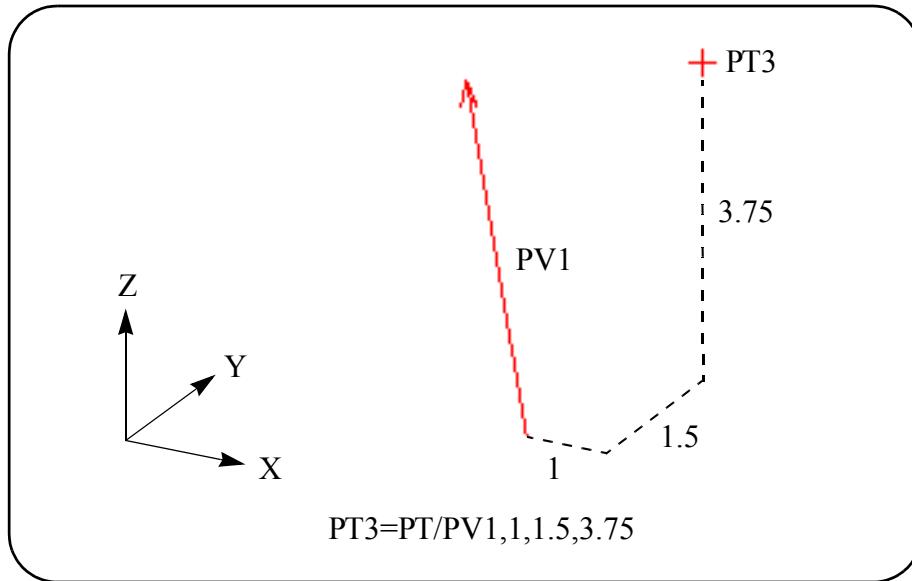
3.6.6 A Point By Its X, Y and Z Delta Displacements Relative To A Point Or A Point-Vector

Command Syntax:

```
POINT/point      [,delta-x[,delta-y[,delta-z]]]  
    point-vector
```

Icon Menu Sequence:





NOTE: The point is defined by its delta distance offsets from the referenced point or the point-vector origin. The delta values are optional. However, the fixed field format must be observed. Any delta offset may be omitted by specifying a zero (0) in the field.

Calculation Method:

- If the optional delta-x or delta-y value is zero (0), then the x-coordinate and/or y-coordinate of the new point will be the same as the referenced point or the origin of the referenced point-vector.
- If the optional delta-z value is omitted, the z-coordinate of the new point will be determined by projecting the xy-coordinates of the new point onto the "current **ZSURF** plane or implied plane" as specified by the **NCL** statements "ZSURF/ plane" or "ZSURF/ z-value".
- If "ZSURF/ NOMORE" is in effect, then the z-coordinate of the new point will be the same as the referenced point or the point-vector origin.

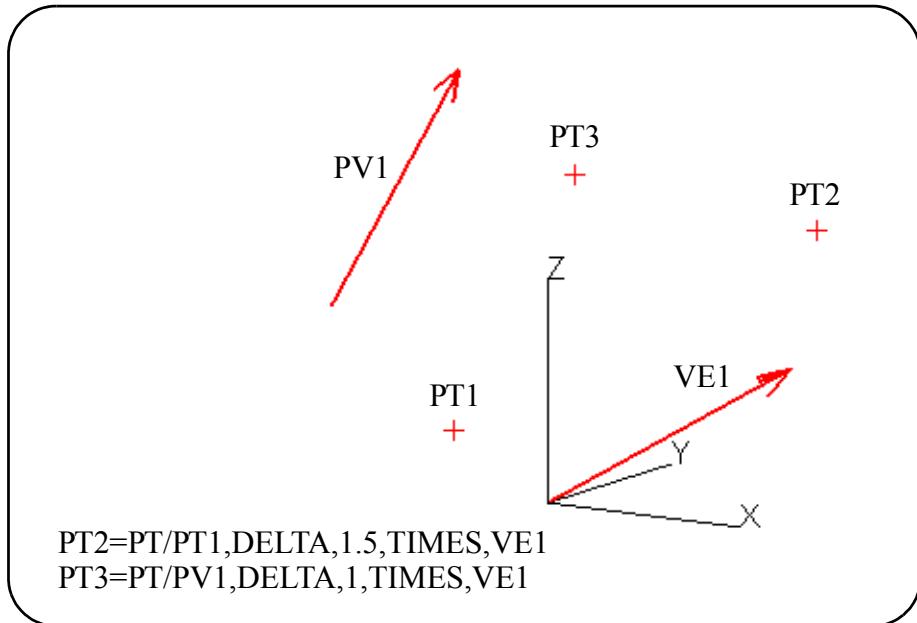
3.6.7 A Point By a Vectorial Displacement

Command Syntax:

```
POINT/point      , DELTA, scale1, TIMES, vector  
          point-vector
```

Icon Menu Sequence:

No corresponding Icon Menu



Calculation Method:

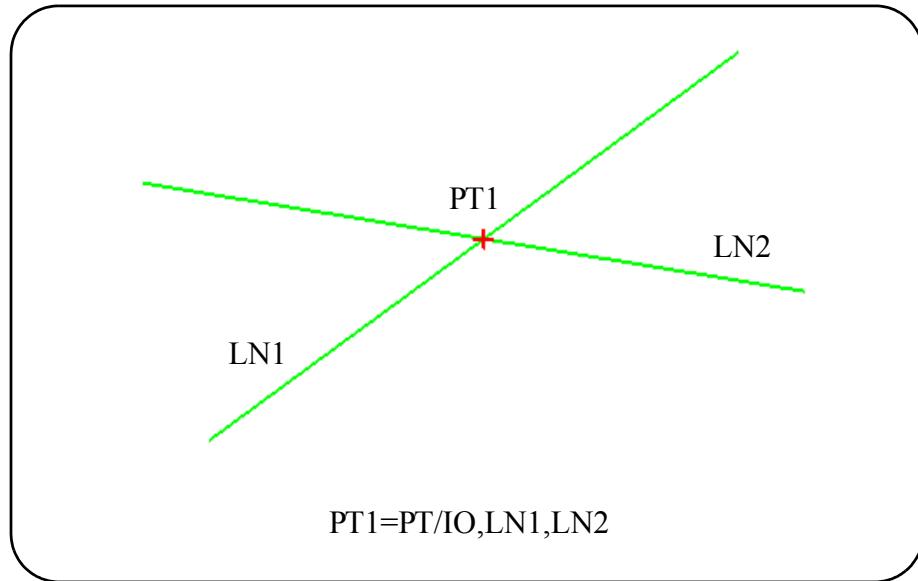
- The point is offset from the specified point in the specified vector direction at a distance equal to “scale1” times the magnitude of the specified vector.
- If a point-vector is specified, the origin of the point-vector will be used as the reference point.

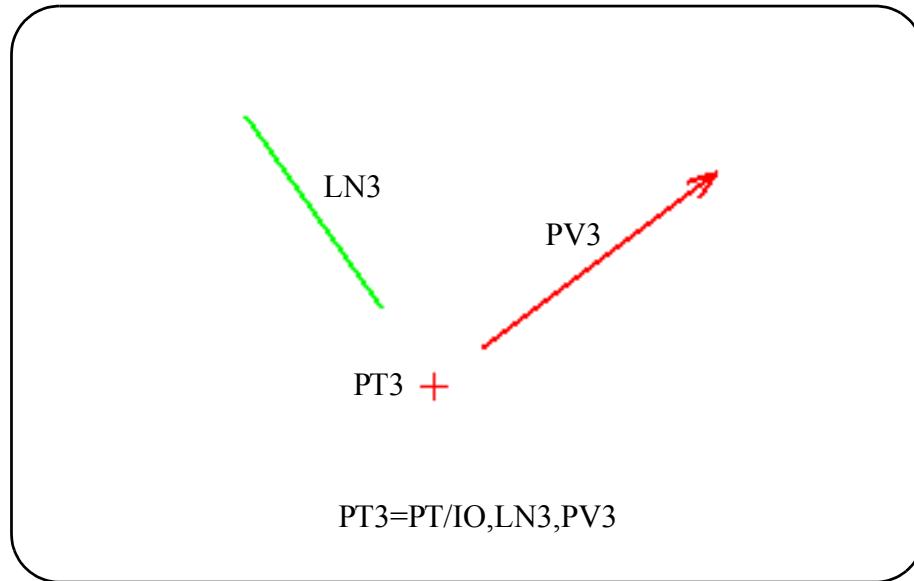
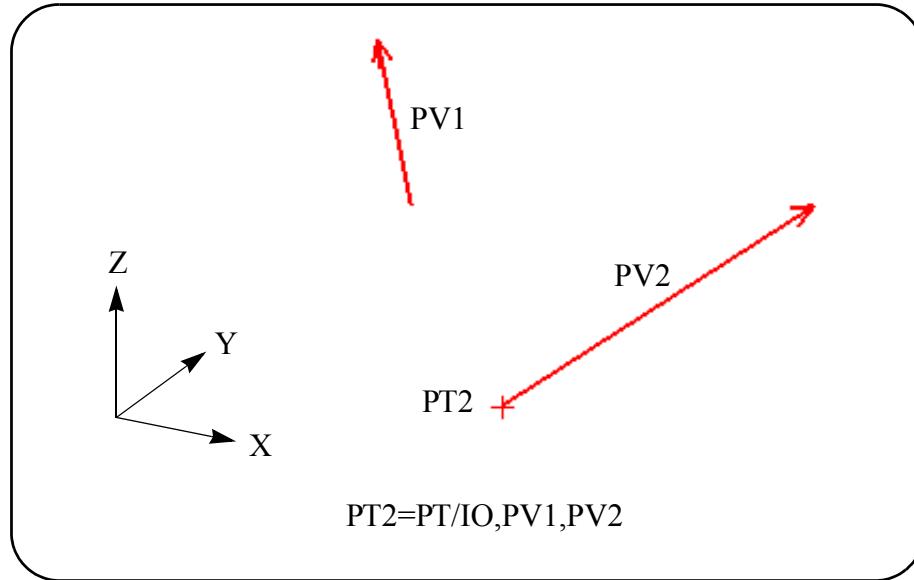
3.6.8 A Point At the Intersection Of Two Lines, Two Point-Vectors, Or One Line And One Point-Vector

Command Syntax:

```
POINT/INTOF, line , line  
          point-vector point-vector
```

Icon Menu Sequence:





Calculation Method:

- The lines/point-vectors are projected onto the XY-plane of the "current" REFERENCE SYSTEM followed by;
- The calculation of the intersection point using the extensions of one or both projected lines/point-vectors if necessary.

- The resultant point is projected onto the "current plane or implied plane" as specified by the **ZSURF** value in effect.
- If no ZSURF statement is in effect, **NCL** will use the System DEFAULT z-value which is Z=0 (the XY-plane of the current REference SYstem).

Error Conditions:

- The referenced lines/point-vectors must not be zero length when projected.
- The referenced lines/point-vectors must intersect after the projections have been performed.

3.6.9 A Point At the Intersection Of A Line And A Plane

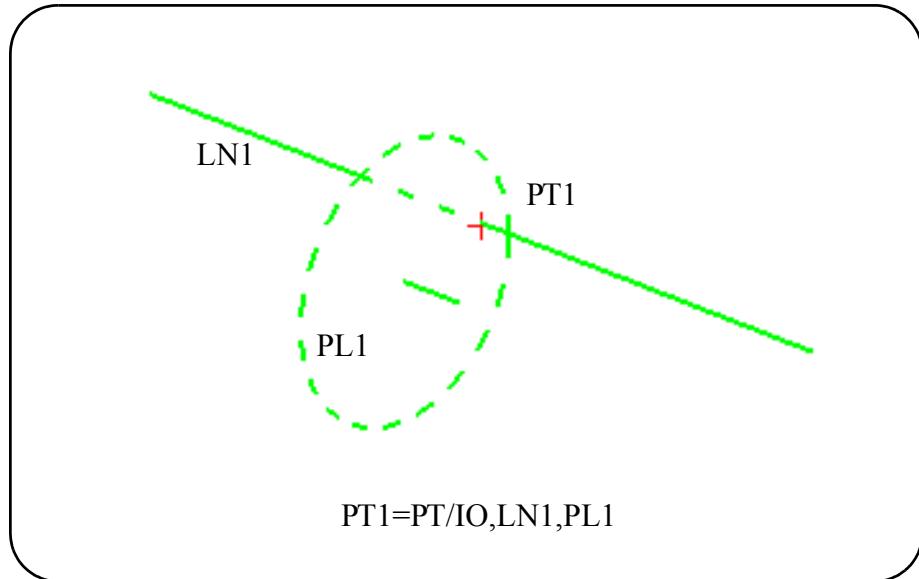
Command Syntax:

POINT/INTOF, line, plane

or

POINT/INTOF, plane, line

Icon Menu Sequence:



Calculation Method:

- **NCL** obtains the canonical data of the referenced line and plane and then calculates the pierce point.

Error Condition:

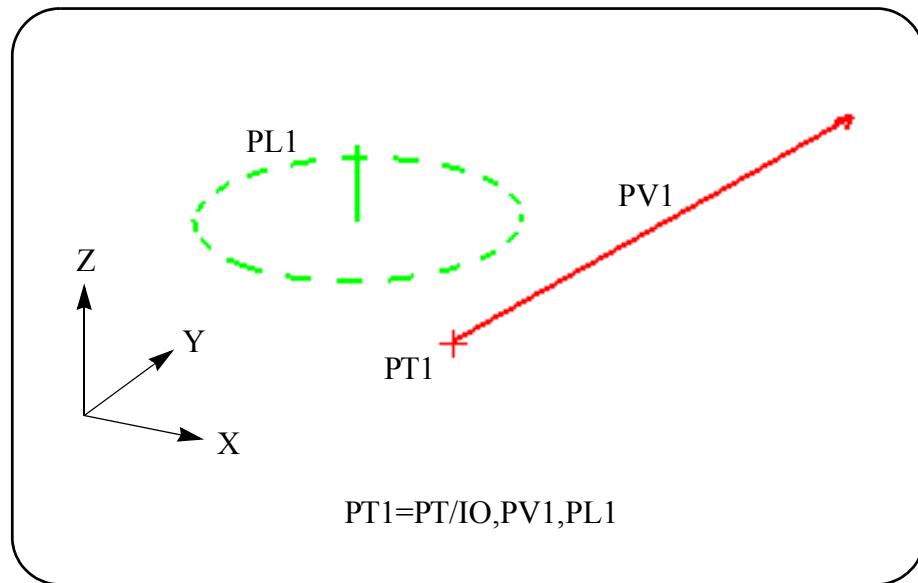
- The line must pierce the plane.

3.6.10 A Point At The Intersection Of A Point-Vector And A Plane

Command Syntax:

```
POINT/INTOF,point-vector,plane
```

Icon Menu Sequence:



Calculation Method:

- **NCL** obtains the canonical data of the referenced point-vector and plane and then calculates the pierce point.

Error Condition:

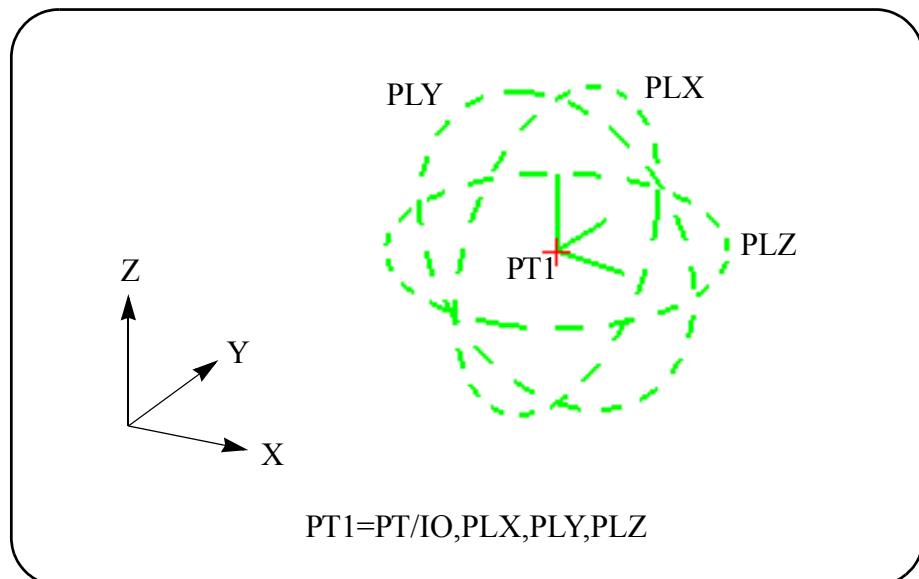
- The point-vector must pierce the plane.

3.6.11 A Point At The Intersection Of Three Planes

Command Syntax:

```
POINT/INTOF,plane,plane,plane
```

Icon Menu Sequence:



NOTE: This example defines a point which is the intersection of the three major planes of the rectangular coordinate system. However, the referenced planes may be any three intersecting planes which are defined at any orientation in three-dimensional space.

Calculation Method:

- **NCL** uses analytical geometry to calculate the intersection point.

Error Condition:

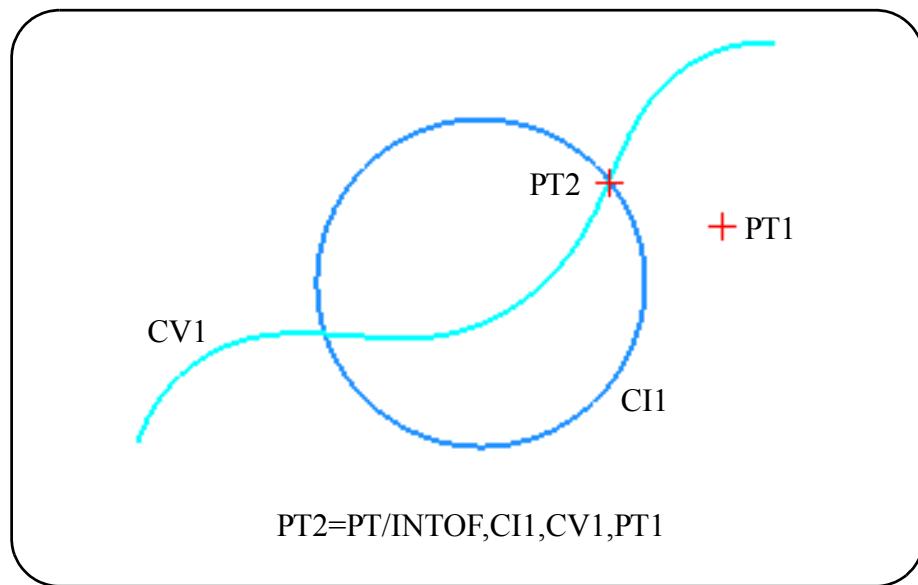
- The three referenced planes must intersect (may not be parallel).

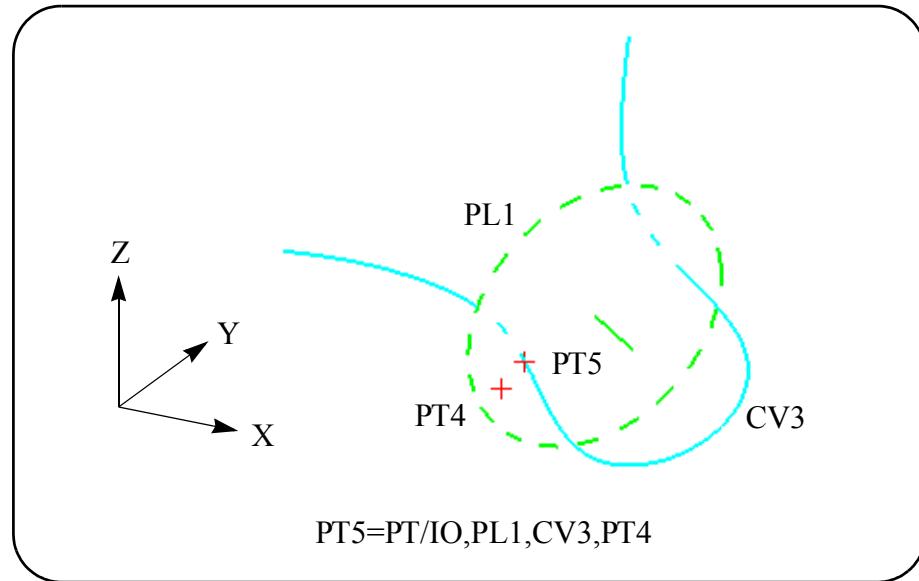
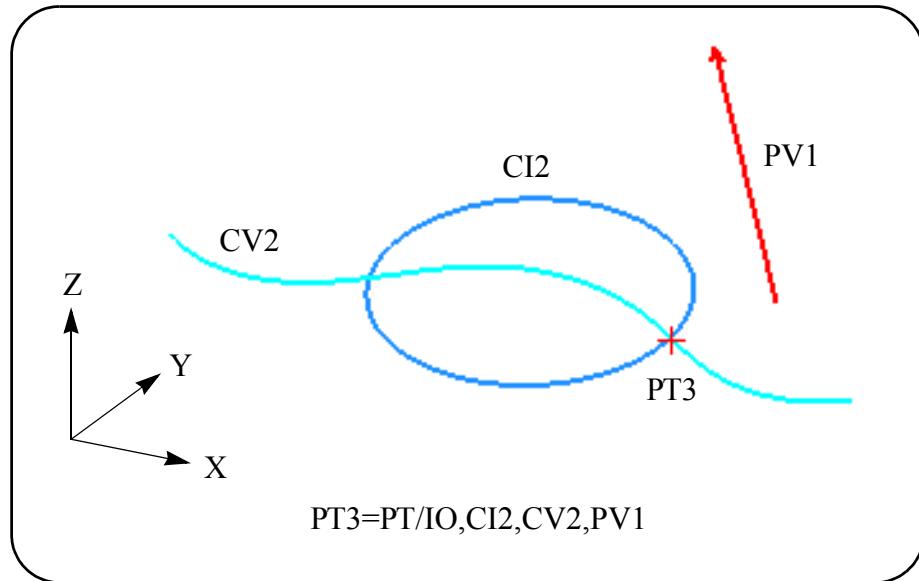
3.6.12 A Point At The Intersection Of A Circle/Plane And A Curve, Nearest To A Referenced Point Or The Origin Of A Referenced Point-Vector

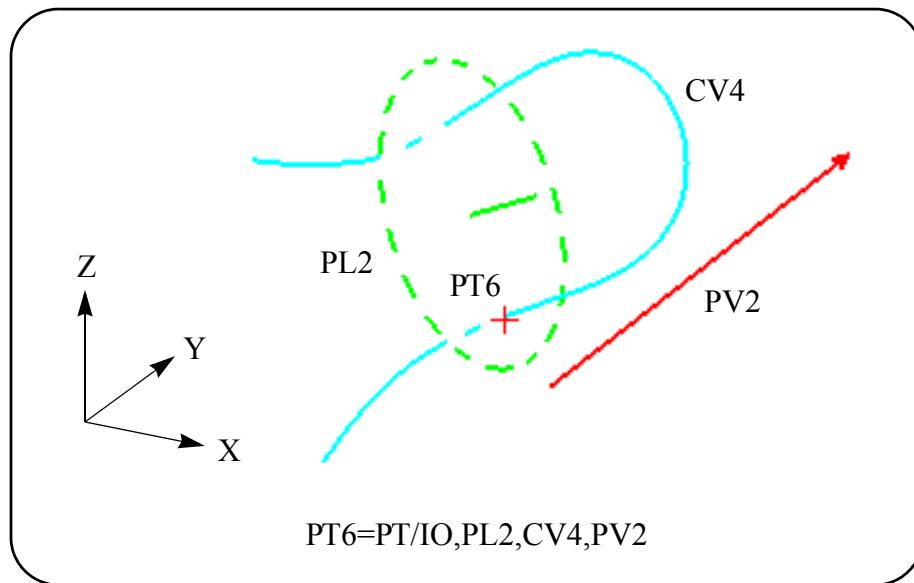
Command Syntax:

```
POINT / INTOF, circle, curve, near-point  
          plane           pntvec
```

Icon Menu Sequence:







NOTE: The near point or the origin of the point-vector is used to determine which of several possible intersection points is to be defined.

Calculation Method:

- The intersection point is calculated using the circle extension if necessary.
- The intersection point is calculated using the bounded limits of the 3D space curve.
- The curve extension is not used to calculate the intersection point.
- The Z-value of the intersection point is determined by the mathematical intersection of the 3D space curve and a right circular cylinder (parallel to the referenced circle) in three-dimensional space if circle is specified instead of a plane.

Error Conditions:

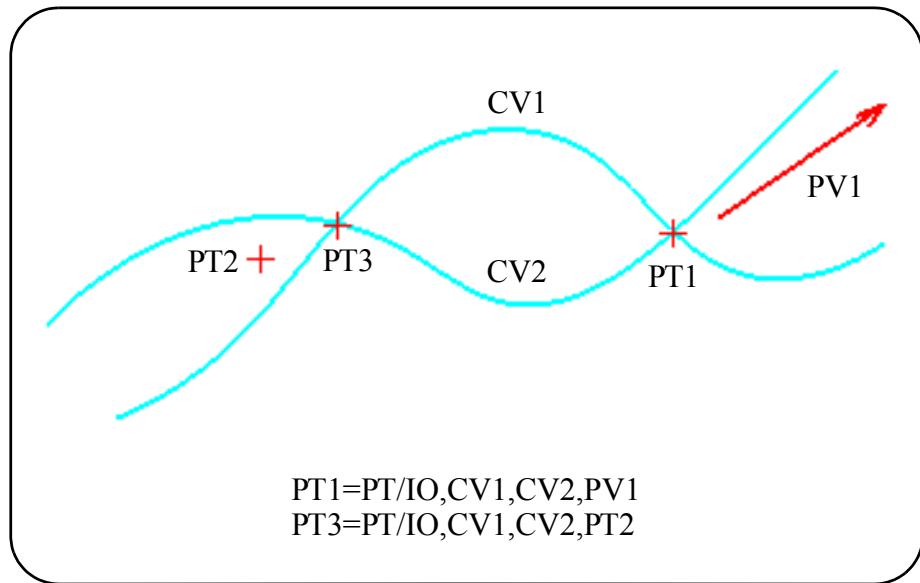
- The circle may not be "tipped" relative to the XY-plane of the current REFERENCE SYStem.
- The "projected" cylinder or the unbounded 3D plane must intersect the bounded 3D space curve in 3D space

3.6.13 All Points At The Intersection Of A Curve And, A Curve Or A Line Or A Point-Vector Or A Circle Or A Plane, Or An Intersection Point Nearest To A Referenced Point Or The Origin Of A Referenced Point-Vector

Command Syntax:

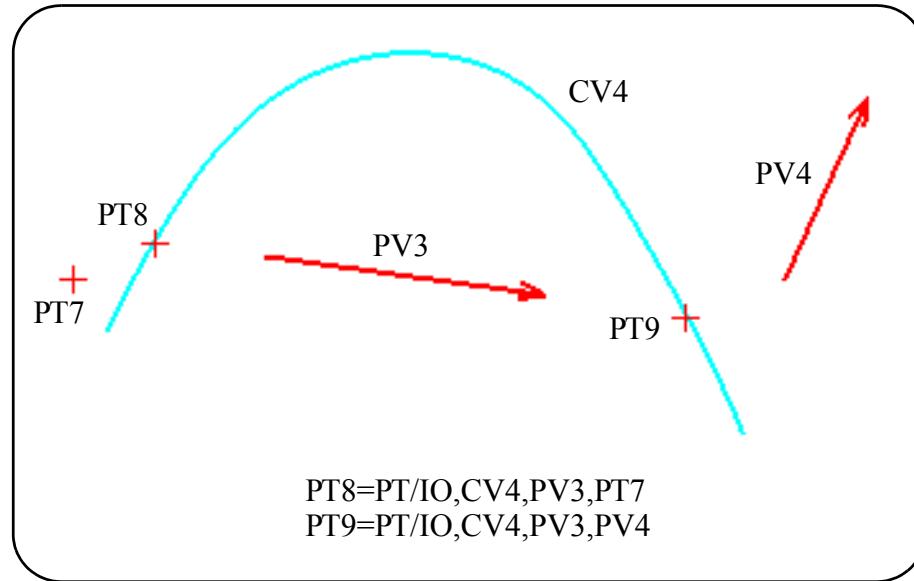
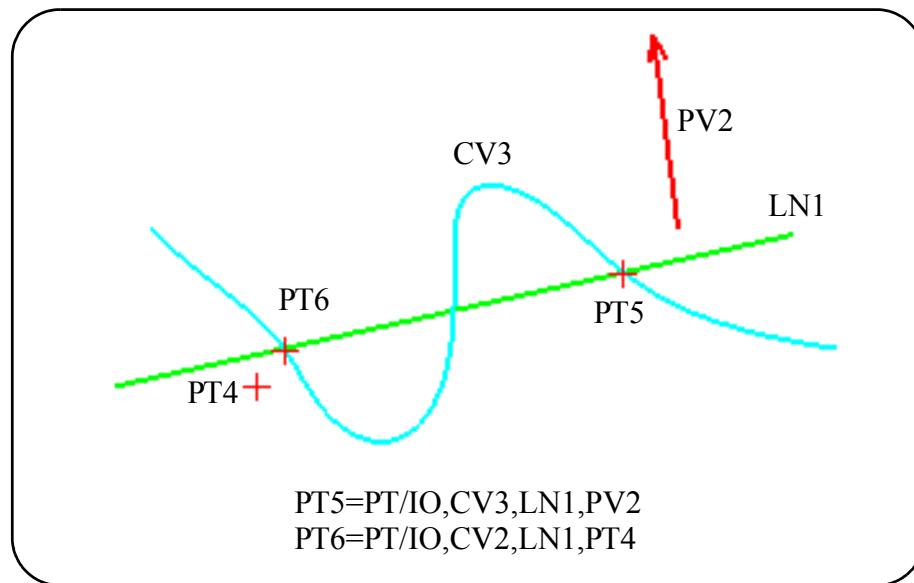
```
POINT/INTOF,curve,curve ,ALL
line    near-point
pntvec   pntvec
circle
plane
```

Icon Menu Sequence:



NOTE:

- ALL means all possible intersection points will be created.
- The near point or the origin of the point-vector is used to determine which of several possible intersection points is to be defined.



Calculation Method:

- The intersection point is calculated using the extension of the curve(s)/line if necessary.
- **NCL** creates a “temporary line” through the specified point-vector for the purpose of calculating the intersection point between the curve and the point-vector.

Error Conditions:

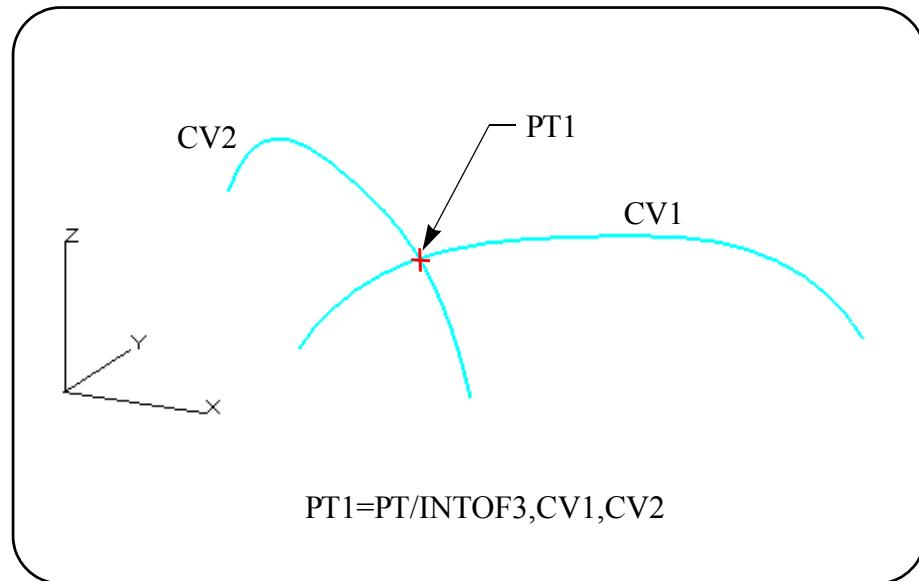
- The two curves must be on the same plane and must intersect. The plane that the two curves are on can be any plane in 3-D space.
- If the 3D space planar curve and the 3D line/point-vector do not intersect in 3D space, the result might not be what would expect.

3.6.14 A Point At The Intersection Of Two 3-D Wire Entities In 3-Dimensional Space

Command Syntax:

```
POINT/INTOF3,curve ,curve [,ALL      ]
          line   line   near-point
          circle circle   pntvec
```

Icon Menu Sequence:



NOTE:

- ALL means all possible intersection points will be created.
- The near point or the origin of the point-vector is used to determine which of several possible intersection points is to be defined.

Calculation Method

- If the optional near-point or near-pntvec is not given, the system chooses the intersection closest to the starting points of the two entities if there are multiple intersections.
- Extension of the two entities will not be used.

Error Condition:

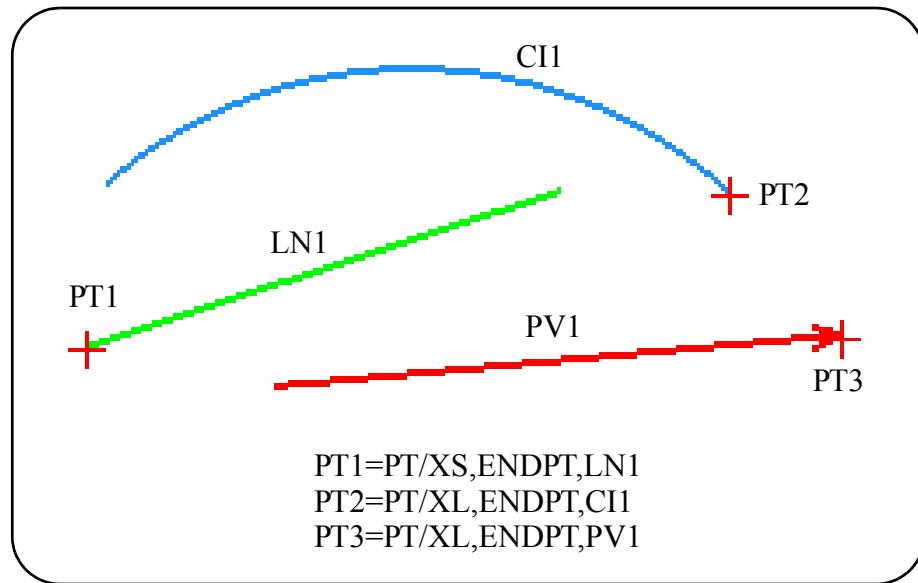
- The two 3D wire entities must intersect in the 3-D space.

3.6.15 A Point At The End Of A Line Or A Circle Or A Point-Vector

Command Syntax:

```
XSMALL
XLARGE      line
YSMALL      circle
POINT/YLARGE, ENDPT, curve
ZSMALL      point-vector
ZLARGE
```

Icon Menu Sequence:



NOTE: The **XLARGE**, **XSMALL**, **YLARGE**, **YSMALL**, **ZLARGE** and **ZSMALL** modifiers designate which of the two possible end points is to be defined. This definition is valid ONLY FOR CIRCLES WHICH ARE LESS THAN 360 DEGREES.

Calculation Method:

- **NCL** obtains the canonical data of the referenced line, circle, curve, point-vector and then calculates both end points.
- The resultant coordinate values are then analyzed to determine which of the two possible end points is desired based on the direction modifier specified.

Error Condition:

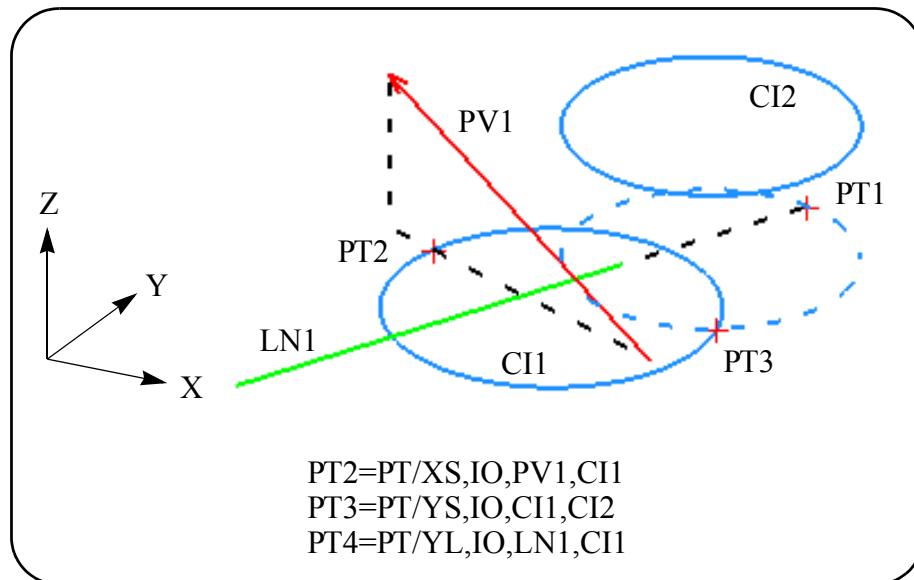
- The circle must be less than a 360 degree arc.

3.6.16 A Point At The Intersection Of A Circle/Line/Point-Vector And A Circle

Command Syntax:

```
XLARGE
XSMALL      circle
POINT/YLARGE, INTOF, line      ,circle
YSMALL      point-vector
```

Icon Menu Sequence:



NOTE: The XLARGE, XSMALL, YLARGE and YSMALL modifiers designate which of the two possible points are to be defined. The modifier describes the location of the desired point relative to the other possible point.

Calculation Method:

- The line, point-vector and the circle(s) are projected onto the XY-plane of the "current" REference SYStem followed by;
- The calculation of the intersection point using the extensions of the projected line, point-vector and/or circle(s) if necessary.
- The resultant point is projected onto the "current plane or implied plane" as specified by the ZSURF value in effect.
- If no ZSURF statement is in effect, **NCL** will use the System DEFAULT z-value which is Z=0 (the xy-plane of the current REference SYStem).

Error Conditions:

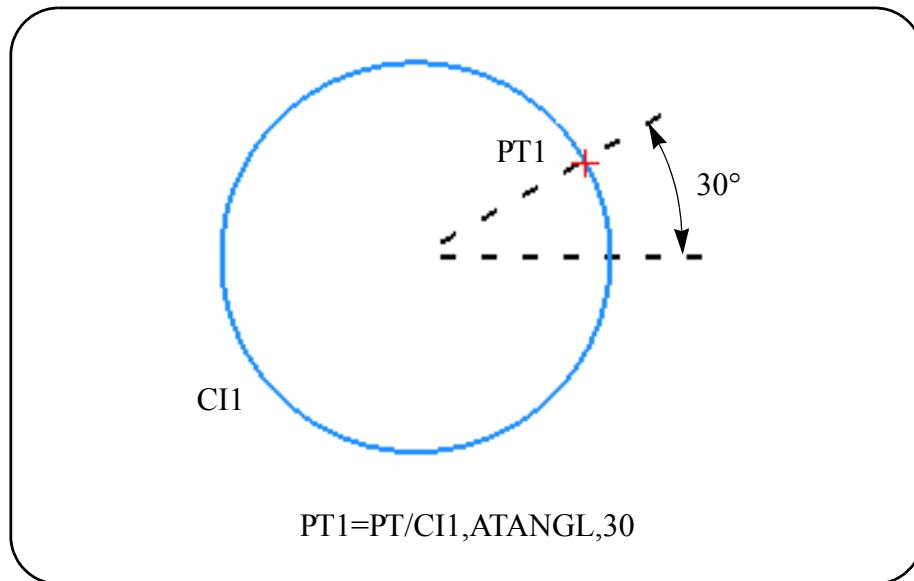
- The line or the point-vector may not be zero length when projected.
- The referenced circle(s) may not be "tipped" relative to the XY-plane of the "current" REference SYStem.
- The referenced line, point-vector and circle must intersect after the projections have been performed.

3.6.17 A Point On a Circle At An Angle With The X-axis

Command Syntax:

```
POINT/circle,ATANGL,angle
```

Icon Menu Sequence:



NOTE: The angle is expressed in decimal degrees or degrees, minutes and seconds (deg'min^{sec}; e.g. 10 minutes 2 minutes and 30 seconds will be entered as 10'2^30) and is measured from the x-axis. The angle may be specified as a positive or negative value. A positive angle is measured in a counterclockwise direction and a negative angle is measured in a clockwise direction.

Calculation Method:

- **NCL** creates a "temporary line" through the center of the circle at a user-specified angle with respect to the x-axis followed by;

- The calculation of the intersection point between the circle and the "temporary line."
- The resultant point is projected onto the "current plane or implied plane" as specified by the **ZSURF** value in effect.
- If no ZSURF statement is in effect, **NCL** will use the System DEFAULT z-value which is Z=0 (the xy-plane of the current **REFerence SYStem**).

Error Condition:

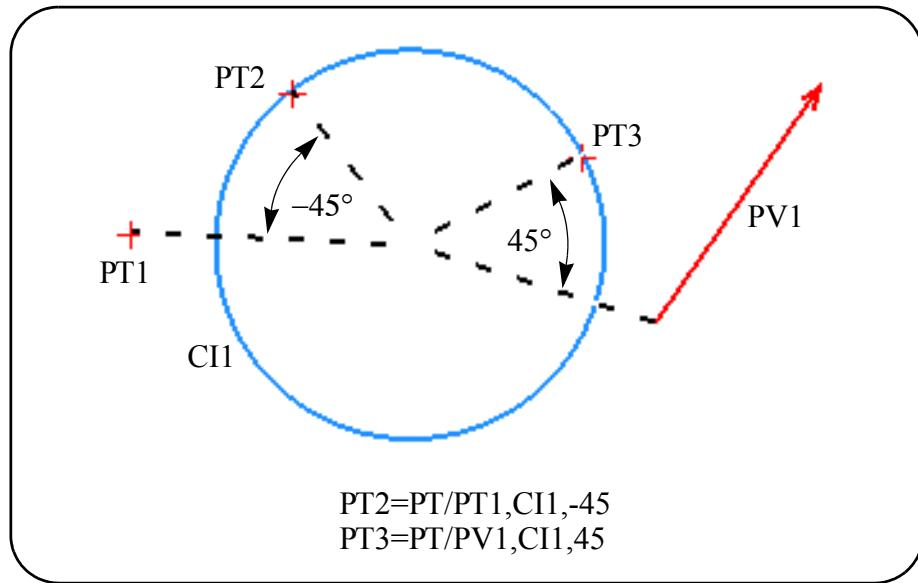
- The referenced circle may not be "tipped" relative to the XY-plane of the current **REFerence SYStem**.

3.6.18 A Point On A Circle At An Angle With A Line Passes Through A Point Or The Origin Of A Point-Vector And The Circle

Command Syntax:

```
POINT/point      ,circle,angle
           point-vector
```

Icon Menu Sequence:



NOTE: The angle is expressed in decimal degrees or degrees, minutes and seconds (deg'min^sec; e.g. 10 degrees 2 minutes and 30 seconds will be entered as 10'2^30) and is measured from a line through the referenced point and the center of the referenced circle. The angle may be specified as a positive or negative value. A positive angle is measured in a counterclockwise direction and a negative angle is measured in a clockwise direction.

Calculation Method:

- The user-specified point is projected onto the plane of the circle.
- **NCL** creates a "temporary line" through the user-specified point/vector and the center of the circle.
- **NCL** creates a second "temporary line" through the center of the circle at the user-specified angle with respect to the first "temporary line."
- The intersection point is calculated between the circle and the second "temporary line."
- The resultant point is projected onto the "current plane or implied plane" as specified by the **ZSURF** value in effect.
- If no ZSURF statement is in effect, **NCL** will use the System DEFAULT z-value which is Z=0 (the xy-plane of the current **REFerence SYStem**).

Error Conditions:

- The referenced circle may not be "tipped" relative to the XY-plane of the current **REFerence SYStem**.
- The user-specified point and the circle center point must not be coincident when projected onto the plane of the circle.

3.6.19 A Point On A Circle/Curve/Line At A Distance

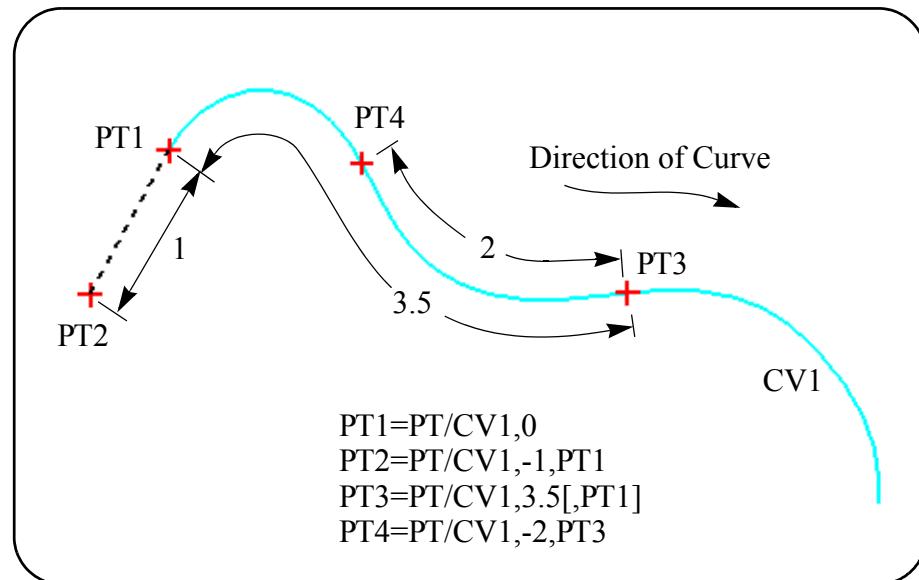
Command Syntax:

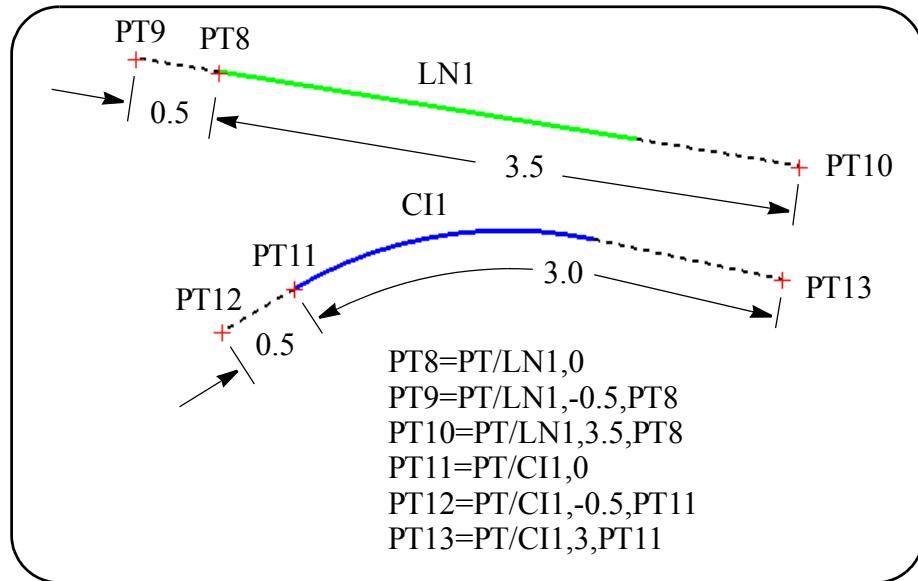
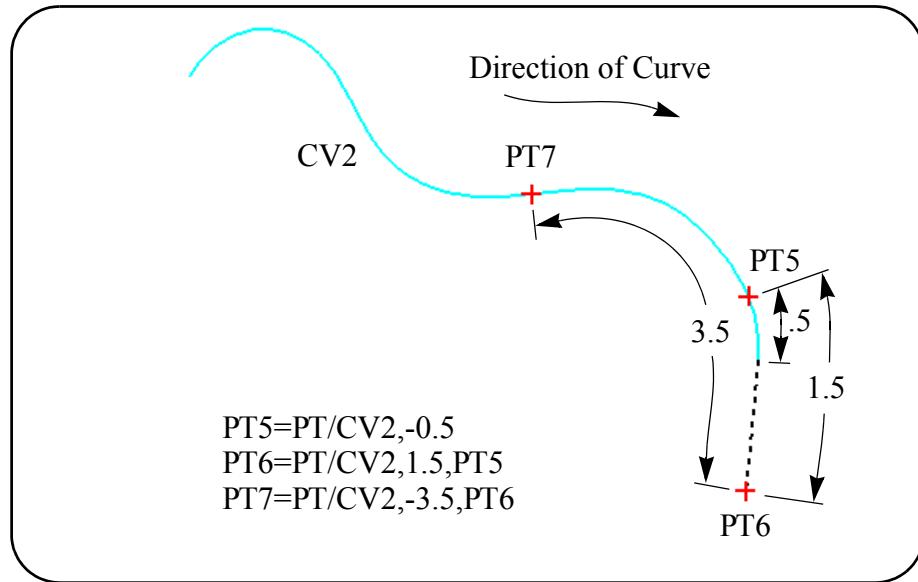
```
POINT/circle,distance[,start-point]
      curve
      line
```

Icon Menu Sequence:



The point is defined along the circle/curve/line at a true distance from the start-point.





Calculation Method:

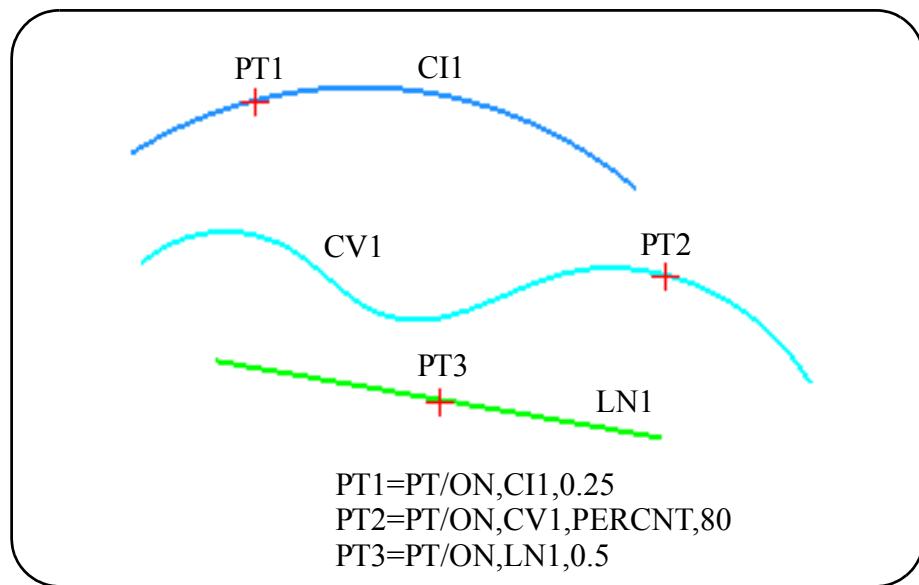
- If the optional [start-point] is not given, the distance is measured forwards along the circle/curve/line from its start point when the scalar is positive, or backwards along the circle/curve/line from its end point when the scalar is negative.
- If the optional [start-point] is given, the distance is measured from that point forward along the circle/curve/line when the scalar is positive, or backwards along the circle/curve/line when the scalar is negative.
- If the optional [start-point] is not on the circle/curve/line, it will be projected onto the circle/curve/line.
- If the scalar given causes the point to be past the start or end of the circle/curve/line, the point will be positioned on the tangent extension of the circle/curve/line.

3.6.20 A Point On A Wire Frame Entity At A Specified U Or Percent Value

Command Syntax:

```
POINT/ON,circle,           per
      line
      curve [PERCNT, ]
```

Icon Menu Sequence:



Where:

PERCNT - An optional parameter for curve entity to denote the “per” value is specified as a percentage along the physical length of the curve.

per - Is a number in the range 0 to 100 if PERCNT specified, otherwise is a number in the range 0 to 1 which represents

the U value along the wireframe entity at which to create the point. Entering 0 would create a point at the beginning of the entity while entering 1 or 100 would create a point at the end of the entity.

Note: Since “per” specifies the U value along the entity if “PERCNT” is not specified that a value of .5 does not mean the point will fall in the middle of the curve. It will be anywhere on the curve that the U has a value of .5.

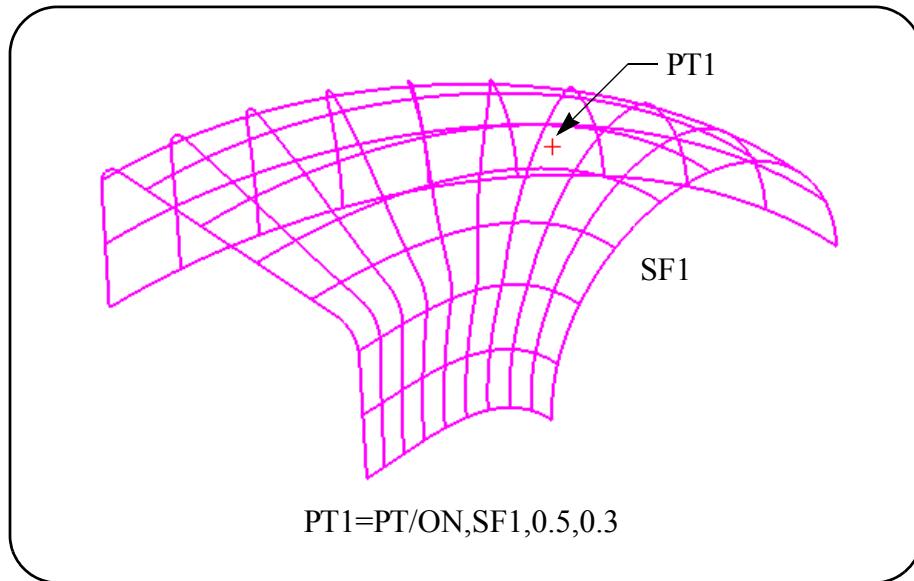
Use the “PERCNT” optional parameter to create a point at a specified percentage along the length of the curve.

3.6.21 A Point On A Surface At A Specified UV Or Percent Values

Command Syntax:

```
POINT/ON, sf, [PERCNT, ]uper,vper
```

Icon Menu Sequence:



Where:

sf - Name of the surface on which to create the point.

PERCNT - An optional parameter to denote the “uper, vper” parameters are specified as a percentage along the length (average) of the surface U and V line parametric curves instead of UV values.

uper - Is a number in the range 0 to 1 (or 0 to 100 for PERCNT) which represents the U or Percentage value of the surface at which to create the point.

vper - Is a number in the range 0 to 1 (or 0 to 100 for PERCNT) which represents the V or Percentage value of the surface at which to create the point.

PT1 is defined to be a point at a specified UV location on the surface. A specified value of .5 for both U and V does not necessarily mean the point will locate at the center of the surface.

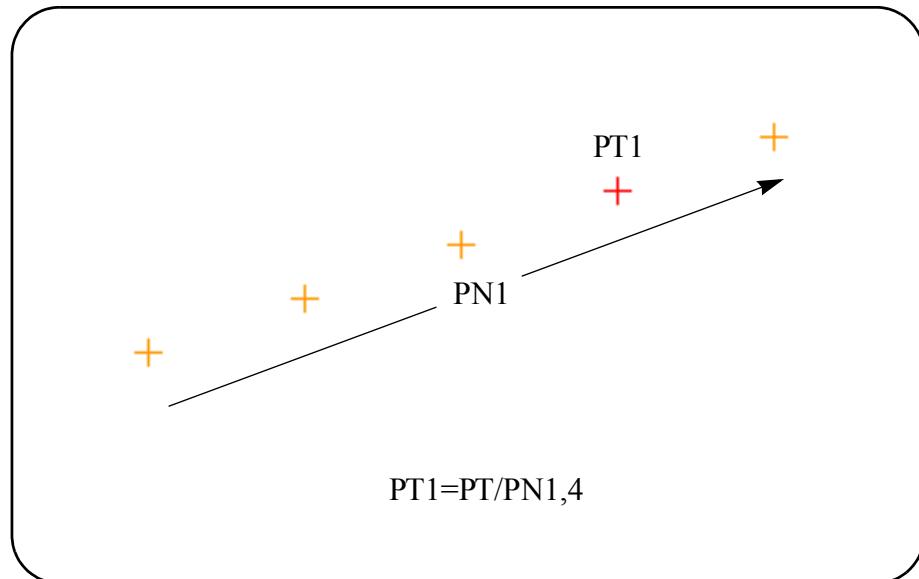
NOTE: If a **trimmed surface** is specified, the underlying (untrimmed) surface is used for the purpose of this definition. For a trimmed surface the percentage is taken on the base surface restricted to the {umin, umax, vimin, vmax} bounding box of the outer boundary.

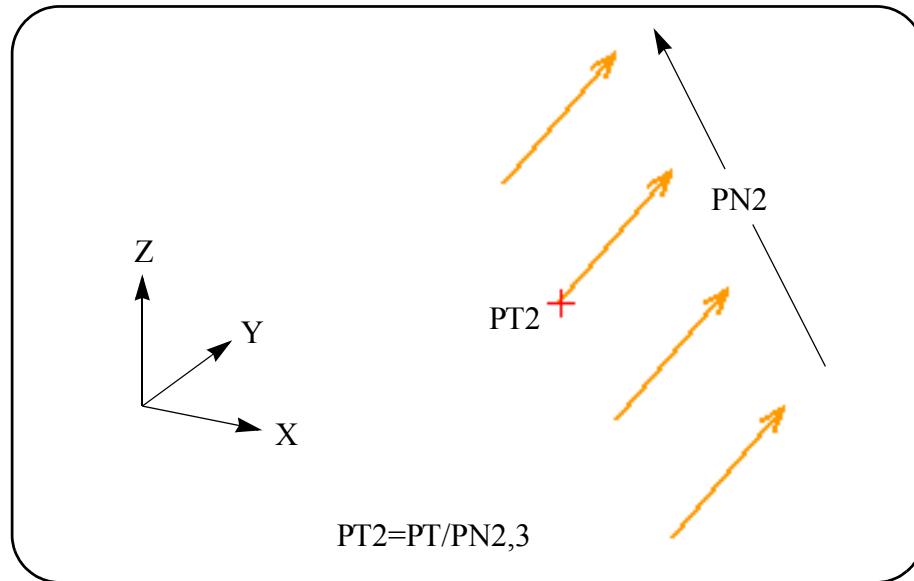
3.6.22 A Point In A Pattern

Command Syntax:

POINT/pattern, id-number

Icon Menu Sequence:





Calculation Method:

- The points in a PATERN are assigned a number in the order the PATERN was defined.
- The origin of the point-vectors will be used to generate the points if a pattern of point-vector is specified.

Error Condition:

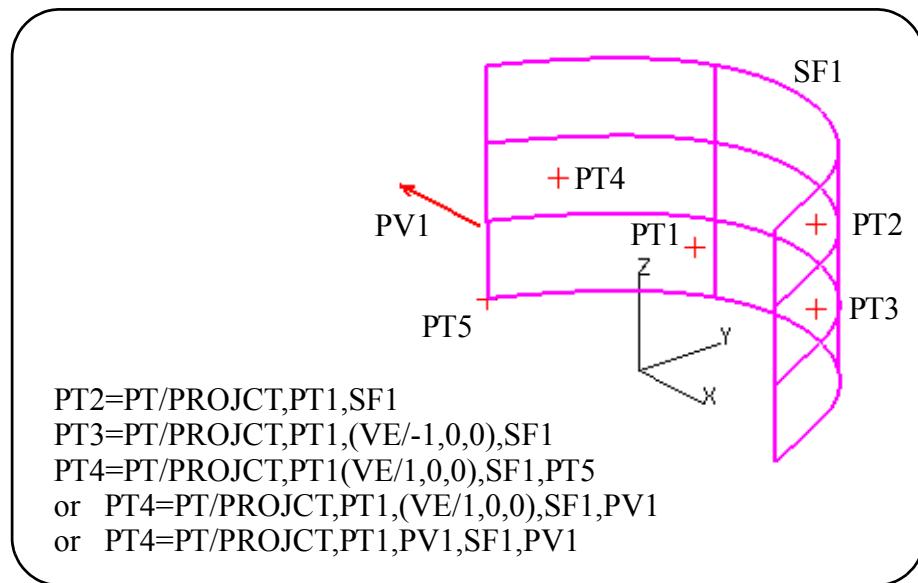
- The ID number must be a positive number within the range of the number of points in the pattern.

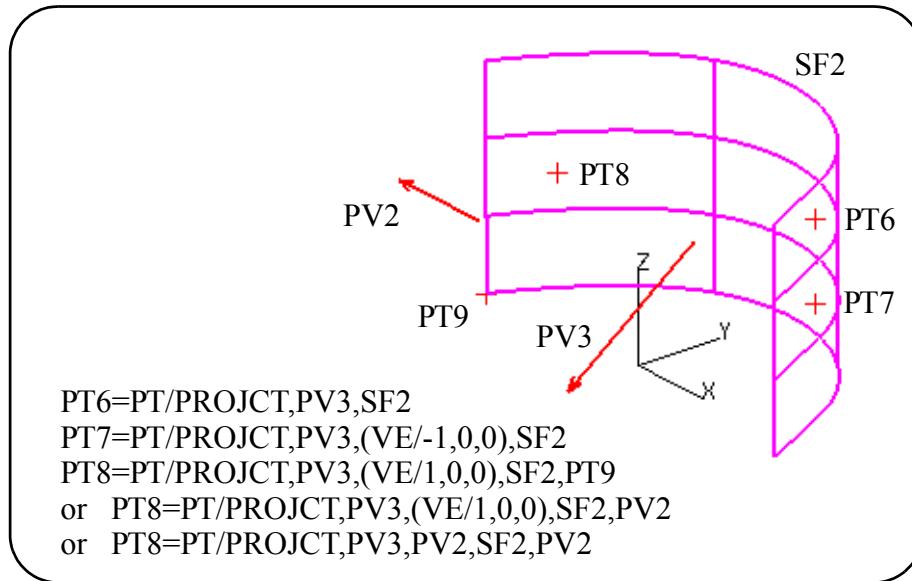
3.6.23 A Point Obtained By Projecting A Point/Point-Vector Onto A Surface/Solid Along An Optional Vector/Point-Vector And An Optional Near Point/Point-Vector

Command Syntax:

```
POINT/PROJCT,point [,vector],surface【u,v】      $  
          pntvec   pntvec   solid  
          [,near-point ]  
          pntvec
```

Icon Menu Sequence:





NOTE: The optional near point/point-vector is used to determine which of several possible projected points is the desired point.

Calculation Method:

- The referenced point (or the point-vector origin) will be projected onto the referenced surface/solid closest to the optional near-point/point-vector along the optional vector/point-vector direction (either positive or negative).
- If the optional vector/point-vector is not specified, projection will be along normal to the surface/solid from the referenced point/point-vector.
- If the optional near point/point-vector is not specified, **NCL** will look for the projected point starting from U=0.5 and V=0.5 on the surface. The default state can be changed by specifying a different U and V values.
- Calculation of the projection point will use the surface extension if necessary.
- When using an extruded, revolved, contour, or STL (non-primitive type) solid it is recommended that a projection vector be used.
- When projecting a point onto an STL solid without a vector the point will be projected onto the tessellation triangle that has the closest to normal projection of the point. In order to modify the results you can lower the

tolerance, which will increase the likelihood of the projection being closer to an expected projection point.

- When using the interface to select the solid to project to, the point where the solid was picked will be provided as a near point.

Error Condition:

- The optional projection vector/point-vector starting from the referenced point (or the point-vector origin) must intersect the surface or the surface extension.

3.6.24 PODDEF And PODPTS

The PODDEF and **PODPTS** commands have been implemented specifically to address the vacuum pod fixture system available on American GFM Ultrasonic Cutting Machines. The vacuum pods are used to secure a workpiece in place during machining operations. Prior to placing the work piece on the grid of pods, selected pods must first be positioned at the appropriate height so as to conform to the surface of the work piece. The vacuum pods swivel freely about a pivot point and will therefore attach to the surface of the work piece as it is placed on the pods. The **NCL** software must therefore calculate the appropriate height of the pod relative to the surface of the workpiece to ensure accurate placement.

3.6.24.1 PODDEF Expression

The PODDEF command is used to establish the geometry and location of the pods and can also reference an **NCL** macro which will automatically be called when the PODPTS command is executed. The PODDEF command simply establishes the default conditions but does not actually calculate the pod heights, this is done with the PODPTS command. The parameters specified in the PODDEF command remain in effect until another PODDEF command is encountered. Certain parameters of the PODDEF command can be overridden by the PODPTS command as explained in the PODPTS section of the manual.

Command Syntax:

```
PODDEF/dia,hig,ang,PN1[,h1],...[,MACRO,mname      ]
n,ptar,pvar1[,pvar2]
```

Where:

- | | |
|---------|---|
| dia | - Diameter of the pod's cup. |
| hig | - Distance from the pod's swivel axis to the cup's support ring. |
| ang | - The maximum angle at which the cup can swivel from the pod's axis. |
| PN1,... | - List of pattern identifiers defining the location of the pods. If a pattern of point-vectors is specified, then the length of the vector is used as the pod's stroke. If a pattern of points is specified, then "h1" must be used to define the |

pod's stroke. The list of patterns cannot exceed 25 patterns.

MACRO, mname - Specifies an optional macro name which will be executed when a PODPTS command is encountered. The macro can be used to output the appropriate commands to position the pods. The macro must be defined prior the PODDEF command and must be defined with at least four (4) macro variables. The first 4 macro variables will be assigned the values described below when the macro is called.

If the optional macro is not specified then the values below are assigned to a list of variable names specified either in the PODDEF command or the PODPTS command:

- | | |
|-------|---|
| n | - Specifies a variable name to store the total number of pods. |
| ptar | - Specifies the name of an array to store points calculated by the PODPTS command. For every pod, the center of the cup is located, prior to swiveling, after positioning the cup on a part surface. |
| pvar1 | - Specifies the name of an array to store point-vectors calculated by the PODPTS command. These point-vectors are the same as the original point-vectors in PN1. |
| pvar2 | - Specifies the name of an array to store point-vectors calculated by the PODPTS command. These point-vectors have points at the swivel axis and vectors normal to the part surface. All of the vectors have a length of "hig." This parameter is optional. |

Example pod definitions:

```
PODDEF/.5,1.25,20,PN1
PODDEF/.625,1.5,30,PN2,2.125
PODDEF/1.2,3,30,PN3,1.875,MACRO,M35
PODDEF/1.5,3,25,PN4,2.375,NPTS,PTA,PTVA,PTVB
```

3.6.24.2 PODPTS

The PODPTS command is used to calculate the pod height according to the pod geometry (as specified in the **PODDEF** command) and the location of the specified part surfaces. When the PODPTS command is processed it calculates the pod heights and returns the information in the form of a scalar, a subscripted point array, and two subscripted point-vector arrays. This information is passed into the variables described by n, ptar, pvar1, and pvar2 respectively. If the optional macro is specified either in the PODDEF command or the PODPTS command the information is passed into the first 4 variables of the specified macro. In this case the information can only be referenced by the macro variables within the macro and cannot be used outside the macro. If a list of variable names is specified instead of a macro, then these variable names can be used in the part program at any time after the PODPTS command is executed. An optional list of patterns and associated stroke values, macro name, or list of variables can be specified in the PODPTS command. When specified, these optional values override the corresponding settings established in the PODDEF command. If this information was not specified in the PODDEF command then it must be specified in the PODPTS command.

Command Syntax:

```
PODPTS / [ PN1 [, h1] , ] SF1 , SFn... [ , MACRO , mname ]  
n , ptar , pvar1 , pvar2
```

Where:

- | | |
|--------------------|--|
| PN1 | - Optional list of patterns and associated stroke values as described in the PODDEF command. When specified these patterns will override those specified in the PODDEF command |
| SF1,SFn... | - List of surfaces (up to 25) on which pods are to be attached. |
| MACRO, mname | - Optional macro name used to override the macro or list of variables specified in the PODEF command. |
| n,ptar,pvar1,pvar2 | - Optional list of variables used to override the macro or list of variables specified in the PODDEF command. |

Example point definitions:

```
PODPTS / PN1 , 5 , SF1 , SF2 , MACRO , MAC2
```

```
PODPTS/PN1,5,SF1,NPTS,PTA,PTVA,PTVB  
PODPTS/PN2,5,SF1,SF2,SF3  
PODPTS/PN1,SF1  
PODPTS/SF1,SF2,SF3
```

When the PODPTS command is executed the calculated points and point-vector arrays are not displayed on the screen. The DISPLAY command can be used to display the point and point-vector arrays.

If a point within the pattern cannot be projected to the specified surfaces (because all or a portion of the pod diameter falls outside the surface edges) or if the stroke value is too small to reach the surface at a particular point then no data will be generated for that point.

The PODDEF and PODPTS commands are used simply to calculate the pod height locations and do not actually output the necessary codes to position the pods. This is done using the appropriate postprocessor commands in conjunction with the data calculated by the PODPTS command. The following is an example of how these commands can be used to create the necessary output data. See the [PostWorks Reference Manual](#) for information on the POD postprocessor commands.

Example:

Assumes the bottom side of a workpiece composed of three surfaces and that all pods which fall under the workpiece need to be raised to the appropriate height. To position the pods a stop is loaded into the spindle, the stop is located at the appropriate Z height over the pod to be positioned, the pod is then raised to the stop and clamped into place. This process is repeated until all pods that fall underneath the surface of the part are positioned.

```
$$  
$$ Define pod parameters  
$$  
poddia=1.0  
pivdis=.35  
podang=45  
rows =10  
cols =20  
$$  
$$ Define pattern representing pod locations  
$$  
pn1 =PATTERN/LINEAR, (PT/-15,-10,0), (VE/2,0,0),cols
```

```

pn2      =PATTERN/PARREL,pn1,(VECTOR/0,2),rows
$$
clrpl =PLANE/0,0,1,10
$$
$$ Macro to position pods at proper Z height
$$
podpos=MACRO/np,ptar1,pvar1,pvar2
$$
pinc   =0  $$ point increment
DO      /200,row=1,rows  $$ start at row 1
    scol=1   $$ start at column 1 on odd numbered rows
    ecol=cols
    nc=1
    IF (row/2 'EQ' INT(row/2)) scol=cols; ecol=1;inc=-1
$$ even numbered row
    DO  /100,col=scol,ecol,INCR,inc
        pinc=pinc+1
        d1=DIST/pvar1(pinc),pvar2(pinc)
$$ check if point was projected
        IF (d1 'EQ' 0) JUMPTO/100
        OBTAIN/ptar1(pinc),x1,y1
        ZSURF/clrpl
        RAPID
        GOTO/(POINT/x1,y1)
        GOTO/ptar1(pinc)
        RAPID
        GODLTA/clrpl
        OBTAIN/pvar2(pinc),x1,y1,z1,i1,j1,k1
        pp1=POINT/(POINT/x1,y1,z1),i1,j1,k1
        cca(pinc)=CIRCLE/CANON,pp1,(VE/i1,j1,k1),poddia/2
$$ Circle representing positioned pod
        POD/row,col,UP,CLAMP
$$ PostWorks pod positioning command
100: CONTIN
200: CONTIN
TERMAC
$$
PODDEF/poddia,pivdis,podang,pn2,5,MACRO,podpos
$$
$$
PODPPTS/SF1,SF2,SF3
$$
FINI

```

POINT-VECTORS

The following list gives an abbreviated notation of all the valid PNTVEC definition formats. See the individual sections for a complete explanation of each format.

1. PNTVEC/x,y,[z,]i,j,k
2. PNTVEC/point,point
3. PNTVEC/point,vector
4. PNTVEC/TE,TA
5. PNTVEC/TE,FWD
6. PNTVEC/line
7. PNTVEC/CENTER,circle
8. PNTVEC/CENTER,curve [,planar]
surface
9. PNTVEC/curve,distance
10. PNTVEC/curve,point
or PNTVEC/point,curve
11. PNTVEC/point,surface
12. PNTVEC/PERPTO,plane
13. PNTVEC/point ,PERPTO,[ON ,]line [,near-point]
pntvec OFF circle pntvec
curve
plane
surface
14. PNTVEC/point ,TANTO,[ON ,]line [,near-point]
pntvec OFF circle pntvec
curve
15. PNTVEC/INTOF,plane,plane,plane,modifier
16. PNTVEC/point-vector,PLUS ,vector
MINUS

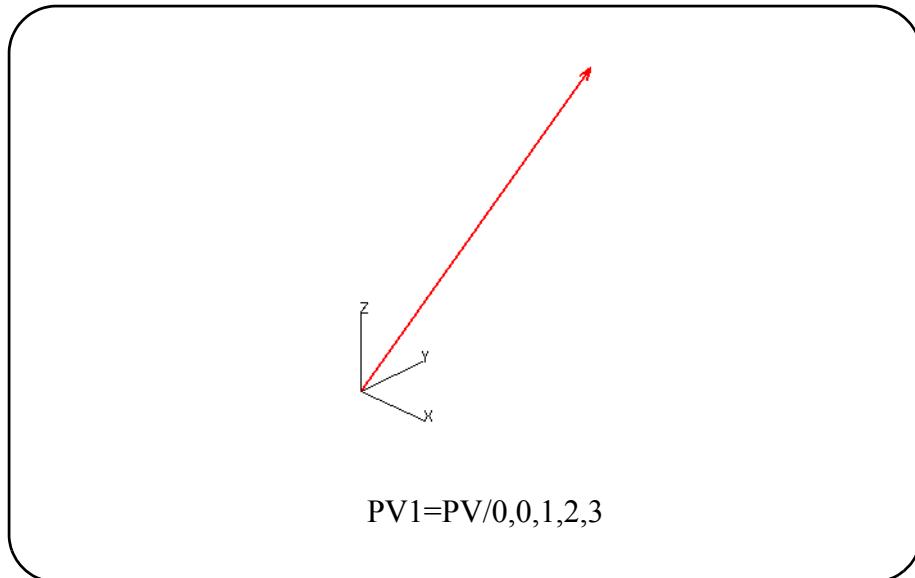
-
- 17. PNTVEC/point-vector,CROSS,vector
 - 18. PNTVEC/point-vector,TIMES,scalar
 - 19. PNTVEC/point-vector,OFFSET,scalar
 - 20. PNTVEC/UNIT,point-vector
 - 21. PNTVEC/pattern,index
 - 22. PNTVEC/PROJCT,point [,vector],surface[[u,v]]
 pntvec pntvec
 [near-point]
 pntvec
 - 23. PNTVEC/ON,circle,[PERCNT,]per
 curve
 line
 - 24. PNTVEC/ON,surface,[PERCNT,]uper,vper
 - 25. PNTVEC/surface

Example Point-Vector Expressions:

```
PV/.312,1.423,1,1,1  
PV/.125,.625,.5,2,3,4  
PV/PT1,PT2  
PV/PT1,VE3  
PV/TE,TA  
PV/TE,FWD  
PV/CE,CI1  
PV/IO,PL1,PL2,PL3,POSY  
PV/CV1,2.5  
PV/PV1,PLUS,V2  
PV/PV1,MINUS,V2  
PV/PV1,CROSS,V2  
PV/UNIT,PV1  
PV/PE,PL1  
PV/PT1,SF2  
PV/PV1,TIMES,3  
PV/PV1,OFFSET,1.625  
PV/PT1,CV2  
PV/PN1,22
```

3.7 POINT-VECTOR Expressions

The Point-Vector entity uses the name PNTVEC or PV. A point-vector represents a directional vector emanating from a defined point in space. The display symbol for a PNTVEC is a red arrow at the point specified in the definition with a magnitude defined by its direction components. The PNTVEC statement can be used in most commands in place of vectors, points or pairs of points, and vectors.



Canonical Form:

PNTVEC/X, Y, [Z,] I, J, K

The **ZSURF** statement applies to the point definition formats where a z-coordinate is not explicitly designated. In these cases, the z-coordinate is designated by the “ZSURF/plane” or “ZSURF/z-value” statement. If no ZSURF statement is in effect, **NCL** will use the XY-plane of the current **REFERENCE SYSTEM**. The i, j, and k values specify the direction components of the vector and its magnitude as the $\text{SQRT}(i^{**2} + j^{**2} + k^{**2})$.

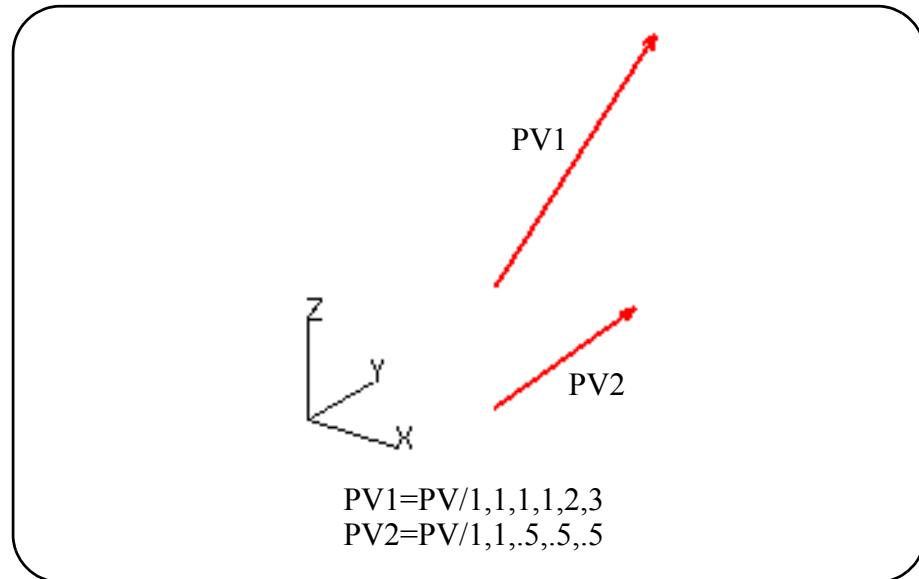
RED is the **NCL** System DEFAULT color for CAM POINT-VECTOR.

3.7.1 A Point-Vector By Its Rectangular Coordinates And Its Vector Components

Command Syntax:

```
PNTVEC/x-coordinate,y-coordinate,[z-coordinate,] $  
i-component,j-component,k-component
```

Icon Menu Sequence:



Calculation Method:

- A unique point of origin in three dimensional space will be defined if the z-coordinate is specified.
- The z-coordinate is calculated by projecting the xy-coordinates onto the "current plane or implied plane" as specified by the **NCL** statements "ZSURF/plane" or "ZSURF/ z-value" if z-coordinate is not specified.

- If no **ZSURF** statement is in effect, **NCL** will use the System DEFAULT z-value which is Z=0 (the xy-plane of the current **REFerence SYStem**) if z-coordinate is not specified.
- The magnitude of the point-vector equals to the square-root of the summation of the square of all the vector components.

Error Conditions:

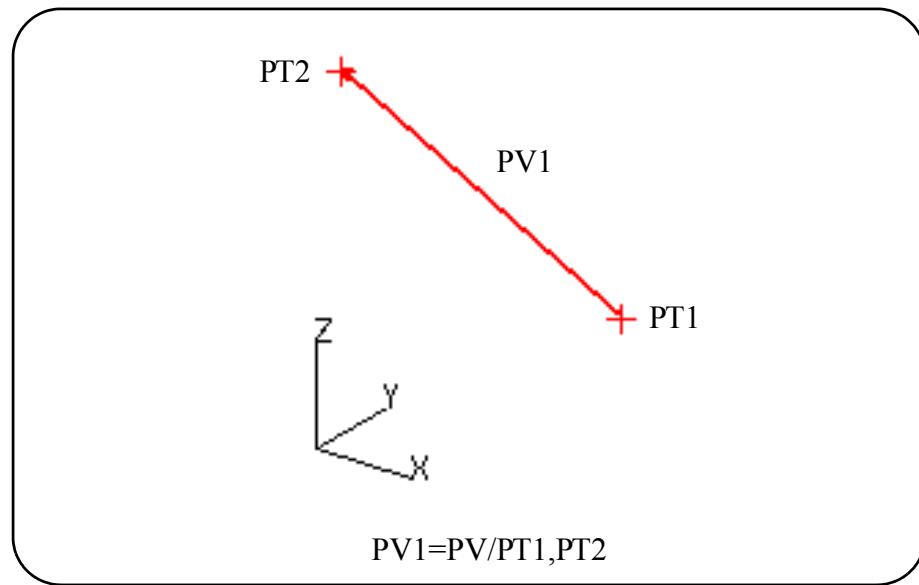
- The minimum input is x, y and i, j, k-vector components where any or all coordinate values may be positive (+), negative (-) or zero.
- If z-coordinate is not specified, the ZSURF plane must not be a plane which is parallel to the Z-axis of the current REFerence SYStem since a z-value cannot be determined.
- The i, j, k-vector components must calculate a magnitude greater than zero.

3.7.2 A Point-Vector From One Point To Another

Command Syntax:

PNTVEC/point,point

Icon Menu Sequence:



Calculation Method:

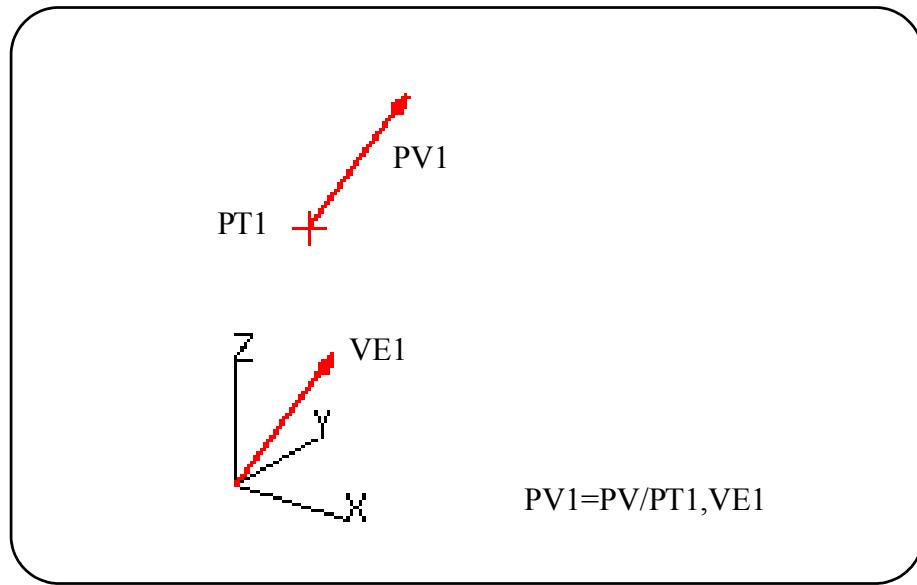
- The magnitude of the point-vector is calculated from the first referenced point to the second referenced point.
- The direction of the point-vector will be from the first referenced point to the second referenced point.

3.7.3 A Point-Vector From A Point And A Vector

Command Syntax:

PNTVEC/point, vector

Icon Menu Sequence:



Calculation Method:

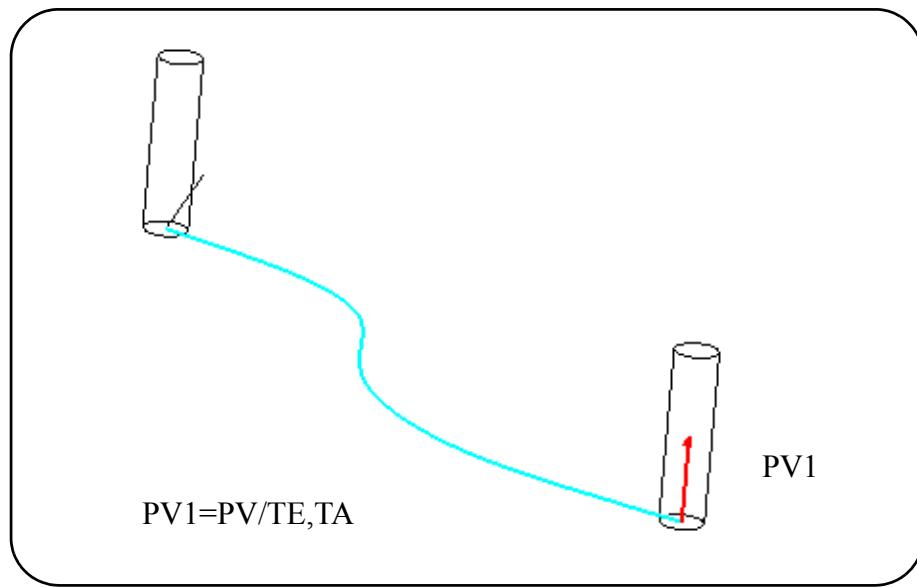
- The magnitude of the point-vector is calculated from the referenced vector.
- The direction of the point-vector will be from the first referenced point in the direction of the specified vector.

3.7.4 A Point-Vector In The Direction From The Current Tool End Up The Current Tool Axis

Command Syntax:

PNTVEC/TE, TLAXIS

Icon Menu Sequence:



Calculation Method:

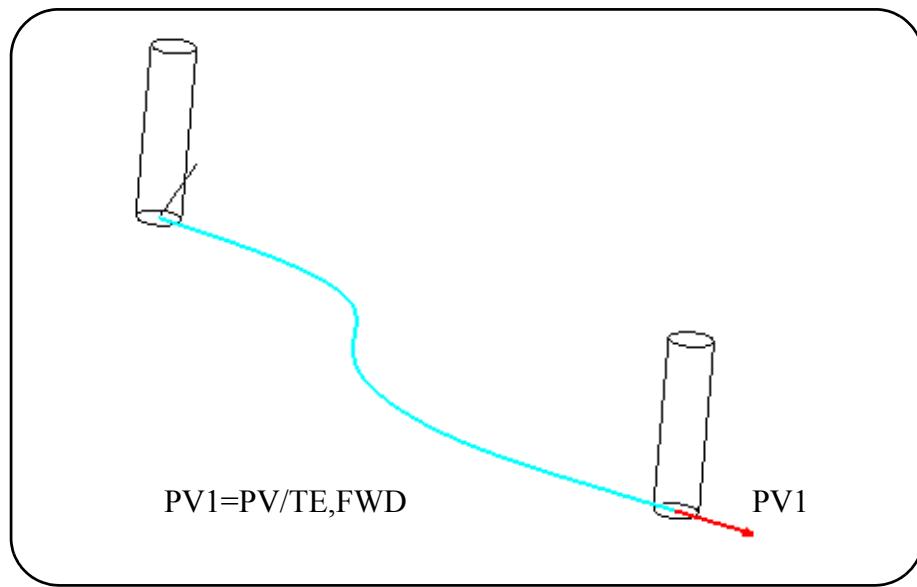
- The magnitude of the point-vector is one unit long,
- The direction of the point-vector points up the tool axis.
- If no **TLAXIS** statement has been specified prior to the use of this statement, **NCL** will use the System DEFAULT TLAXIS which is 0, 0, 1 (positive Z).

3.7.5 A Point-Vector In A Direction From The Current Tool End Toward The Current Forward Motion

Command Syntax:

PNTVEC / TE , FWD

Icon Menu Sequence:



Calculation Method:

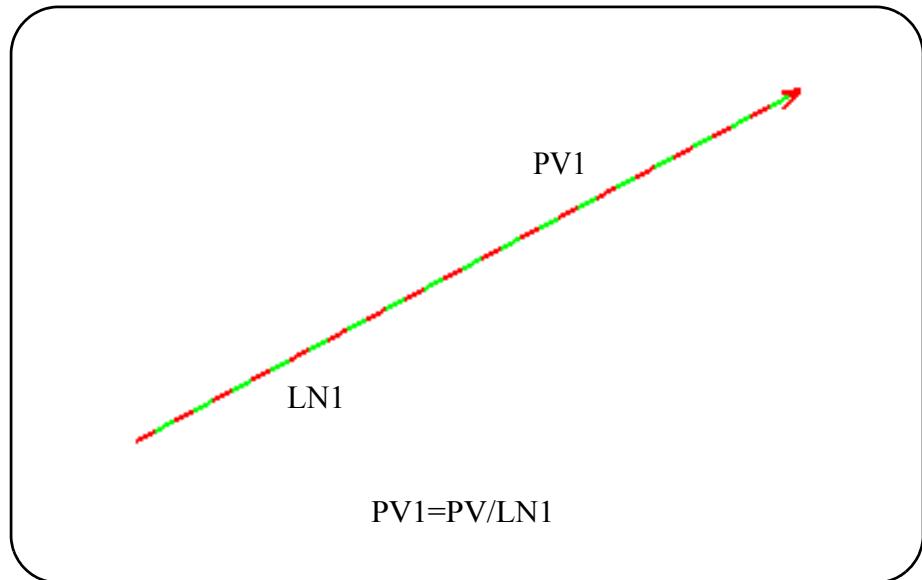
- The magnitude of the point-vector is one unit long,
- The direction of the point-vector points in the direction of the current tool motion.

3.7.6 A Point-Vector Along A Line

Command Syntax:

PNTVEC/line

Icon Menu Sequence:



Calculation Method:

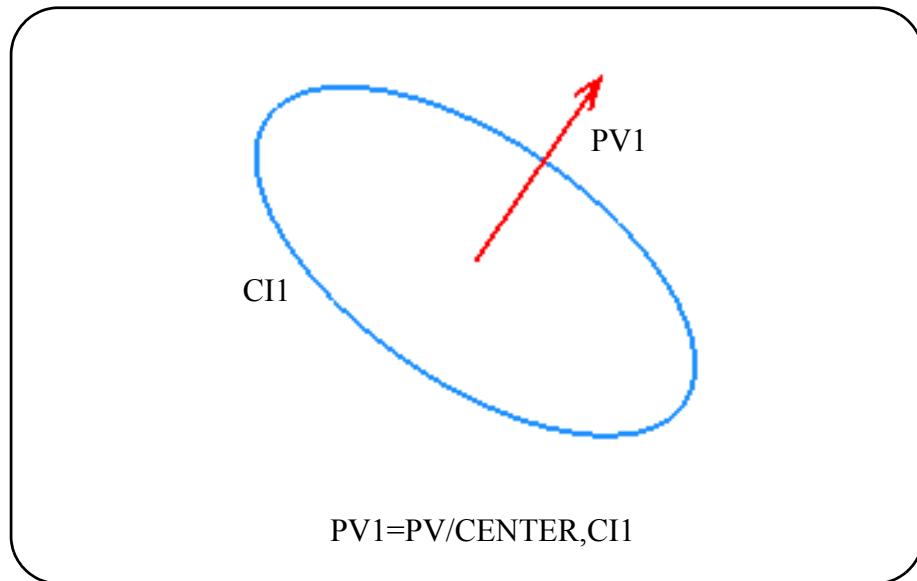
- The magnitude of the point-vector is same as the underlying line.
- The direction of the point-vector is in the same direction as how the line was defined.

3.7.7 A Point-Vector From The Center Of A Circle

Command Syntax:

PNTVEC/CENTER,circle

Icon Menu Sequence:



Calculation Method:

- The magnitude of the point-vector is one unit long originating from the circle center.
- The direction of the point-vector is in the same direction of the plane upon which the circle defined.

3.7.8 A Point-Vector At The Center Of Gravity Of A Curve Or A Surface Perpendicular To The Optimal Plane

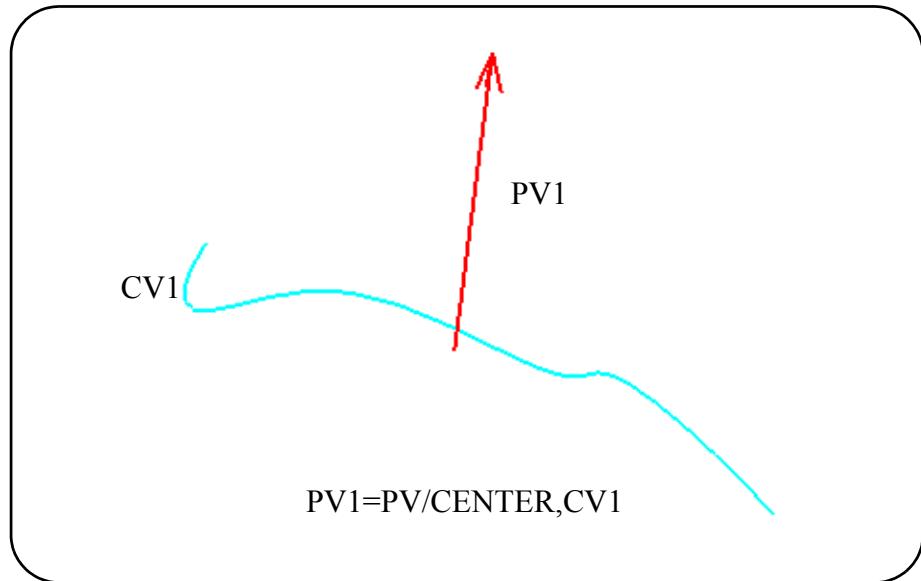
Command Syntax:

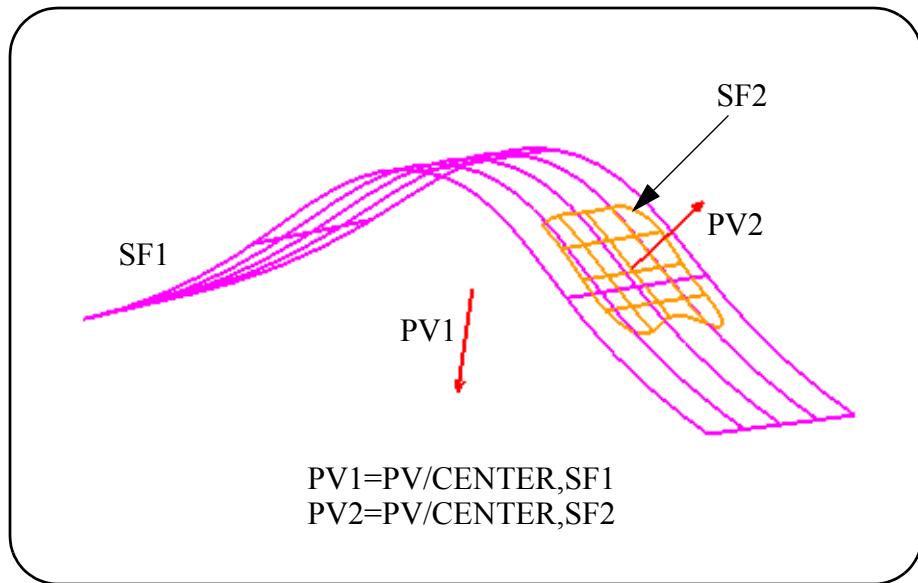
```
PNTVEC/CENTER,curve [,planar]  
surface
```

where:

planar An optional scalar holds a value of 1 if the curve or surface is planar, otherwise it holds a value of 0.

Icon Menu Sequence:





Calculation Method:

- The magnitude of the point-vector is one unit long originating from the center of gravity of the curve or surface.
- The point-vector direction is normal to the optimal plane of the curve or surface.
- The trimmed surface itself will be considered for the center of gravity and the optimal plane. The underlying parent surface will not be considered.

Error Condition:

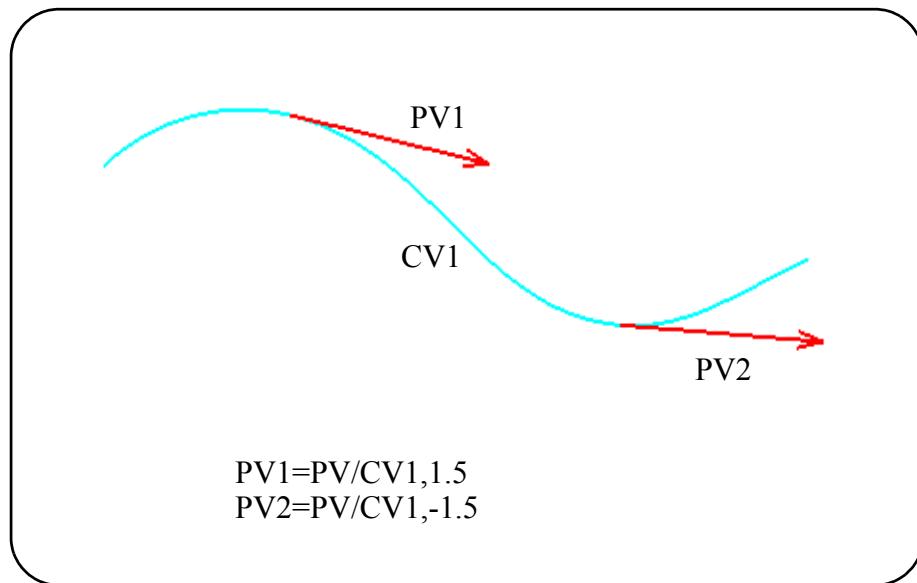
- A straight curve is specified.

3.7.9 A Point-Vector From A Point On A Curve Along A Specified Distance And Tangent To The Curve

Command Syntax:

PNTVEC/curve,distance

Icon Menu Sequence:



Calculation Method:

- The point-vector is on the curve, and the direction will be a function of the direction of the curve. Moreover, the point-vector will have the slope of the curve at that point.
- The tangent point-vector will be one unit in magnitude along the forward sense of the curve.
- The distance is measured from the point-vector forward along the curve if the distance specified is positive, or backwards along the curve when the distance specified is negative.

3.7.10 A Point-Vector From An Existing Point And In The Direction Of The Slope Of A Curve

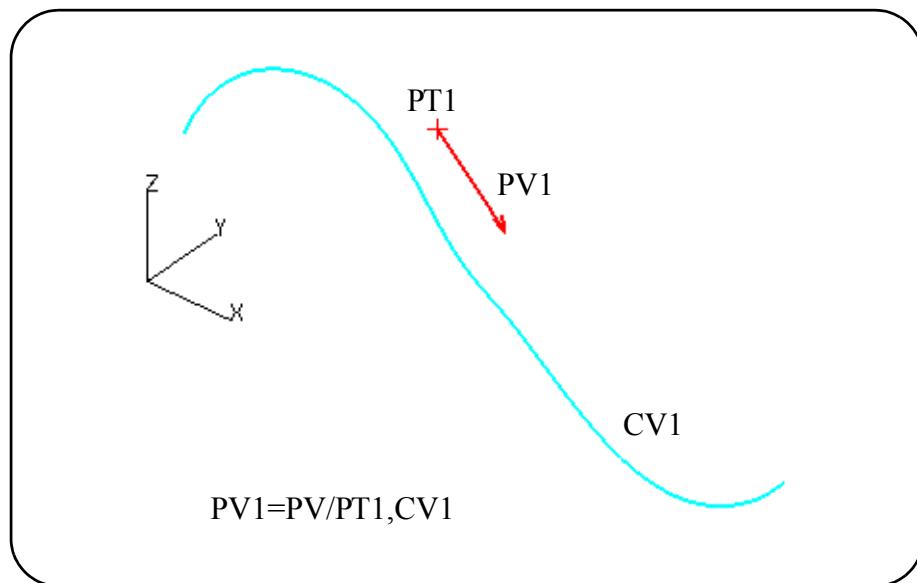
Command Syntax:

PNTVEC/point,curve

or

PNTVEC/curve,point

Icon Menu Sequence:



Calculation Method:

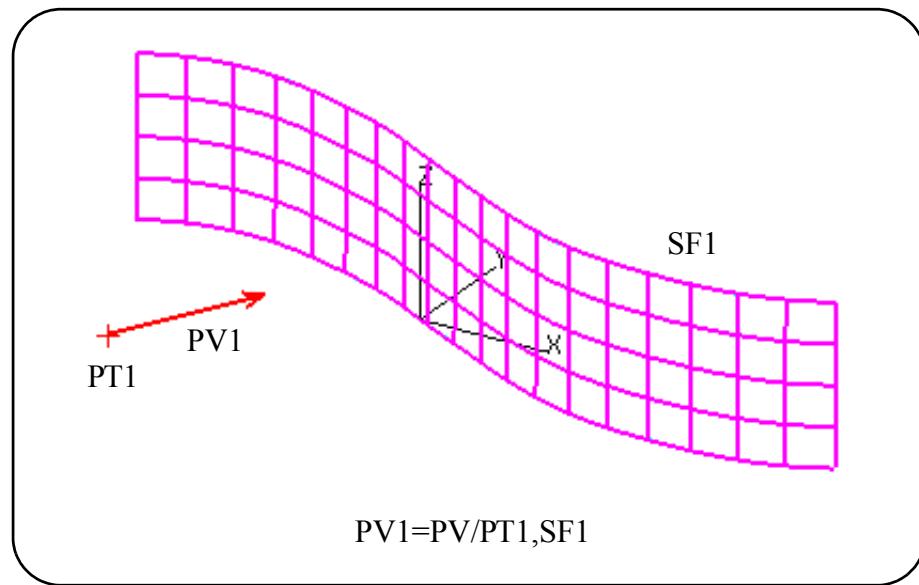
- The point-vector origin is at the point, and in the direction of the slope of the curve.
- If the point is not on the curve it is projected on to it along the shortest distance from the point to the curve.

3.7.11 A Point-Vector As The Directed Distance From A Point Normal To A Surface

Command Syntax:

```
PNTVEC/point,surface
```

Icon Menu Sequence:



Calculation Method:

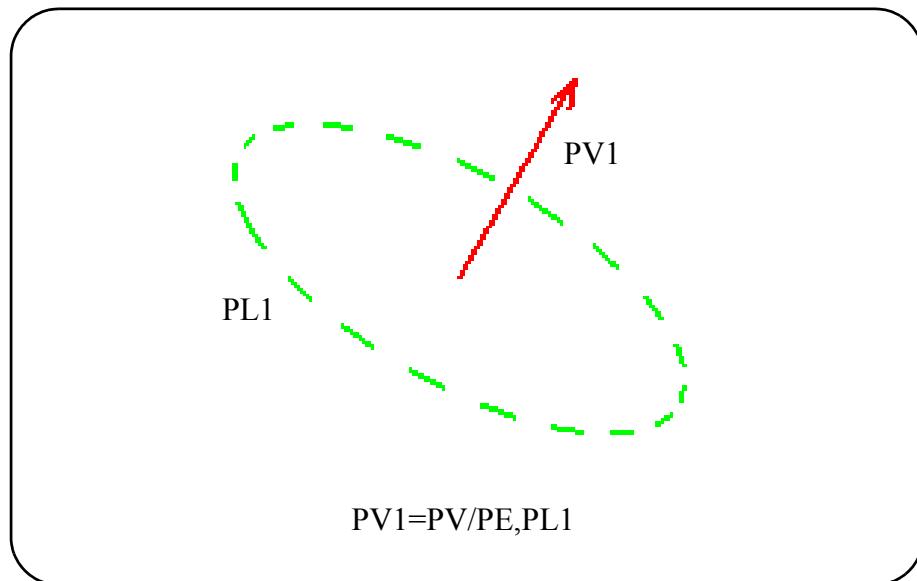
- The magnitude of the point-vector is one unit.
- The direction of the point-vector is from the specified point along a line normal to the surface.

3.7.12 A Point-Vector Perpendicular To A Plane

Command Syntax:

PNTVEC / PERPTO, plane

Icon Menu Sequence:



Calculation Method:

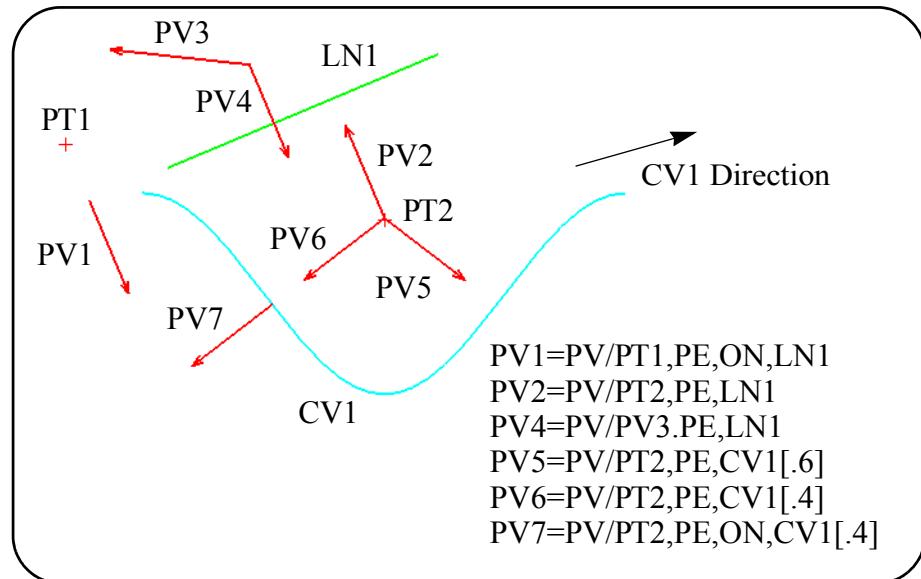
- The magnitude of the point-vector is one unit long.
- The direction of the point-vector is determined by the canonical form of the specified plane.

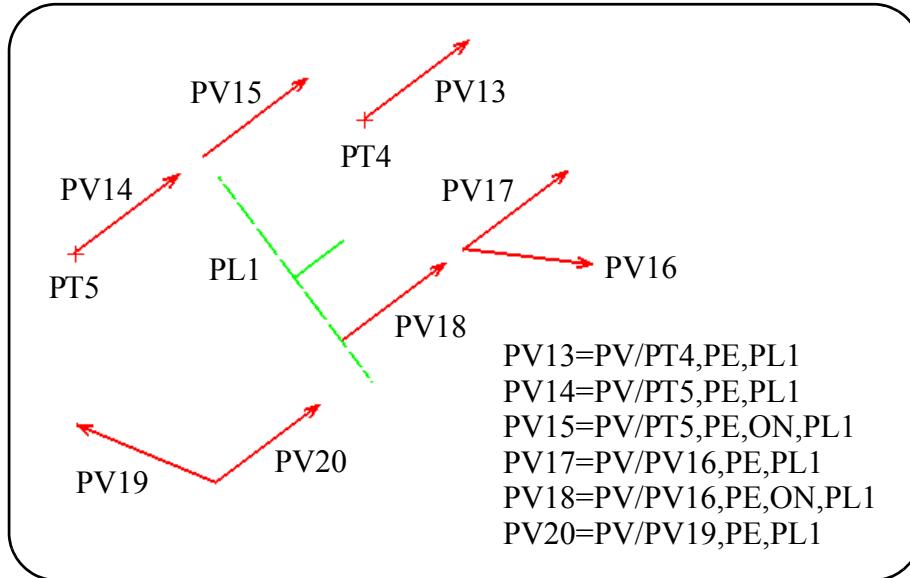
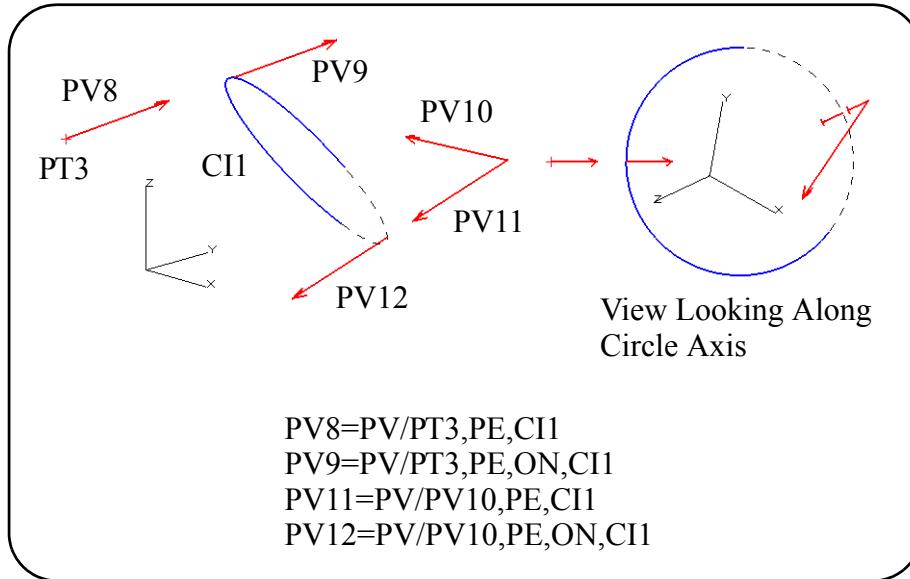
3.7.13 A Point-Vector Through A Point Or Point-Vector And Perpendicular To A Line, Circle, Curve, Plane Or Surface

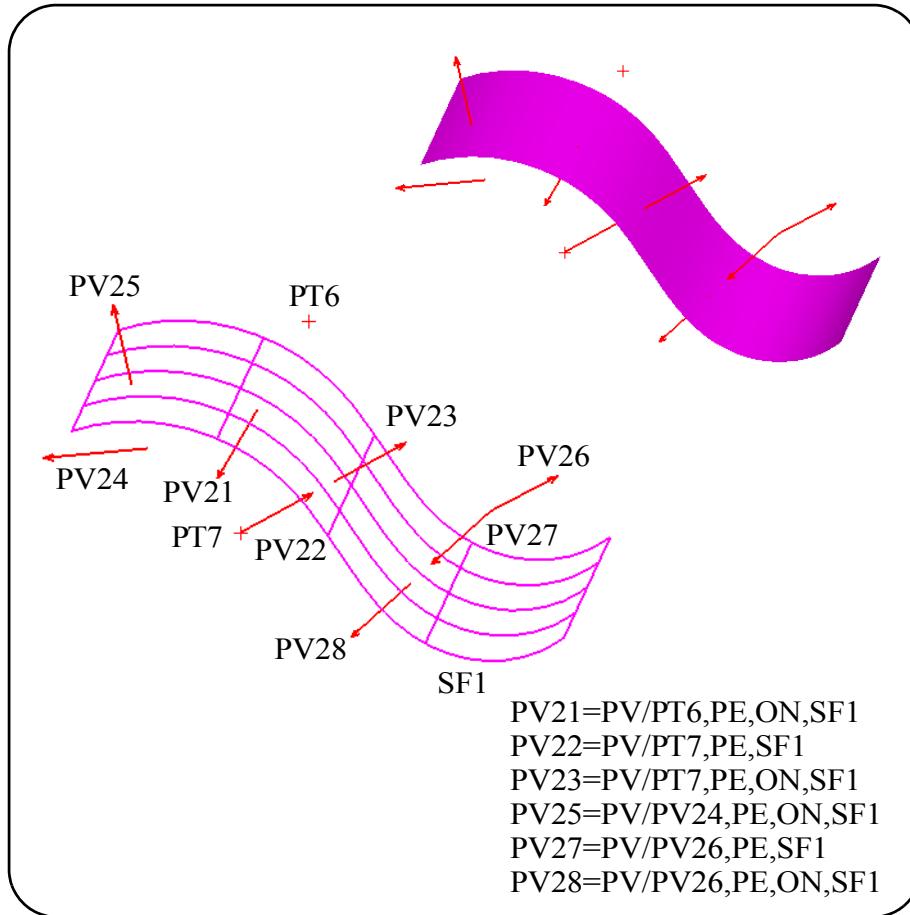
Command Syntax:

```
PNTVEC/point , PERPTO, [ON ,]line      [,near-point ]
          pntvec           OFF circle      pntvec
                           curve
                           plane
                           surface
```

Icon Menu Sequence:







Calculation Method:

- The magnitude of the point-vector is one unit.
- “OFF” specified the origin of the point-vector is the specified point or the origin of the specified point-vector. This is the default condition.
- “ON” specified the projected point on the entity is the origin of the point-vector.
- The direction of the point-vector is determined by the canonical form of the specified plane.
- For all other entities, the direction of the point-vector is from the specified point or point-vector to the specified entity. If the specified point or point-vector lies exactly on the surface, the direction will be determined by the surface normal at that point.

- If the optional near-point or near-point-vector is not specified for curve or surface entity, **NCL** will look for the projected point starting from U=0.5 (and V=0.5, surface only) of the entity. The default state can be changed by specifying a different U and V values.

Error Conditions:

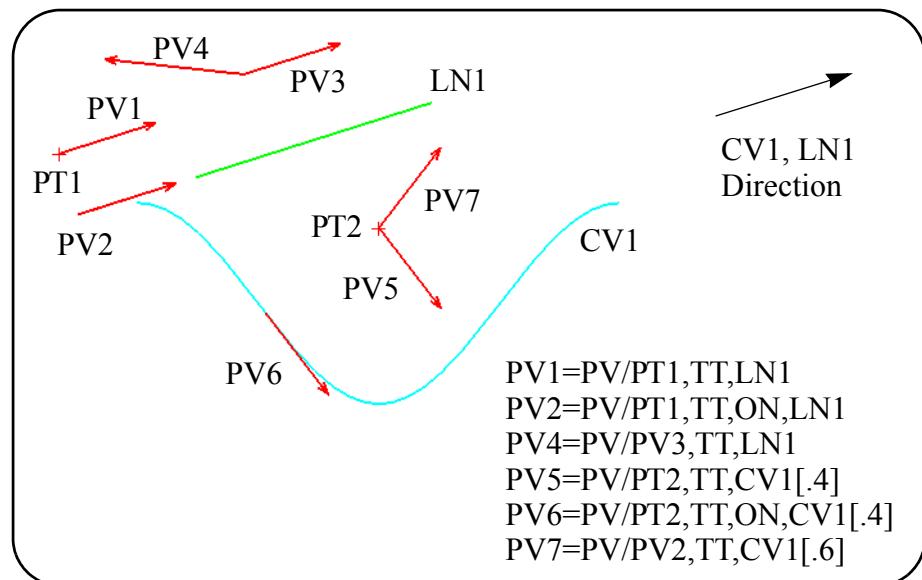
- The specified point or point-vector lies exactly on the specified curve, line or circle.
- The specified point or point-vector lies along the normal axis of the specified circle.

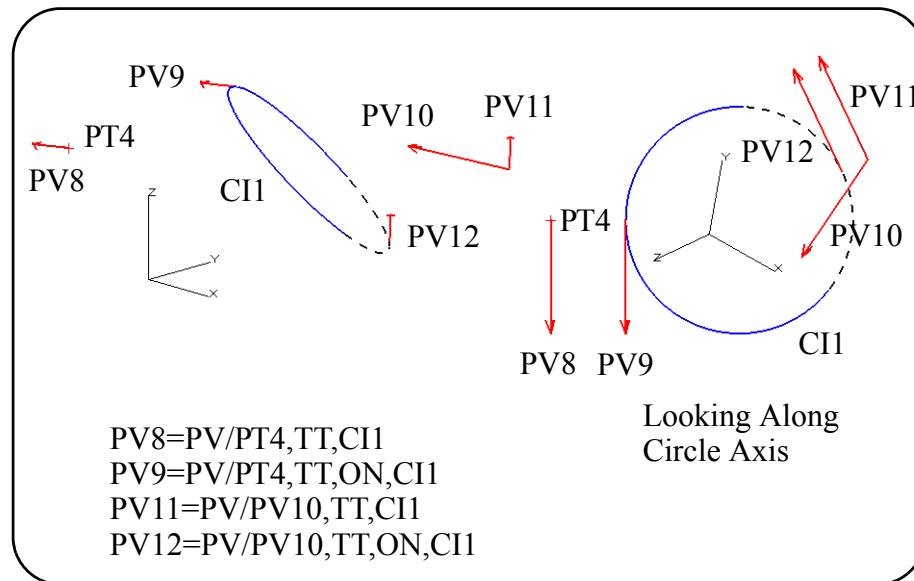
3.7.14 A Point-Vector Through A Point Or Point-Vector And Tangent To A Line, Circle, Curve

Command Syntax:

```
PNTVEC/point ,TANTO,[ON ,]line [,near-point ]
          pntvec      OFF   circle      pntvec
                           curve
```

Icon Menu Sequence:





Calculation Method:

- The magnitude of the point-vector is one unit.
- “OFF” specified the origin of the point-vector is the specified point or the origin of the specified point-vector. This is the default condition.
- “ON” specified the projected point on the entity is the origin of the point-vector.
- The direction of the point-vector is determined by the direction of the specified entity at the projected point.
- If the optional near-point or near-point-vector is not specified for the curve entity, **NCL** will look for the projected point starting from U=0.5 of the curve. The default state can be changed by specifying a different U value.

Error Condition:

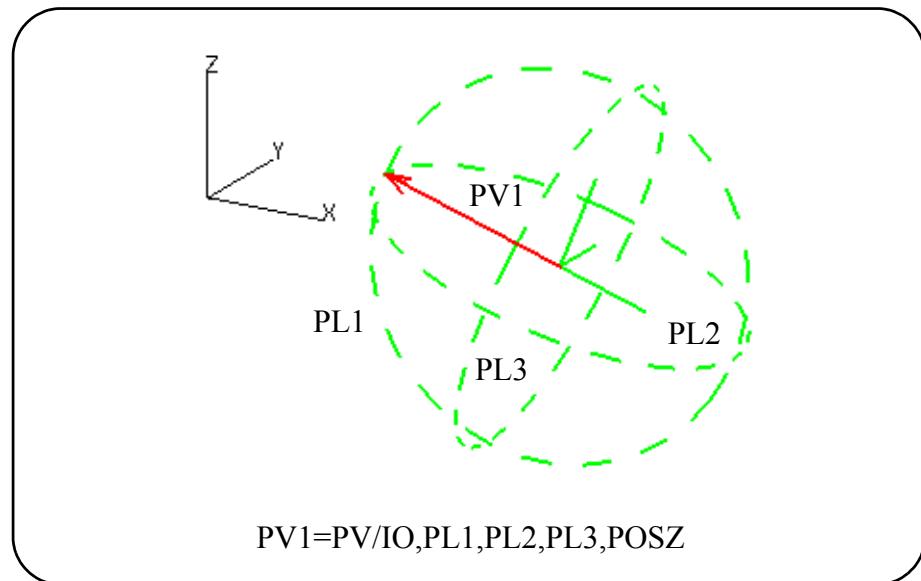
- The specified point or point-vector lies along the normal axis of the specified circle.

3.7.15 A Point-Vector As The Intersection Of Three Planes

Command Syntax:

```
POSX
NEGX
POSY
PNTVEC/INTOF,plane,plane,plane,NEGZ
POSZ
NEGZ
```

Icon Menu Sequence:



Calculation Method:

- The point-vector originates at the intersection point of the three planes.
- The magnitude of the point-vector is one unit long.

- The direction of the point-vector will be along the intersection of the first two planes specified.
- The orientation modifiers **POSX**, **NEGX**, **POSY**, **NEGY**, **POSZ** and **NEGZ** indicate the general direction of the point-vector to be defined.

Error Condition:

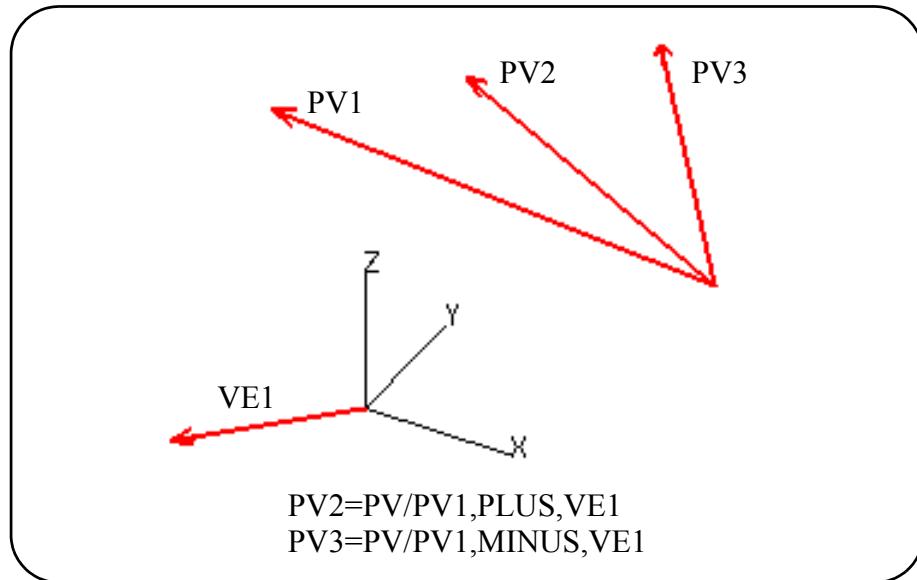
- The planes must intersect.

3.7.16 A Point-Vector As The Sum Or Difference Of A Point-Vector And A Vector

Command Syntax:

```
PLUS
PNTVEC/point-vector,MINUS,vector
```

Icon Menu Sequence:



Calculation Method:

- The magnitude of the new point-vector is calculated as the sum or difference of the referenced point-vector and vector as follows:
- Sum = $\text{SQRT}(a^{**2} + b^{**2} + 2*a*b*\cos(a))$
- Difference = $\text{SQRT}(a^{**2} + b^{**2} - 2*a*b*\cos(a))$

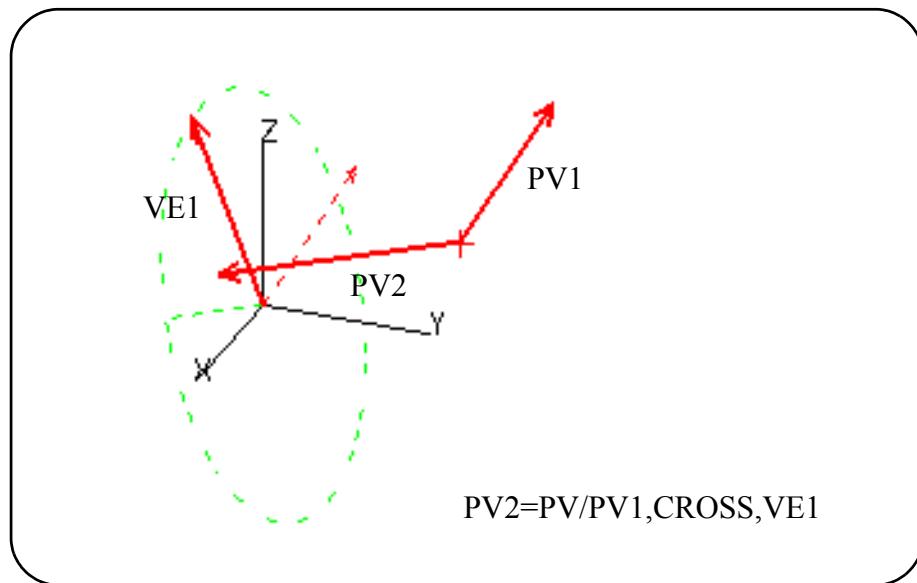
- Where "a" is the length of the point-vector and "b" is the length of the vector.
- The direction of the new point-vector will be the sum or difference of the x, y and z-coordinates depending on the modifier used. Each coordinate is calculated separately; i.e. the x-coordinate of the vector is added to or subtracted from the x-coordinate of the point-vector.
- The origin of the new point-vector will be at the origin of the referenced point-vector.

3.7.17 A Point-Vector As The Cross Product (Normal) Of A Point-Vector And A Vector

Command Syntax:

```
PNTVEC/point-vector,CROSS,vector
```

Icon Menu Sequence:



Calculation Method:

- The magnitude of the new point-vector is equal to the product of the magnitudes of the referenced point-vector and vector times the sine of the angle between them.
- **NCL** creates a “temporary vector” by utilizing the vector portion (i, j, k) of the point-vector.

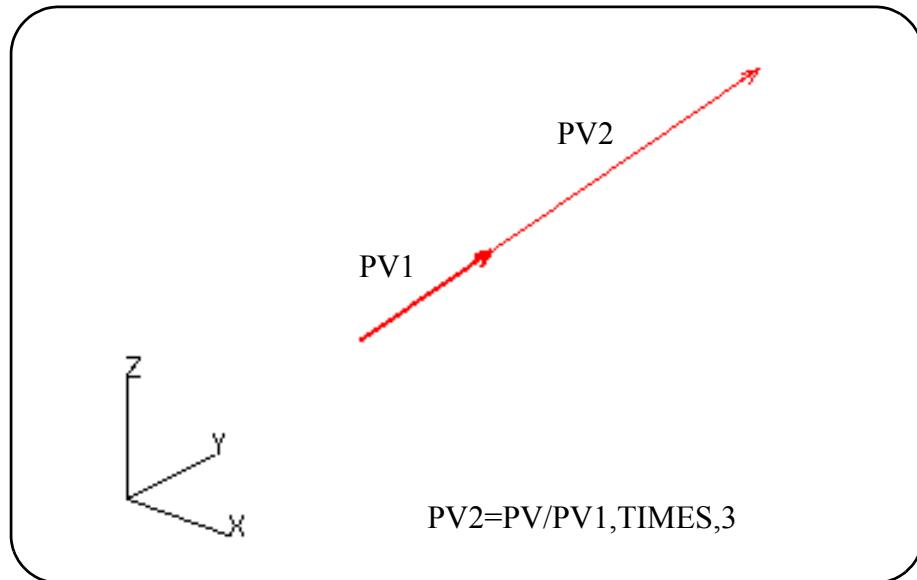
- The direction of the generated point-vector is perpendicular to the plane upon which the two vectors (temporary and specified) are defined using the "right hand" rule.
- The origin of the specified point-vector will be used as the origin of the generated point-vector.

3.7.18 A Point-Vector Multiplied By A Scalar

Command Syntax:

```
PNTVEC/point-vector,TIMES,magnitude-scalar
```

Icon Menu Sequence:



PV2 has the same direction as PV1 and a magnitude equal to PV1 TIMES the magnitude scalar.

Calculation Method:

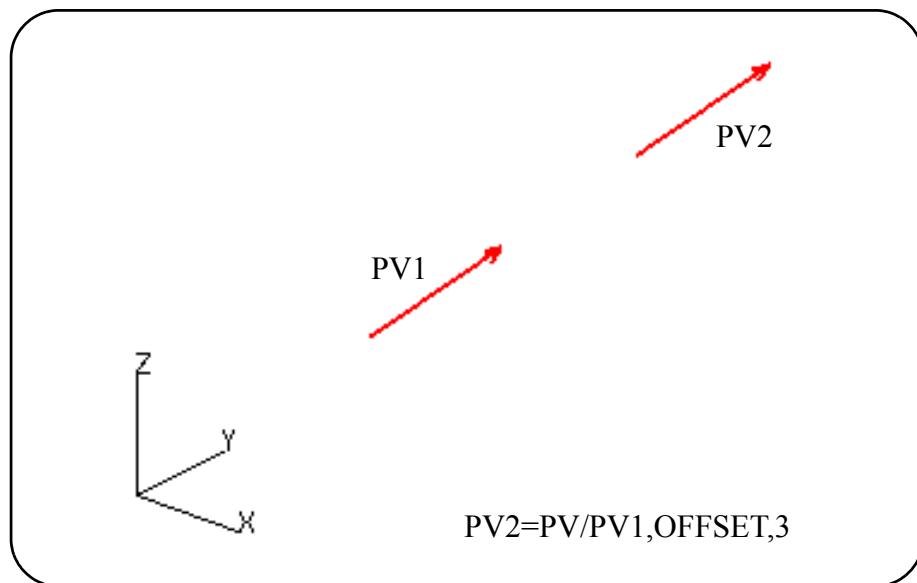
- The x, y and z-coordinates of the specified point-vector will each be multiplied by the magnitude-scalar to produce the defined point-vector's magnitude.
- The direction of the new point-vector will be the same as the specified vector.

3.7.19 A Point-Vector Offset By A Scalar

Command Syntax:

PNTVEC/point-vector,OFFSET,scalar

Icon Menu Sequence:



PV2 has the same direction and magnitude as PV1 but its origin is offset in the same direction by the amount of the scalar quantity 3.

Calculation Method:

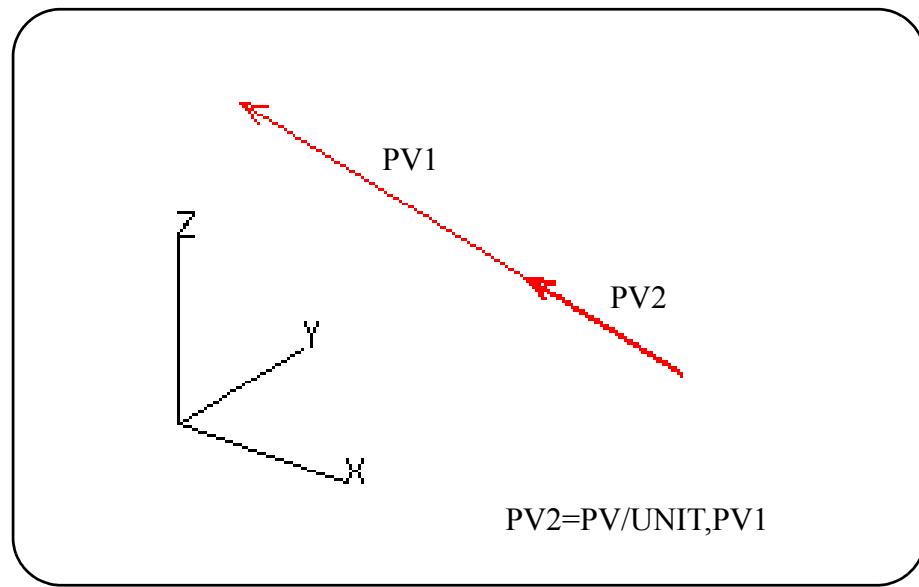
- The origin of the new point-vector will be offset in the direction of the specified point-vector by the amount of the specified scalar.
- The direction of the new point-vector will be the same as the specified vector.

3.7.20 A Point-Vector Which Is the Result Of Unitizing Another Point-Vector

Command Syntax:

PNTVEC/UNIT,point-vector

Icon Menu Sequence:



Calculation Method:

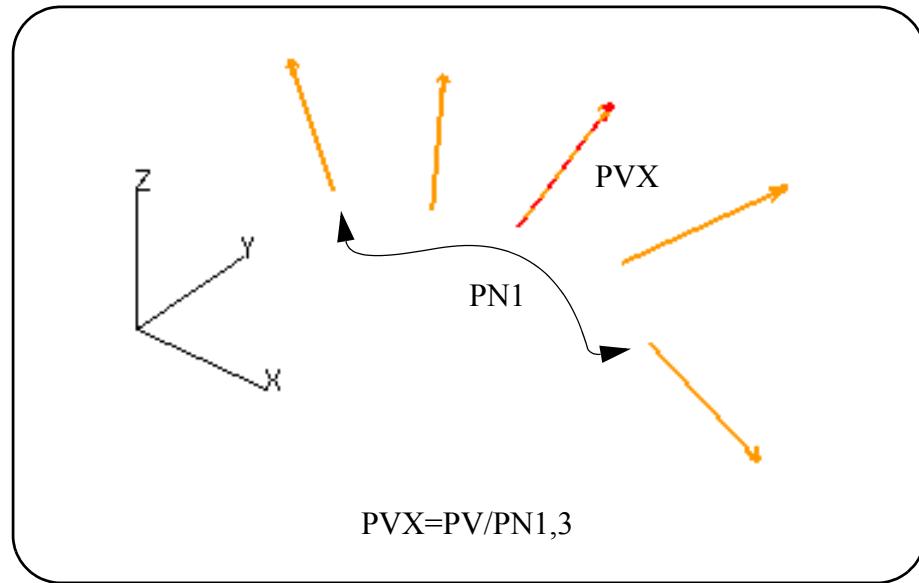
- The magnitude of the point-vector is one unit long.
- The direction of the point-vector is the same as the referenced point-vector.

3.7.21 A Point-Vector As A Point-Vector In A Pattern Of Point-Vectors

Command Syntax:

PNTVEC/point-vector-pattern, index

Icon Menu Sequence:



Calculation Method

- The point-vectors in a point-vector pattern are assigned a number in the order the point-vectors were generated. This example selects the 3rd point-vector of PN1 and creates PVX from it.

Error Condition:

- The index must be a positive number within the range of the number of point-vectors in the pattern.

3.7.22 A Point-Vector Normal To A Surface Obtained By Projecting A Point/Point-Vector Onto The Surface Along An Optional Vector/Point-Vector And An Optional Near Point/Point-Vector

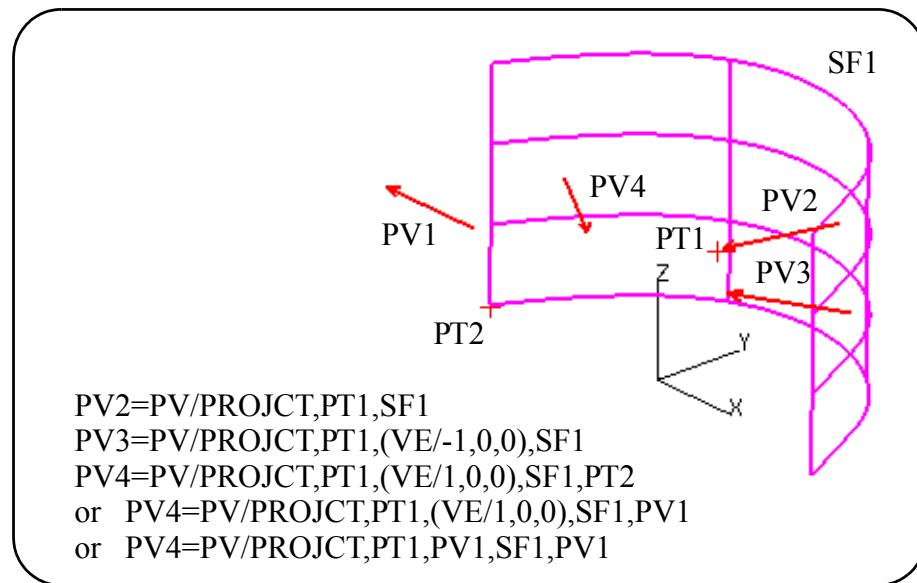
Command Syntax:

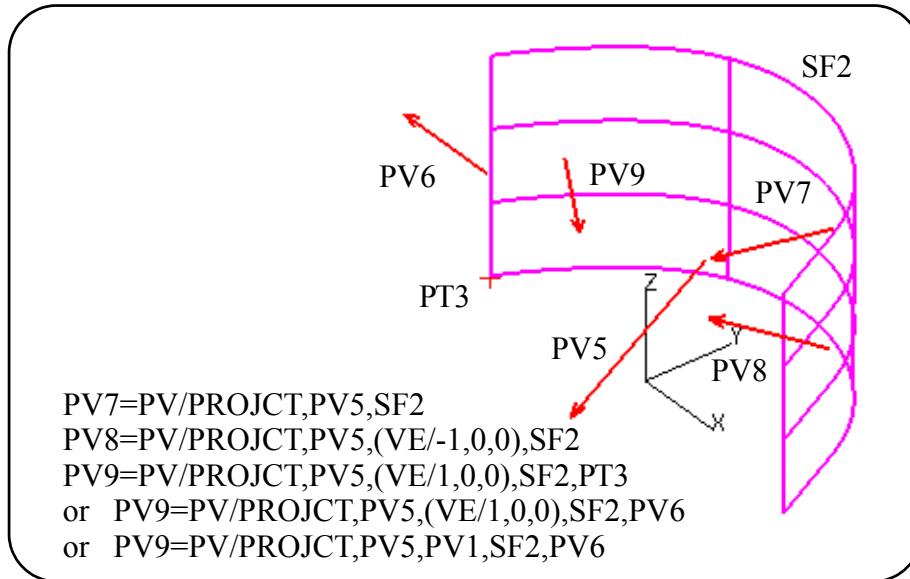
```
PNTVEC/PROJCT,point [,vector],surface[u,v]      $
          pntvec  pntvec
          [,near-point ]
          pntvec
```

Icon Menu Sequence:



NOTE: The optional near point/point-vector is used to determine which of several possible projected points is the desired point to create the point-vector.





Calculation Method:

- The referenced point (or the point-vector origin) will be projected onto the referenced surface closest to the optional near-point/point-vector along the optional vector/point-vector direction (either positive or negative).
- If the optional vector/point-vector is not specified, projection will be along normal to the surface from the referenced point/vector.
- If the optional near point/vector is not specified, **NCL** will look for the projected point starting from U=0.5 and V=0.5 on the surface. The default state can be changed by specifying a different U and V values.
- Calculation of the projection point will use the surface extension if necessary.
- The created point-vector will have a length of one inch regardless of **UNITS**. It is always normal to the surface and point into the direction of the referenced point/vector.

Error Condition:

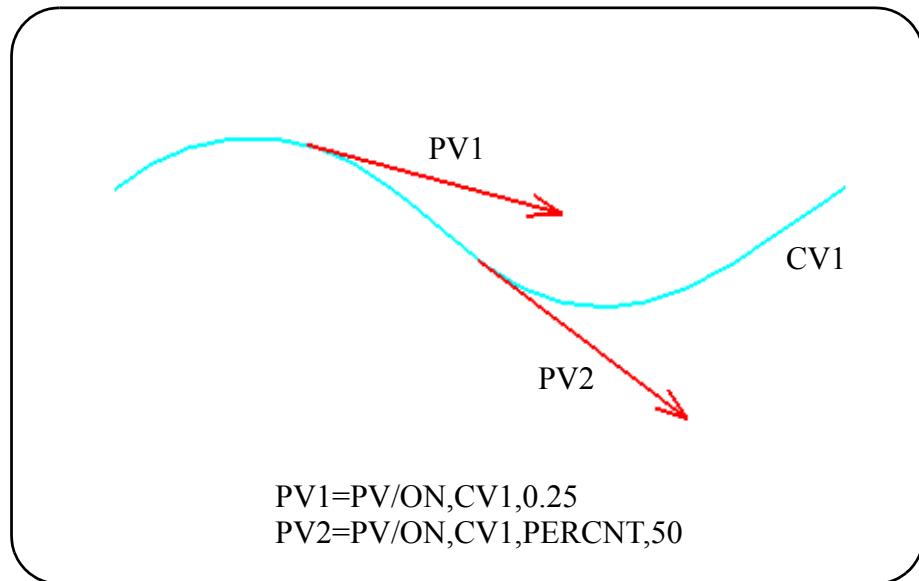
- The optional projection vector/point-vector starting from the referenced point (or the point-vector origin) must intersect the surface.

3.7.23 A Point-Vector From A Point On A Wire Frame Entity At A Specified UV or Percent Values And Tangent To The Wire Frame Entity

Command Syntax:

```
PNTVEC/ON, circle, [PERCNT, ]per
curve
line
```

Icon Menu Sequence:



Where:

PERCNT - An optional parameter to denote the “per” value is specified as a percentage along the physical length of the curve.

per - Is a number in the range 0 to 100 if PERCNT specified, otherwise is a number in the range 0 to 1 which represents

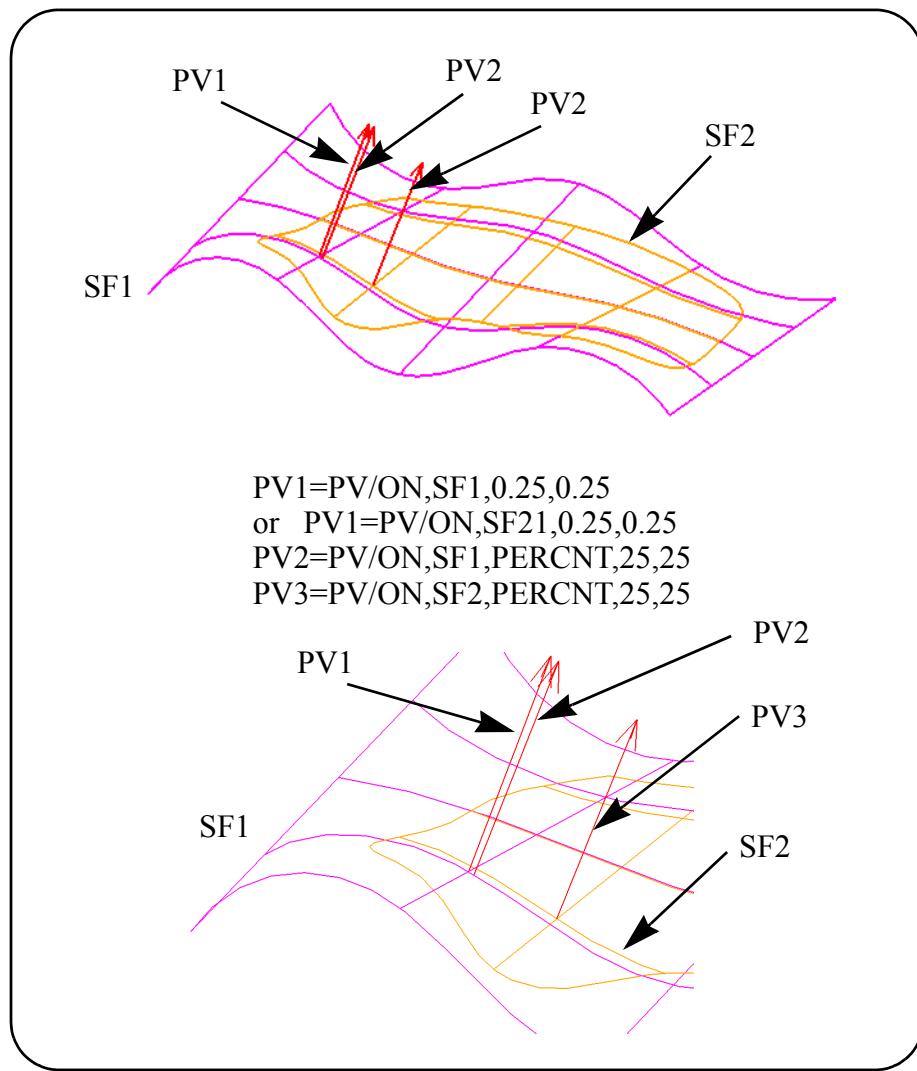
the U value along the wireframe entity at which to create the point. Entering 0 would create a point at the beginning of the entity while entering 1 or 100 would create a point at the end of the entity.

3.7.24 A Point-Vector Normal To A Surface Obtained By Specifying The UV Or Percent Values

Command Syntax:

```
PNTVEC/ON, surface, [PERCNT, ], uper, vper
```

Icon Menu Sequence:



Where:

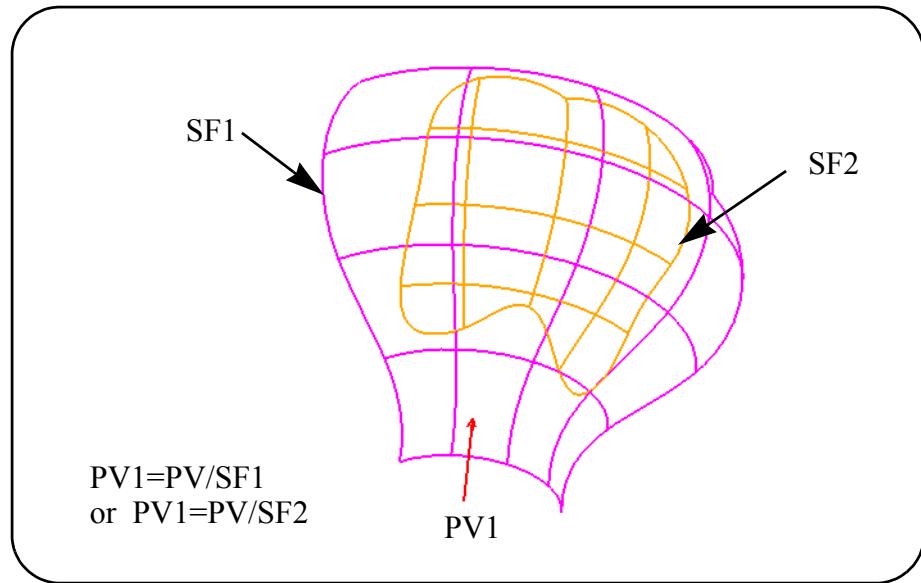
- sf - Name of the surface on which to create the point-vector.
- PERCNT - An optional parameter to denote the “uper, vper” parameters are specified as a percentage along the length (average) of the surface U and V line parametric curves instead of UV values.
- uper - Is a number in the range 0 to 1 (or 0 to 100 for PERCNT) which represents the U or Percentage value of the surface at which to create the point-vector.
- vper - Is a number in the range 0 to 1 (or 0 to 100 for PERCNT) which represents the V or Percentage value of the surface at which to create the point-vector.

3.7.25 A Point-Vector Along The Axis of A Surface Of Revolution And At The Center Of The Base

Command Syntax:

PNTVEC/surface

Icon Menu Sequence:



Calculation Method:

- The surface must be a revolved surface or a surface with either cone or cylinder as the primitive type. Surface with sphere as the primitive type is excluded.
- The point-vector is along the axis of revolution and at the center of the base.
- The base surface is considered if a trimmed surface is specified.

SURFACES

The following list gives an abbreviated notation of all the valid SURFACE definition formats. See the individual sections for a complete explanation of each format.

1. **NSURF**/point ,line *point not valid for NSURF
 SURF line circle
 circle curve
 curve
- or NSURF/line ,point
 SURF circle line
 curve circle
 curve
2. **NSURF**/line ,0 [...] ,line ,0
 SURF circle point circle point
 curve vector curve vector
 line line
 circle circle
 curve curve
 point-vector point-vector
3. **SURF**/XLARGE,plane,XLARGE,plane,RADIUS,st-rad \$
 XSMALL XSMALL
 YLARGE YLARGE
 YSMALL YSMALL
 ZLARGE ZLARGE
 ZSMALL ZSMALL

 [,en-rad,]point point
 point-vector point-vector
4. **SURF**/FILLET,surface,surface,near-point , \$
 plane point-vector

 radius-curve [,limit-curve]
 st-rad[,en-rad] patern

 or **SURF**/FILLET,surface,surface,near-point , \$
 plane point-vector

 radius-curve [,limit-curve]
 st-rad[,en-rad] patern

5. **NSURF/FIT**,curve[...],curve,curve
or **SURF/FIT**,curve [...],curve ,curve
 line line line
 circle circle circle
6. **SURF/OFFSET**,sf,vector,distance
 XLARGE
 XSMALL
 YLARGE
 YSMALL
 ZLARGE
 ZSMALL
7. **SURF/OUT**,trimmed-surface
8. **SURF/sf**[...] [,THRU,sf] [,sf]
9. **SURF/REDEF**[,sf],OUT,cv[,IN,cv[[...],cv]
 pl
10. **NSURF/REVOLV**,line ,point-vector \$
 SURF circle point, vector
 curve
 [,start-angle,end-angle]
11. **NSURF/EDGE**,interp,cv-U0,cv-V1,cv-U1,cv-V0
 SURF ci ci ci ci
 ln ln ln ln

Example Surface Expressions:

```
SF/LN1,CI3  
SF/PT2,LN5  
SURF/CV1,CV2,CV3,CV4  
SURF/CV1,0,CV2,0,CV3,0,CV4,0  
SF/CV1,THRU,CV4  
SF/1,THRU,4  
SF/XL,PL1,ZL,PL2,RA,1.5,PT1L,PT2L  
SF/XL,PL1,YS,PN2,RADIUS,.05,PV1,PV2  
SF/CV5,CV6,LN2,CI4  
SF/CV1,LN1  
SF/FIT,CV1,THRU,CV20  
SF/OFFSET,SF3,ZLARGE,3.5  
SF/SF1,SF3,THRU,SF2,SF5  
SF/FILLET,SF1,SF2,PT1,R1,R2  
SF/FILLET,SF4,SF2,PT1,CV3,CV7  
SF/FILLET,SF1,SF2,PV1,.1,.4,CV2  
SF/FILLET,SF1,SF2,PV1,CV4,CV5  
SF/SELECT,SFM23,4,12  
SF/SELECT,SFM24,7  
NSURF/CV1,PV1,LN1,PV2,CV2,PV3  
NSURF/CV1,THRU,CV4  
NSURF/FIT,CV1,CV2,CV3  
NSURF/REVOLV,CV1,PT1,VE1,0,270  
NSURF/REVOLV,CV1,PV2,0,30  
NSURF/EDGE,2,CV1,CI3,CV2,LN1
```

3.8

SURFace Expressions

A **NCL** surface may range from a simple ruled surface to a complex sculptured surface. All surfaces are bounded by their definition and have tangent slope extensions that extend to infinity. A surface is defined by providing **NCL** with the boundary geometry and slope constraints. **NCL** generates a surface between the boundaries and stores it in its mathematical form.

NCL supports three different types of surfaces. The first one is the native **NCL** Surface or a Ruled Surface. The second one is the Non-Uniform Rational B-Spline Surface. The third one is a Surface of Revolution.

The SURF command creates a native **NCL** Surface or a Ruled Surface which is a Bézier surface. **NCL** will process this type of surface (geometry definitions, tool motion, etc.) faster than the Non-Uniform Rational B-Spline Surface.

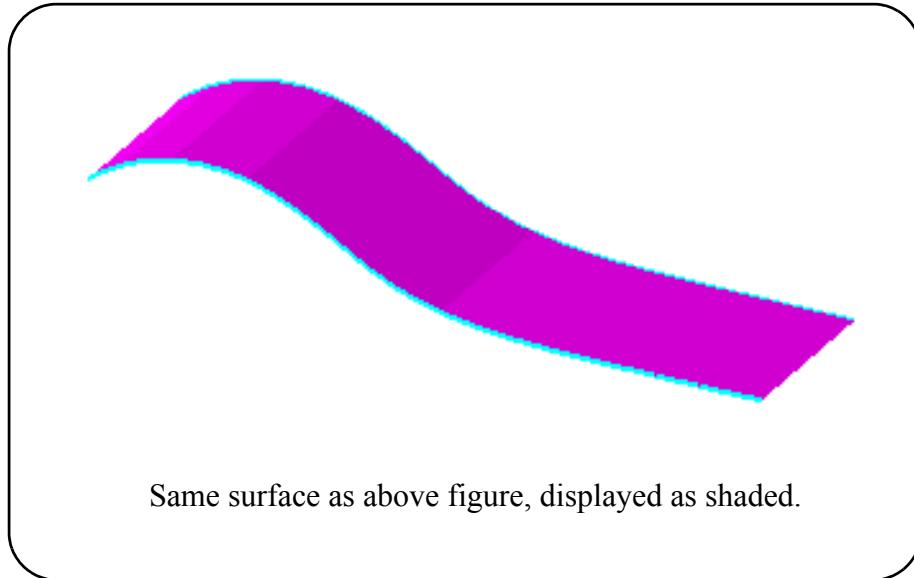
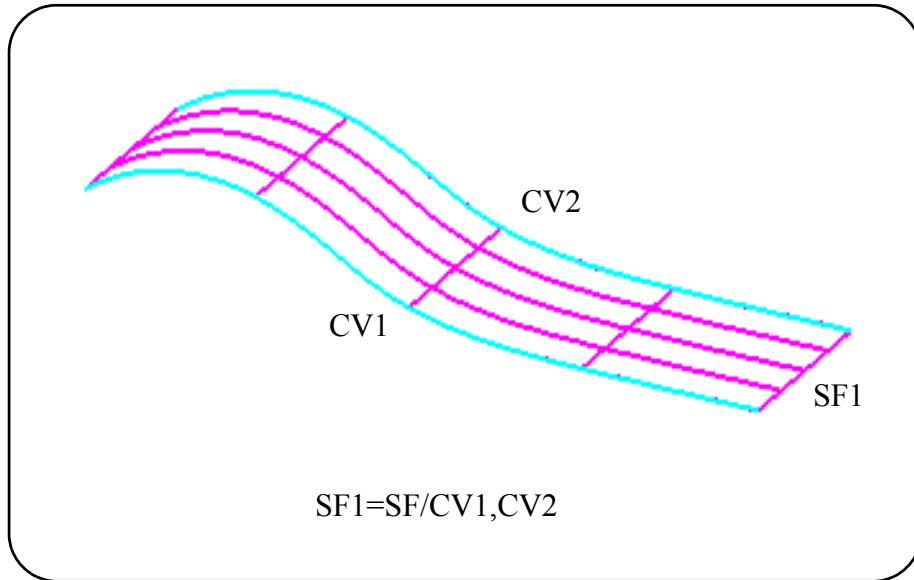
The NSURF command creates a Non-Uniform Rational B-Spline Surface (NURBS). NURBS have become the de facto standard for surface representation in most modern CAD/CAM systems. The advantage of NURBS is that they allow more precise data translation between systems and within **NCL** allow more complex surfaces to be defined. The disadvantage when compared to native **NCL** surfaces is that operations performed on NURB surfaces (geometry definitions, tool motion, etc.) will process slightly slower.

The **NCL** type Surface of Revolution applies only to a surface of revolution. This type of surface is generated by the commands [SURF/REVOLV](#), or [NSURF/REVOLV](#).

Surfaces can be displayed either as SHADED or UNSHADED. Only the UV lines of the surfaces will be displayed if displayed as UNSHADED. The default condition can be changed by setting the parameter “UU_SHADING” in the file: \NCCS\NCL100\interface\ncl.init. If this parameter is set to ON, the default display of surfaces will be SHADED. If it is set to OFF, the default surface display will be UNSHADED. If the default condition is set to SHADED, the shading attribute of each individual surface can be modified by clicking the interface icons as follows: TOOLS > CAM > SURFACE > SHADED SF or UNSHADE SF or MOD ATTRIB.

After version 9.1 the direction of the wire frame boundary used to create a surface (except surface creation with slope control) is not important. **NCL** automatically reverses the boundary curves to avoid creating “Bow-tie” surfaces. This automatic feature can be turned off/on by using the command “[*SET/VER,n](#)” with n set to less than 9.1 for OFF and n set to greater or equal to 9.1 for ON. This feature can

also be changed by entering the commands “*SET/SCHECK” for ON and “*RESET/SCHECK” for OFF, or by clicking TOOLS > CAM > SURFACE > MODALS.



Canonical Form:

Due to the complexity of the data involved, the canonical form of an **NCL** surface is not available to the user.

The only surface parameters which can be accessed by users are:

a. Primitive Field:

You can access this field by using the SRFTYP(surface) command as follows:

$A = \text{SRFTYP}(SF1)$

where: A is a scalar. **NCL** will retrieve the value from the “primitive” field of SF1’s structure and assign a corresponding integer to A. The value of A would be one of the following:

- 0 no primitive analysis has been performed
- 1 a free form surface
- 2 a ruled surface
- 3 a planar type surface
- 4 a spherical type surface
- 5 a cylindrical type surface
- 6 a cone type surface

b. Prim-param fields:

This field contains an array of real numbers which give the corresponding key parameters that describe the “Primitive Field” data. They are as follows:

For a planar type surface:

- | | |
|----------------------|---|
| param(1) to param(3) | i, j, k values of the planar surface normal. |
| param(4) | the distance from the coordinate origin to the planar surface with the current REFERENCE System . |

For a spherical type surface:

- | | |
|----------------------|---|
| param(1) to param(3) | x, y, z values of the spherical surface center point. |
| param(4) | The radius of the spherical surface. |

For a cylindrical type surface:

- | | |
|----------------------|--|
| param(1) to param(3) | x, y, z values of the cylindrical surface bottom center. |
| param(4) to param(6) | i, j, k values of the cylindrical surface axis. |
| param(7) | The radius of the cylindrical surface. |
| param(8) | The height of the cylindrical surface. |

For a cone type surface:

param(1) to param(3)	x, y, z values of the cone surface apex point.
param(4) to param(6)	i, j, k values of the cone surface axis vector pointing from the apex point to the cone base.
param(7)	The angle between the cone surface side lines to its axis.
param(8)	The height of the cone surface.
param(9)	Distance between the cone apex point and the top of the actual cone surface

These parameters can be accessed by using the **OBTAIN** or **CAN** statements in the normal manner.

MAGENTA (pen 6) is the **NCL** System DEFAULT color for CAM surfaces.

BROWN is the **NCL** System DEFAULT color for **NCL** generated net surfaces.

3.8.1 A Surface Constructed From Two Geometric Entities Using System-Generated Slope Constraints

Command Syntax:

```
NSURF/point ,line
SURF  line   circle
      circle spline
      spline curve
      curve
```

or

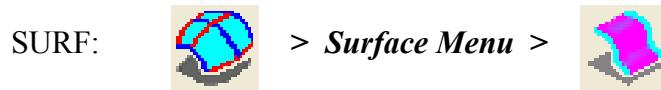
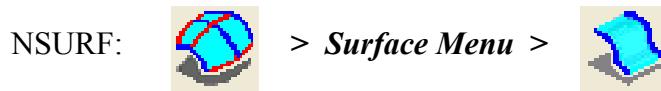
```
NSURF/line ,point
SURF  circle line
      spline circle
      curve   spline
      curve
```

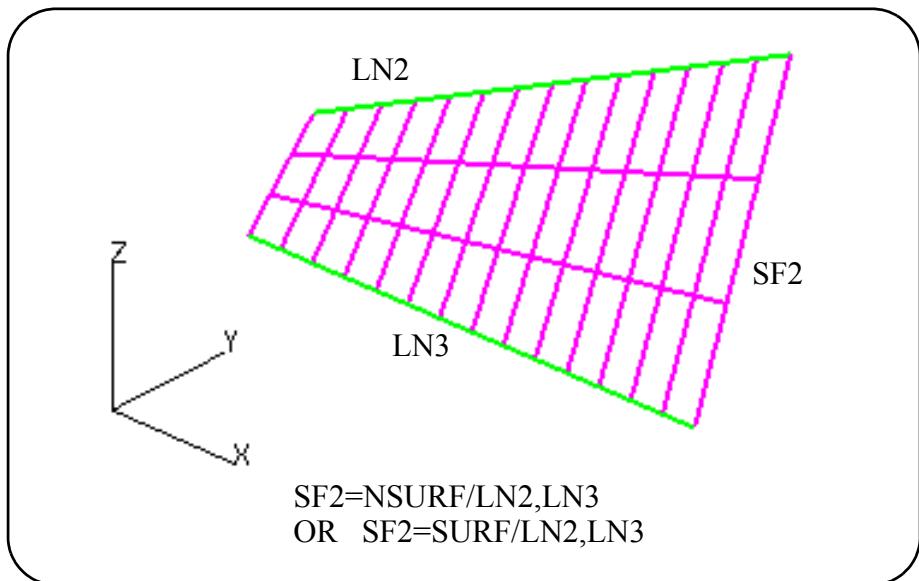
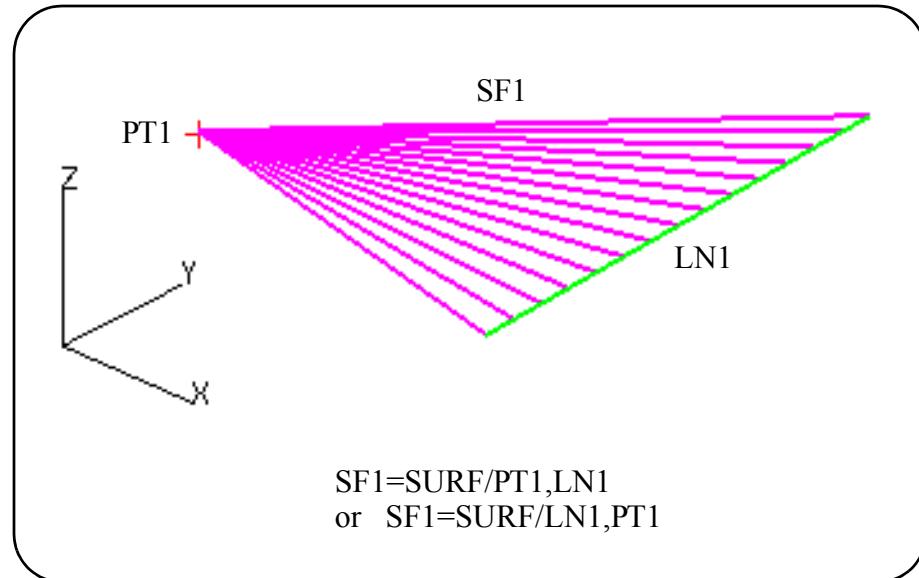
where:

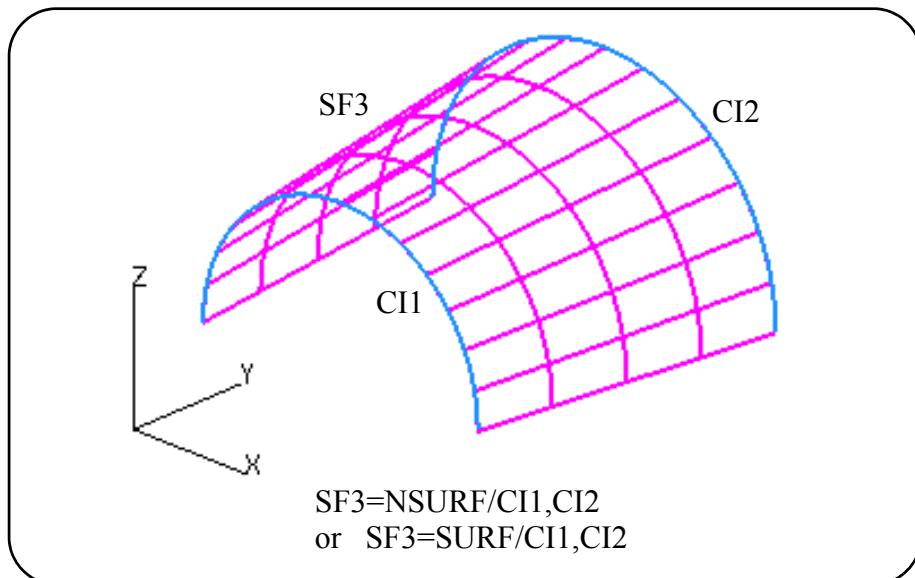
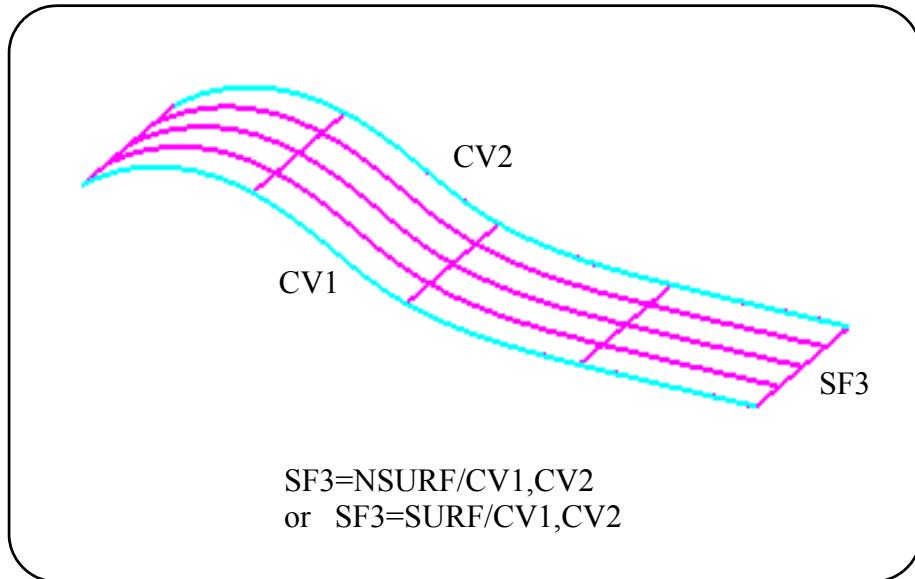
NSURF Specifies to create a Non-Uniform Rational B-Spline Surface. This specifier allows B-Spline type curve as input and does not allow Point type entity as input.

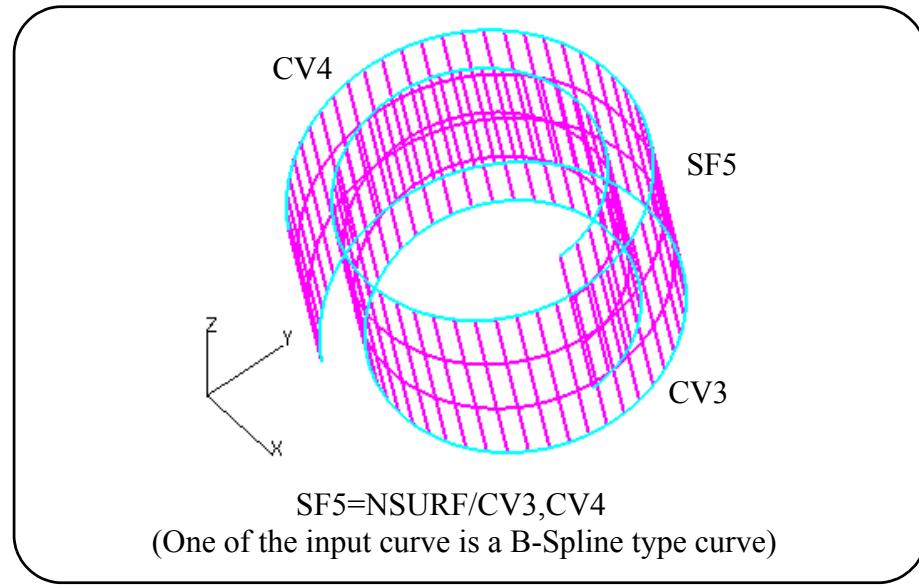
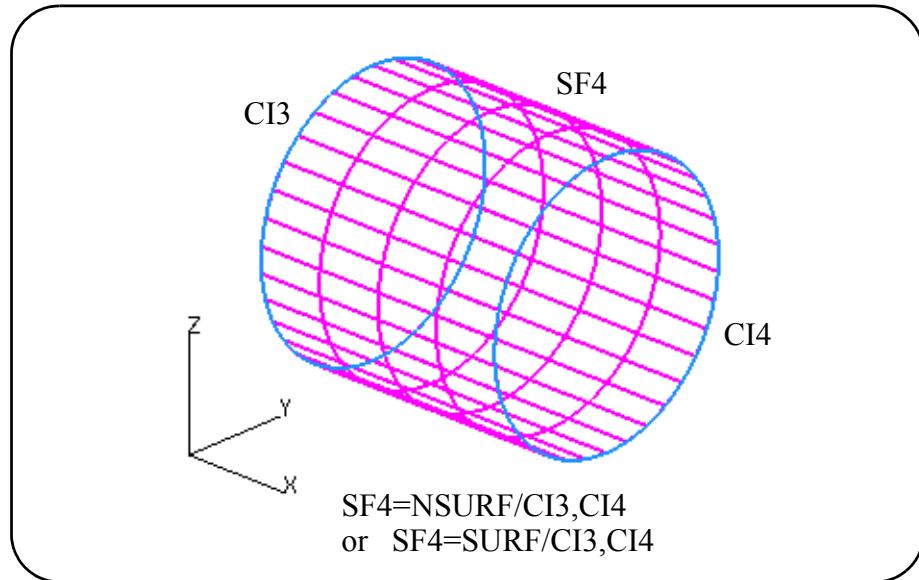
SURF Specifies to create an **NCL** Type Native Surface. This specifier does not allow B-Spline type curve as input.

Icon Menu Sequence:







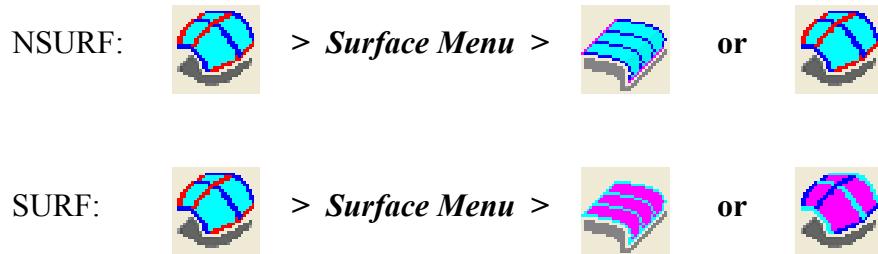


3.8.2 A Surface Constructed Through A Series Of Geometries With Slope Control

Command Syntax:

zero(0)	zero(0)
point	point
NSURF/line	vector
SURF	circle, line, [...], circle, line
	spline circle
	curve spline
	curve
	pntvec
	line vector
	spline circle
	curve spline
	curve
	pntvec

Icon Menu Sequence:



Calculation Method:

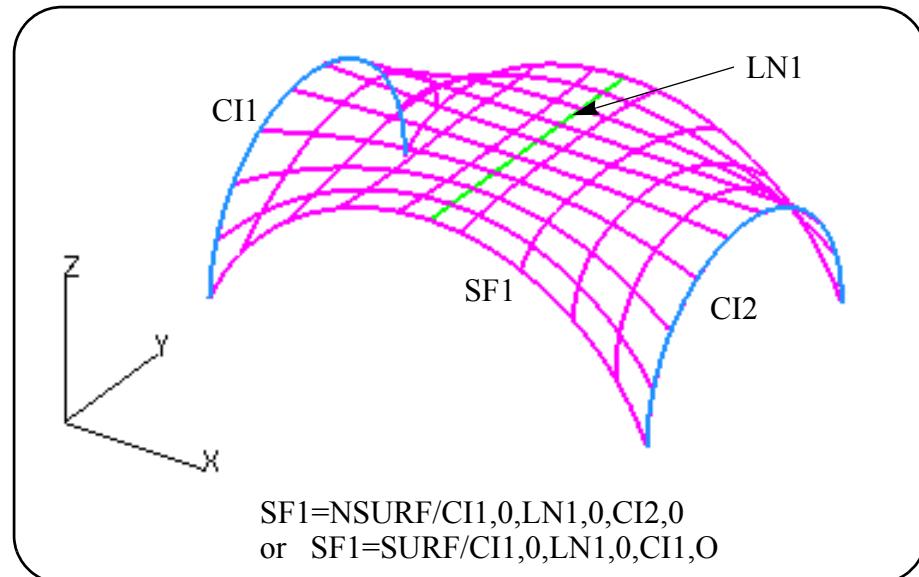
- NSURF specifies to create a Non-Uniform Rational B-Spline type surface. This specifier will allow B-Spline type curves as the input boundary or slope control entities. There is no limit to the number of boundary/slope couplets for this type of specification.
- SURF specifies to create an **NCL** native type surface. This specifier does not allow any B-Spline type curve as the input boundary or slope control entity. Only native **NCL** type curves are allowed. A maximum of 20 boundary/slope couplets is allowed.
- Every second entry specifies the slope constraint for the previous boundary entry. The system internally builds a ruled surface between the slope geometry and the boundary geometry. It then causes the surface being defined to be tangent to that ruled surface at the corresponding boundary geometry.
- The specifier “NSURF” is sensitive to the location or direction of the slope control entry with respect to both the current and next geometric boundary.

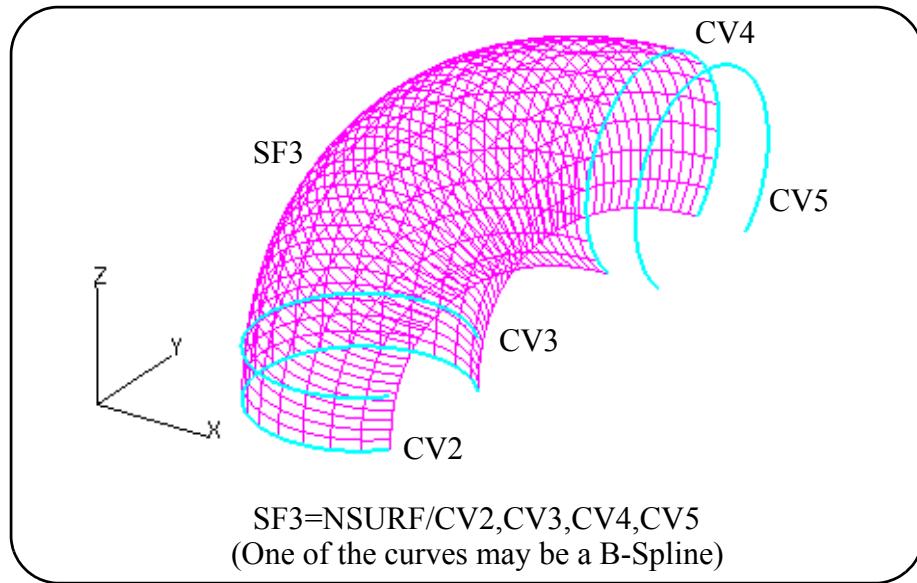
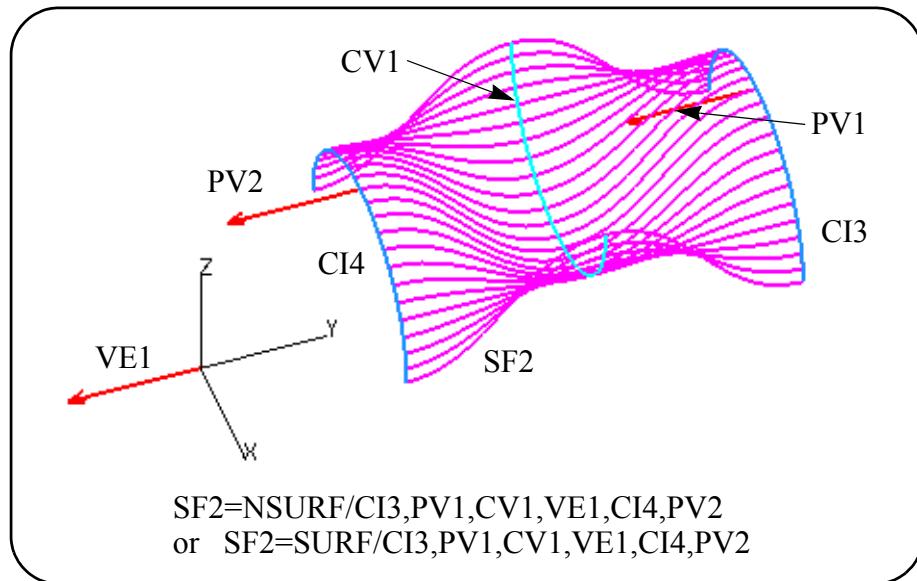
Different surfaces or unrealistic surfaces can be created if the slope control entry is not in the same general direction of the next geometric boundary with respect to the current geometric boundary.

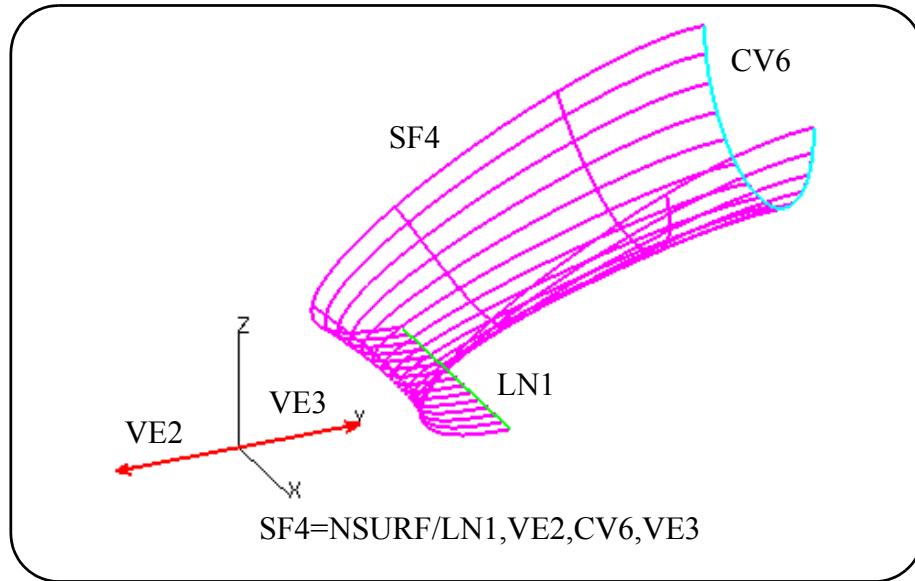
- The location or direction of the slope control entry does not have any effect for the specifier “SURF”. However, it is recommended that the slope control entry location and direction is always in the same general direction of the next geometric boundary with respect to the current geometric boundary.
- If the number “0” is specified, the system will generate the “best fit” slope constraint for the previous entry.

Error Conditions:

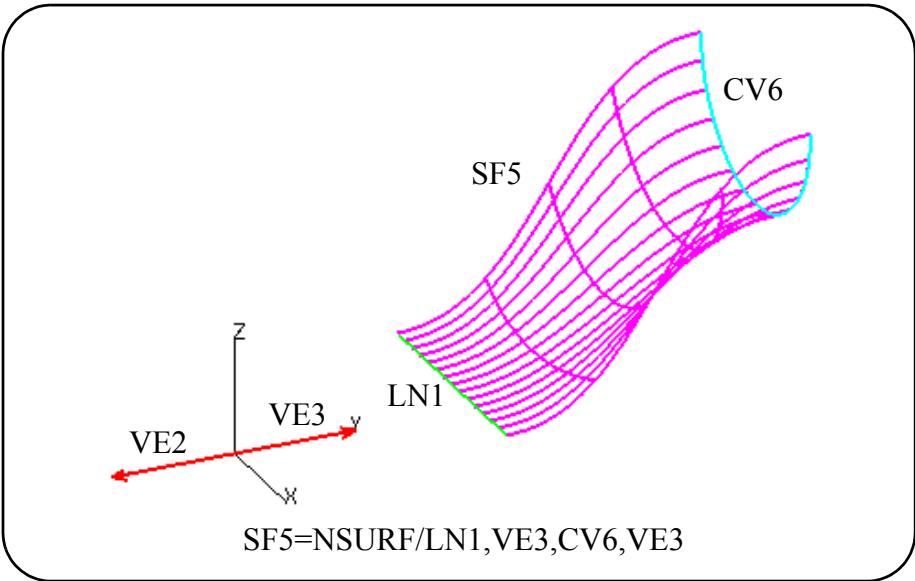
- There must be a minimum of 2 boundary/slope couplets.
- There is not an even number of entries.
- A B-Spline type curve is specified with the SURF specifier.
- More than 20 boundary/slope couplets are specified with the SURF specifier.
- The same entity is specified as both the boundary and the slope control within a boundary/slope couplet.

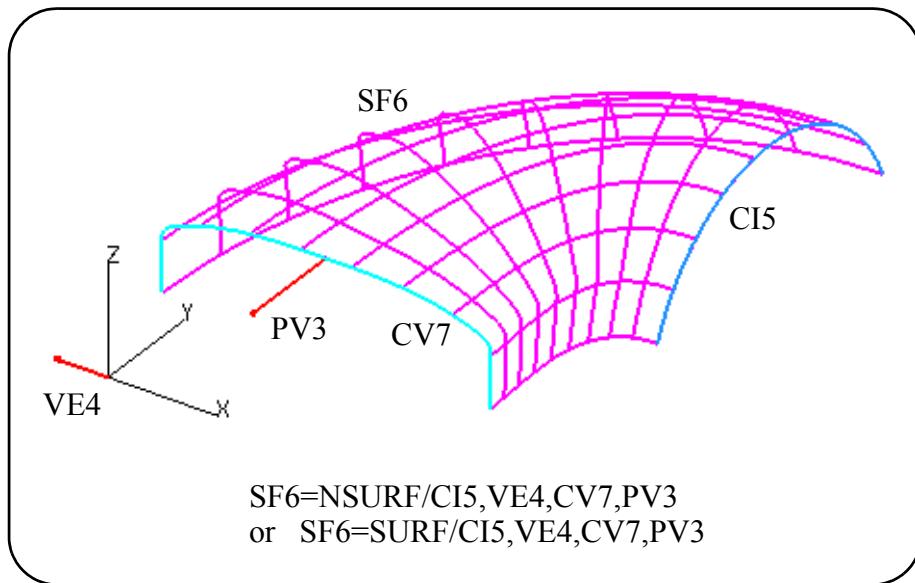






Same geometric boundaries except the first slope control entry points in different direction generate different surface.





3.8.3 A Fillet Surface Tangent To Two Intersecting Planes Using Point Or Point-Vector Boundaries

Command Syntax:

```

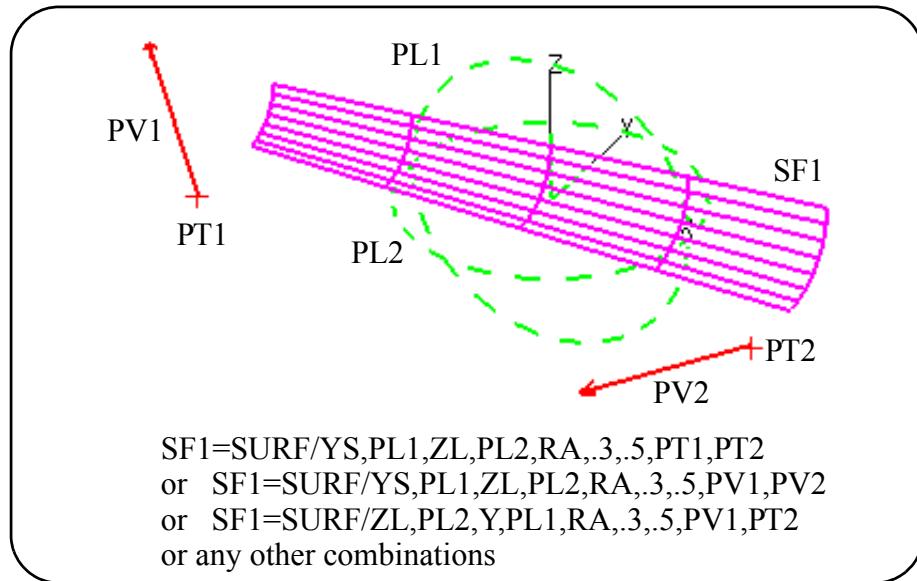
XLARGE      XLARGE
XSMALL      XSMALL
SURF/YLARGE,plane,YLARGE,plane,RADIUS,      $
YSMALL      YSMALL
ZLARGE      ZLARGE
ZSMALL      ZSMALL

start-radius[,end-radius],point      ,
point-vector                         $

point
point-vector

```

Icon Menu Sequence:



The fillet surface is bounded by two parallel planes. Each of the parallel planes is perpendicular to both of the intersecting planes and passing through a user-specified point or the origin of a user-specified point-vector. The radius of the fillet surface can either be a constant or change continuously from one end to the other end.

The start and the optional end radius values must be greater than the current **TOLER** setting. The distance between the two boundary entities must be greater than zero along the intersection of the two specified planes.

The two specified planes cannot be parallel.

3.8.4 A Fillet Surface Between Two Surfaces Or A Surface And A Plane

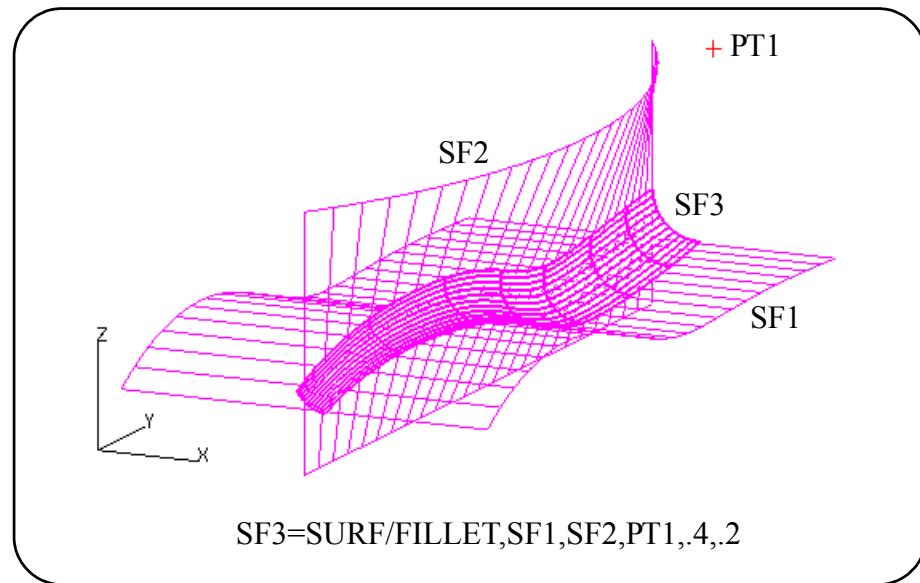
Command Syntax:

```
SURF/FILLET,surface,surface,near-point , $  
          plane           point-vector  
          radius-curve      [,limit-curve ]  
          start-radius[,end-radius] limit-pattern
```

or

```
SURF/FILLET,surface,surface,near-point , $  
          plane           point-vector  
          radius-curve      [,limit-curve ]  
          start-radius[,end-radius] limit-pattern
```

Icon Menu Sequence:



The fillet surface is bounded by two planes normal to the internally generated intersection curve of the two entities where the intersection ends. The radius of the fillet surface may be constant or variable. The radius throughout the surface can vary either linearly (incrementally from large to small or vice versa), or it can change based on the Y-values of an XY-curve. The fillet surface radius must be greater than the current **TOLER** setting anywhere along the created surface.

If the specified surface is of the type **TRIMMED**, the underlying surface will be used for the generation of the fillet surface.

One side of the fillet surface will be tangent to one of the specified entities or its extension. The other side will be tangent to the other entity or its extension. The two entities cannot have the same slope at the tangency points.

SF1 and SF2 are the two surfaces that the fillet surface is to be tangent to. The near-point (PT1 in this example) is a point used to indicate which side of the two defining surfaces the fillet surface is to lie on.

The near-point or point-vector origin will be projected normal to each defining surface and that point will define both the side and which intersection is to be used if the two surfaces intersect more than once. It will also be used to determine the start of the fillet surface. The limit point closest to the near-point or point-vector will be at the beginning of the fillet surface.

The radius of the fillet surface may be defined using one of three different methods:

1. A starting radius and ending radius with the radius varying in a linear manner between the two.
2. A single radius that does not vary over the entire surface.
3. A curve that defines the radius at each point along the "U" direction of the surface. The "U" direction goes from the starting point to the ending point ($U=0$ to 100%). The absolute Y value of the curve will be used to set the radius of the surface. The X value of the curve will be converted to a 0 to 100% distance along the "U" direction. At each percentage increment along the "U" direction that is used for internal calculations, the Y value corresponding to the X value will be used to set the radius of the fillet surface.

The start-radius is a scalar that specifies the radius of the fillet surface at the starting boundary of the surface ($U=0\%$).

The end-radius is a scalar that specifies the radius of the fillet surface at the ending boundary of the fillet surface (U=100%). If no end-radius is given, the value given for the start-radius will be used to define a fillet surface with a constant radius.

The starting and ending limits of the fillet surface may be specified using one of three methods:

1. A curve whose starting and ending points will be used as the limit points. These will be projected normal to the two defining surfaces and those points along with the given limit points will define planes at the "U" limit boundaries of the fillet surface.
2. A **PATERN** of points or point-vectors.
3. If no **CURVE** or **PATERN** is given, the curve defined by the intersection of the two defining surfaces will be used. This curve will be used as the curve in option #1.

The limit-curve is a curve that defines the limiting boundaries of the generated fillet surface as the end points of the limit-curve. This curve is also used instead of an internally generated curve which is the intersection of the two defining surfaces. The curve is stepped along to calculate the points in the tangency curve which are used as explained in the "Calculation Method" to follow. The limit-curve should be relatively close to the intersection of the two defining surfaces. Optimally, it should be a curve that contains the center points of all the circular arcs that describe the tangency points of the fillet surface and the two defining surfaces.

The two specified entities (surfaces/plane) do not need to be physically intersected as long as the extensions do. If they do not physically intersect or only partially intersect, a limit-curve/limit-pattern must be specified to use in place of an internally generated intersection curve for the start and ending boundary. See above paragraph about how to define an optimal limit-curve.

The extension of the entities will only be utilized in the possible direction of intersection. It will not be used in the non-intersecting direction. So do not specify a limit-curve or limit-pattern larger than all the possible intersections, otherwise, unexpected results may occur. The fillet surface will be generated according to the following calculation method.

Calculation Method:

- The start and end of the fillet surface are bounded by two planes each normal to the beginning and the end of the intersecting curve between the two surfaces, or the given limit curve.
- The sides of the fillet surface are bounded by two curves which are composed of points calculated as described next.
- A series of points along the intersection curve or the limit curve will be used to calculate the required points for generation of the side curves.
- For each of the points along this intersection curve or the limit curve, a plane normal to the curve at this point will be used to calculate two intersecting curves between the plane and the two surfaces. A circular arc with the required or corresponding radius at this point will be generated between these two curves. Finally all the generated circular arcs will be used to create the required fillet surface. If the two surfaces do not physically intersect, the extension of the surfaces will be used for the calculation.

Example fillet surface statements:

```
SURF/FILLET, SFA, SFB, NPT3, 1.5, 3.125
```

Defines a surface tangent to SFA and SFB on the sides of those surfaces closest to NPT3. It will have a starting radius of 1.5 that varies linearly to an ending radius of 3.125. The edges of the surface will stop at the ends of the curve defined by the intersection of SFA and SFB.

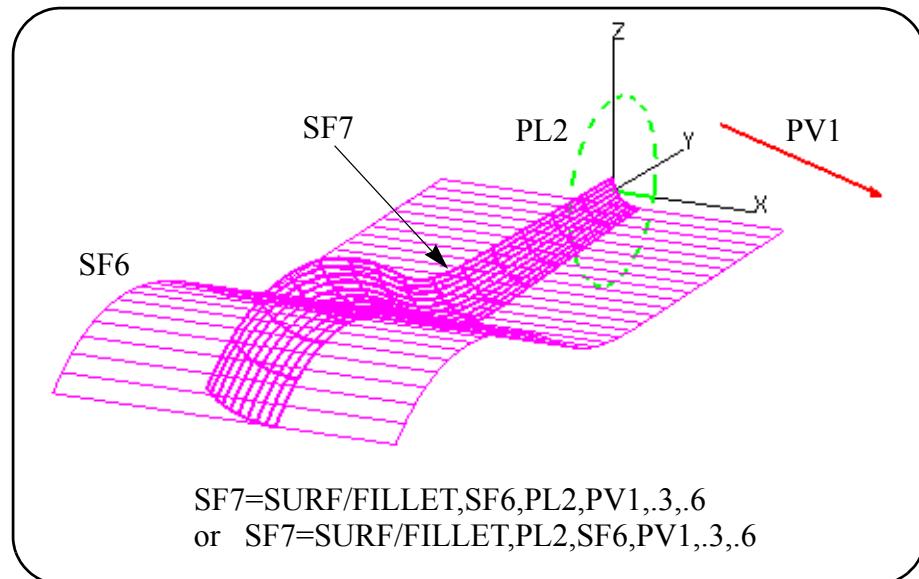
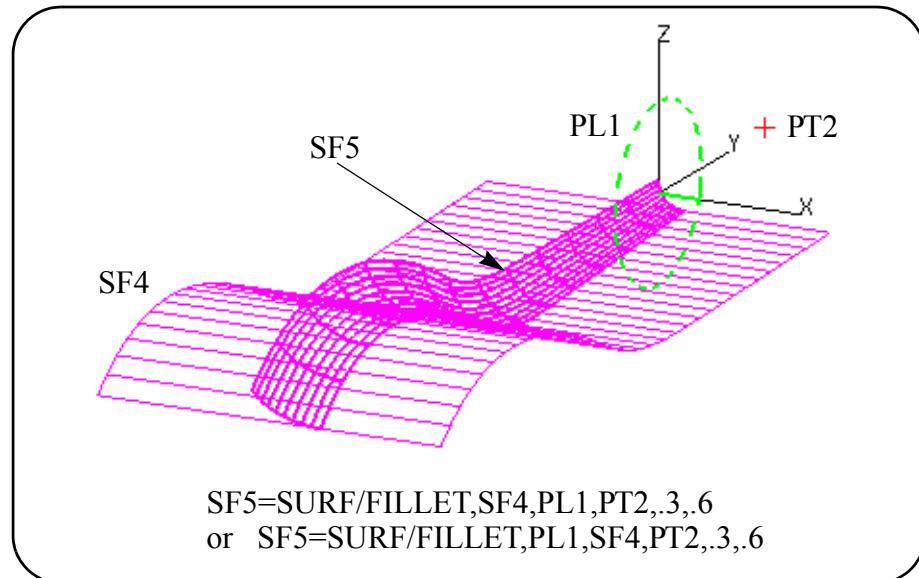
```
SURF/FILLET, SF3, SF4, PT2, CV1
```

Defines a surface similar to the one shown in the example at the beginning of this section. The surface is tangent to SF3 and SF4 on the sides of those surfaces closest to PT2. The radius will vary based on the Y value of the curve CV1. By default, the edges of the surface will stop at the ends of a curve defined by the intersection of SF3 and SF4.

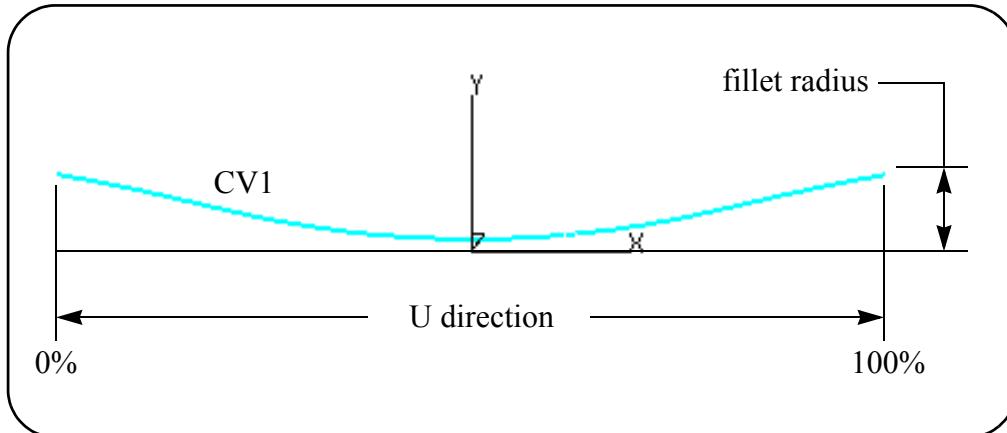
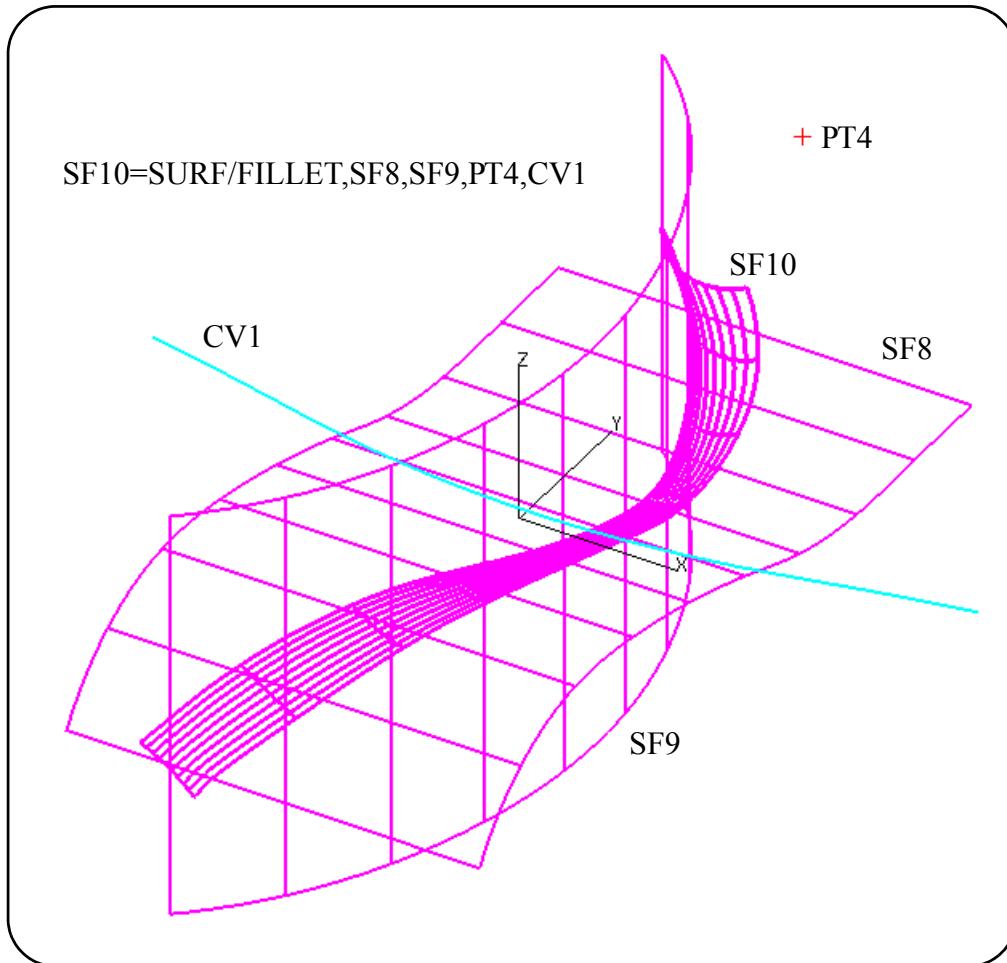
```
SURF/FILLET, PL1, SFA, NPT1, .3, .6
```

Defines a surface tangent to PL1 and SFA on the sides of those surfaces closest to NPT1. It will have a starting radius of .3 that varies linearly to an ending radius of .6. The edges of the surface will stop at the ends of the curve defined by the intersection of PL1 and SFA.

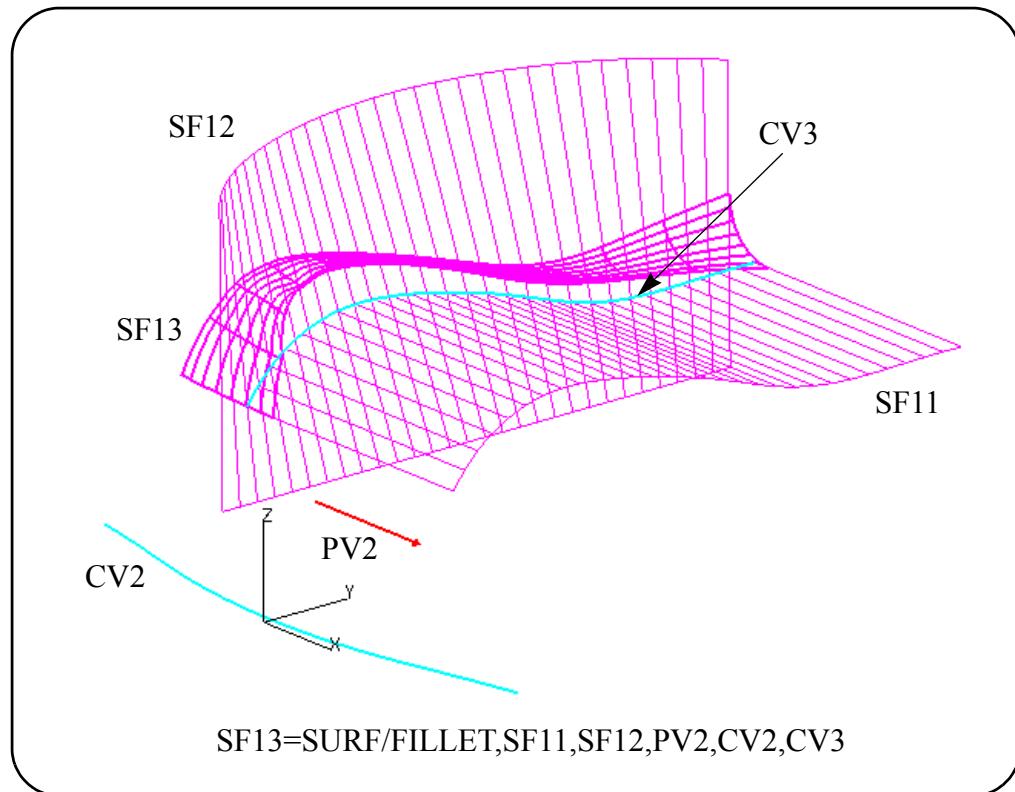
See figures on next page.



The following displayed surface uses two curves to create a variable fillet between two surfaces.



In this case, the two surfaces SF11 and SF12 do not intersect. Therefore, it becomes necessary to use the curve CV3 to indicate the boundaries of the defined surface SF13. The curve CV2, which lies on the XY-plane is used to indicate the variable fillet radii.



3.8.5 A Surface Defined By A Series Of Geometric Curves, Lines, Or Circles

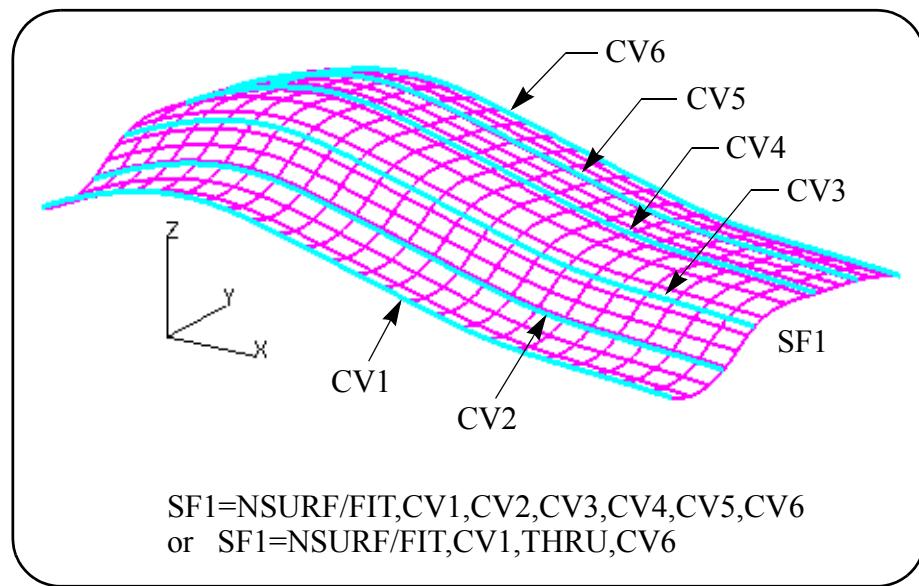
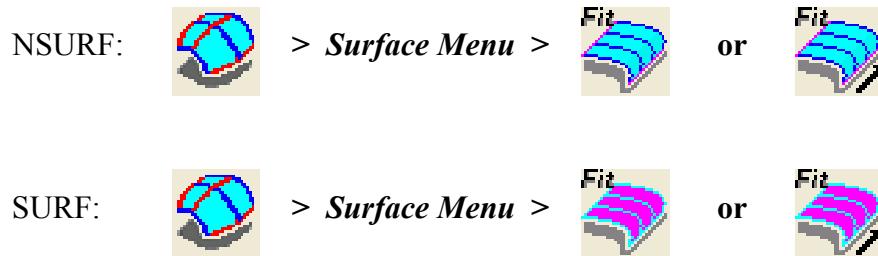
Command Syntax:

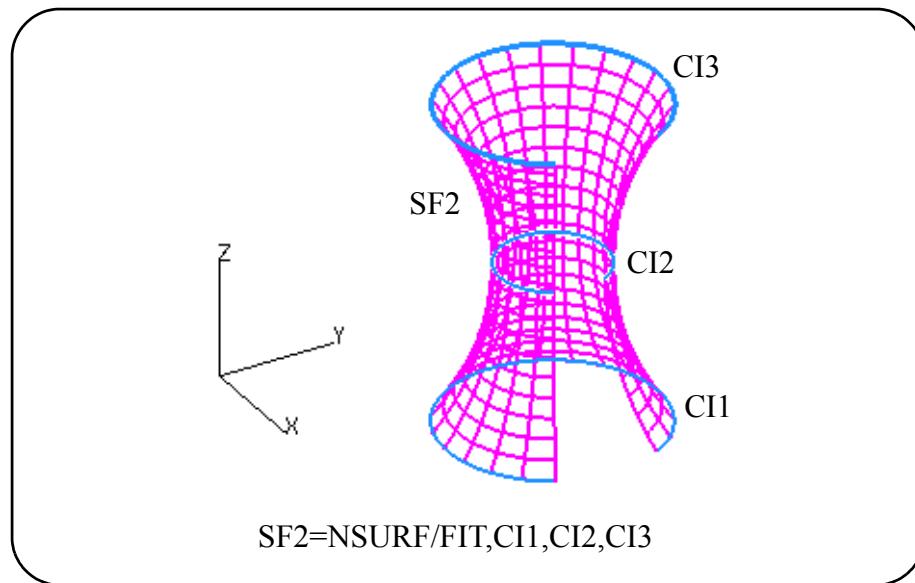
```
NSURF/FIT,spline,spline[...],spline
      curve   curve   curve
      line    line    line
      circle  circle  circle
```

or

```
SURF/FIT,curve,curve[...],curve
```

Icon Menu Sequence:





Calculation Method:

- NSURF specifies to create a Non-Uniform Rational B-Spline type surface. This specifier will allow all kind of wire frame geometries as input entities. While there is no limit to the number of input entities, **NCL** will attempt to eliminate as many input entities as possible from the final surface and still maintaining a specified tolerance (as specified in the **TOLER** statement) to all the input entities.
- SURF specifies to create an **NCL** native type surface. This specifier only accepts **NCL** native type curves as input boundaries. A maximum of 50 input curves will be allowed. If more than 20 input curves are specified, **NCL** will reduce the number to 20 from the final surface while still maintaining a specified tolerance (as specified in the **TOLER** statement) to all the input curves.
- A minimum of three input entities must be specified.
- If the input entities labels are in sequential order, the modifier **THRU** can be used instead of specifying each individual entities explicitly, such as “CV1, **THRU**, CV10”, etc.

Error Conditions:

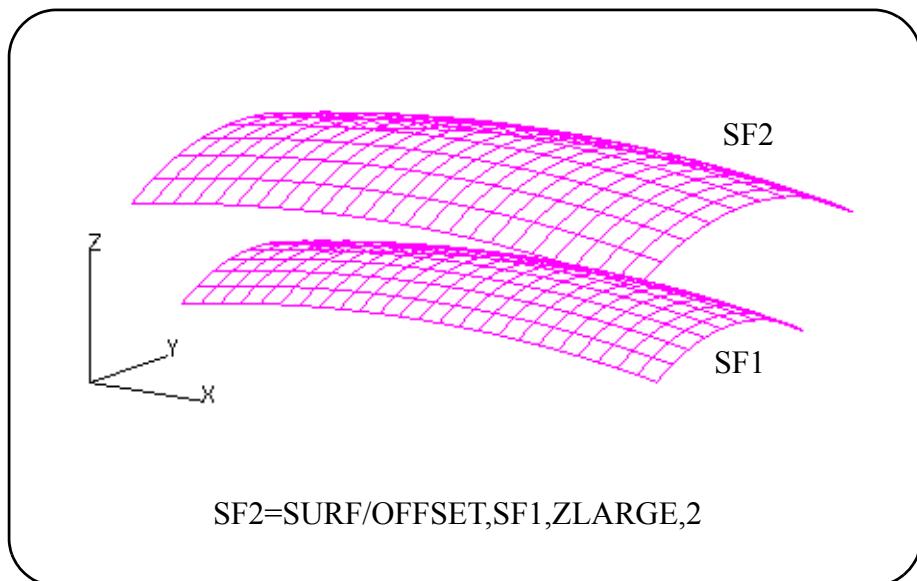
- There must be a minimum of 3 input entities.
- Other than native **NCL** type curve is specified with the SURF specifier.

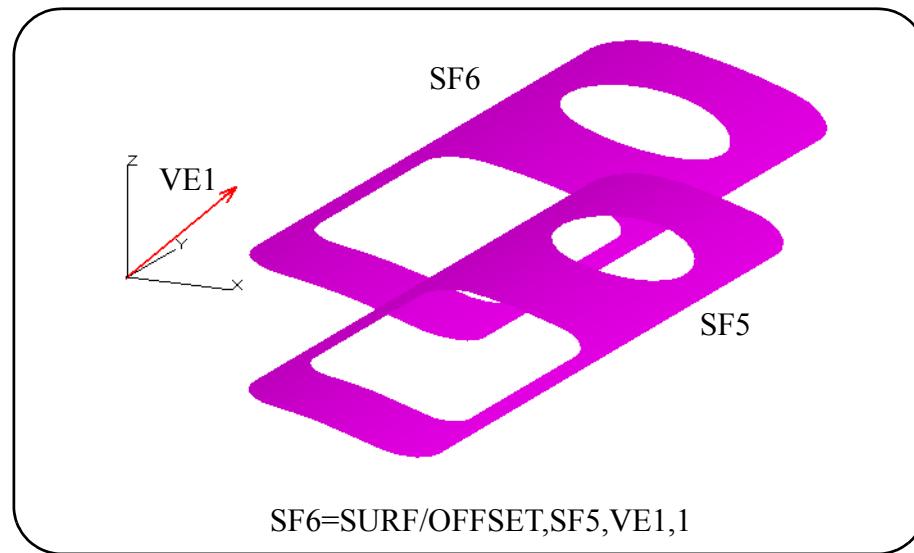
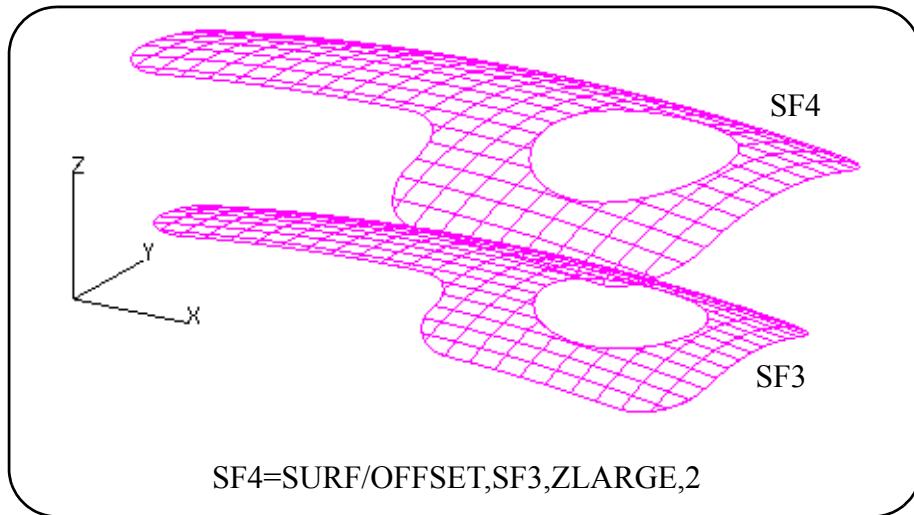
3.8.6 A Surface Offset From A Surface In A Direction At A Distance

Command Syntax:

```
vector  
XLARGE  
XSMALL  
SURF/OFFSET,surface,YLARGE,distance  
YSMALL  
ZLARGE  
ZSMALL
```

Icon Menu Sequence:





This surface definition creates a surface that is offset normal to the specified surface by an amount equal to the distance specified.

The vector or the direction-modifier is used to determine which of the two possible surfaces is the one desired.

The input surface may be any type except [QUILT](#) or [NET](#).

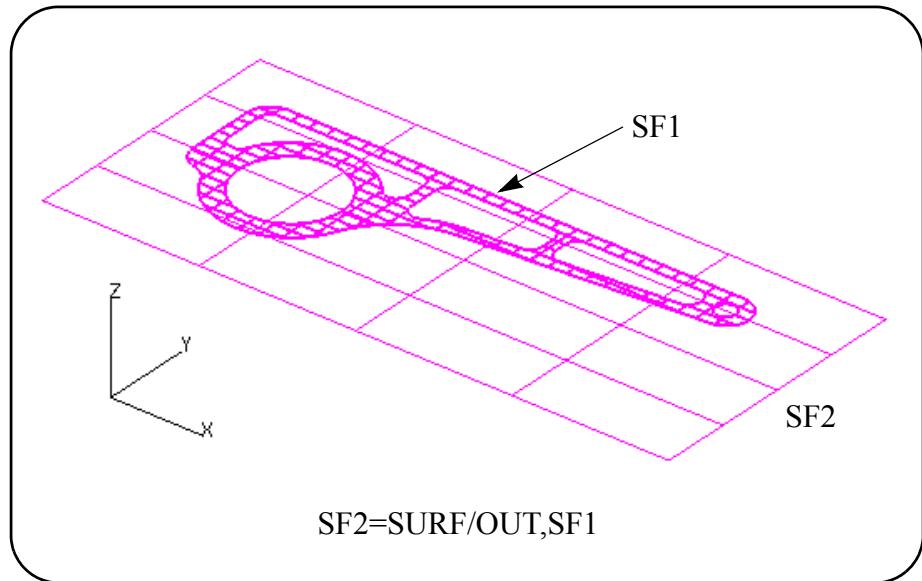
The created offset surface will be the same type as the input surface. For example, a native **NCL** type surface will generate a native **NCL** type offset surface, a Non-Uniform B-Spline surface will generate a Non-Uniform B-Spline offset surface, a trimmed surface will generate a trimmed offset surface.

3.8.7 A Surface Obtained By Untrimming A Trimmed Surface

Command Syntax:

```
SURF/OUT,trimmed-surface
```

Icon Menu Sequence:



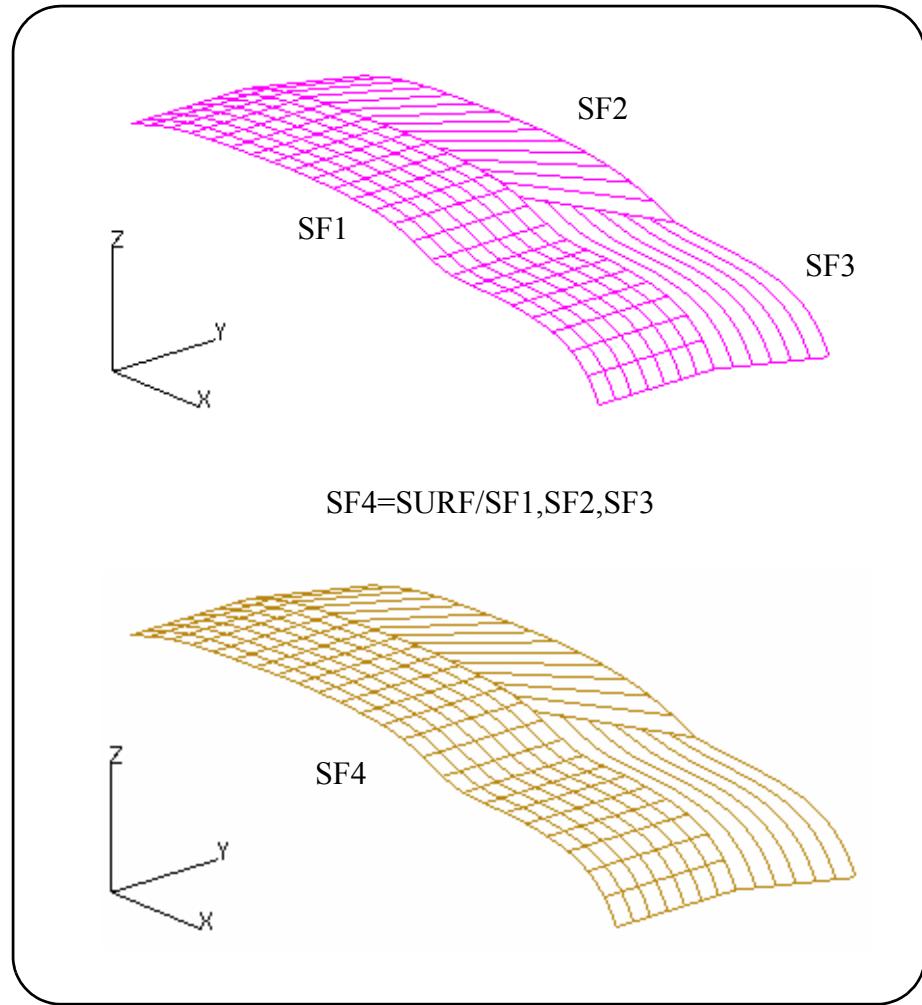
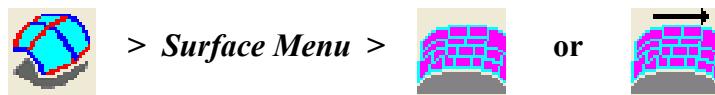
A trimmed surface is a surface that has been imported into **NCL** via **NCL/IGES** or a surface created by the [SURF/REDEF](#) command.

3.8.8 A Surface As A Collection Of A Number Of Other Surfaces

Command Syntax:

```
SURF/surface[...] [,THRU,surface] [,surface]
```

Icon Menu Sequence:



This surface definition creates one surface, called a NET surface, out of a number of other surfaces. The surfaces in the definition may not be **QUILT**, or other NET surfaces and the maximum number allowed is 40. Trimmed surfaces are allowed in a NET surface definition and netted surfaces do not have to be slope contiguous (tangent).

NET surfaces may be used in any geometry definition and motion statement with the following exceptions:

- **OFFSET** of a NET surface is not allowed.
- Edge curve of a NET surface is not allowed.
- Projection of curve(s) or Pattern(s) onto a NET surface is not allowed.
- **SCRUB** and **FMILL** motion is not allowed.
- When the NET surface consists of **TRIMMED** surface(s) is utilized as the drive surface, **NCL** will see the components of the NET surface as untrimmed and will probably not produce the desired results.

Driving the extensions of NET surfaces is permitted, but it can affect the system performance, especially when the NET surfaces involved contain a large number of component surfaces.

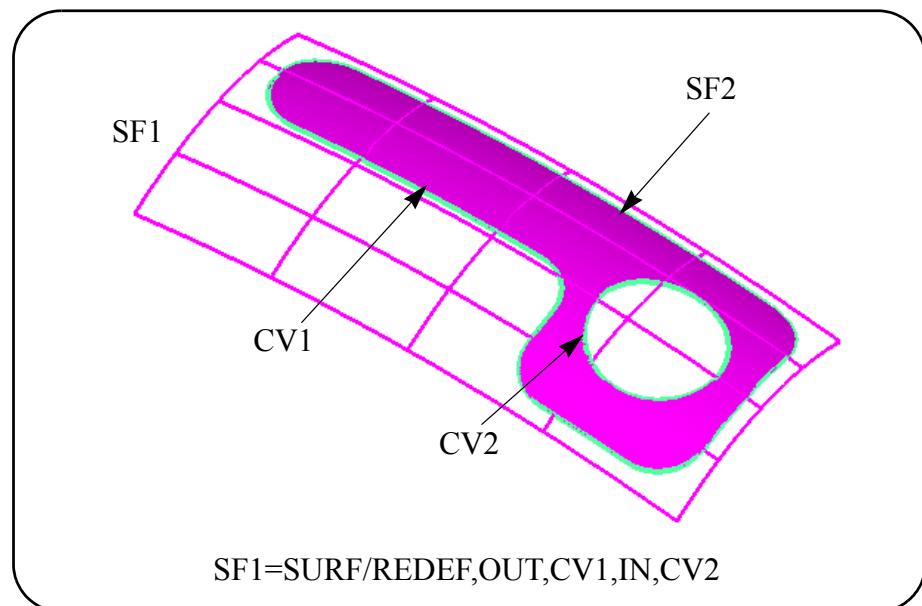
3.8.9 A Trimmed Surface Created By Specifying The Outer Boundary With Optional Inner Boundary(ies)

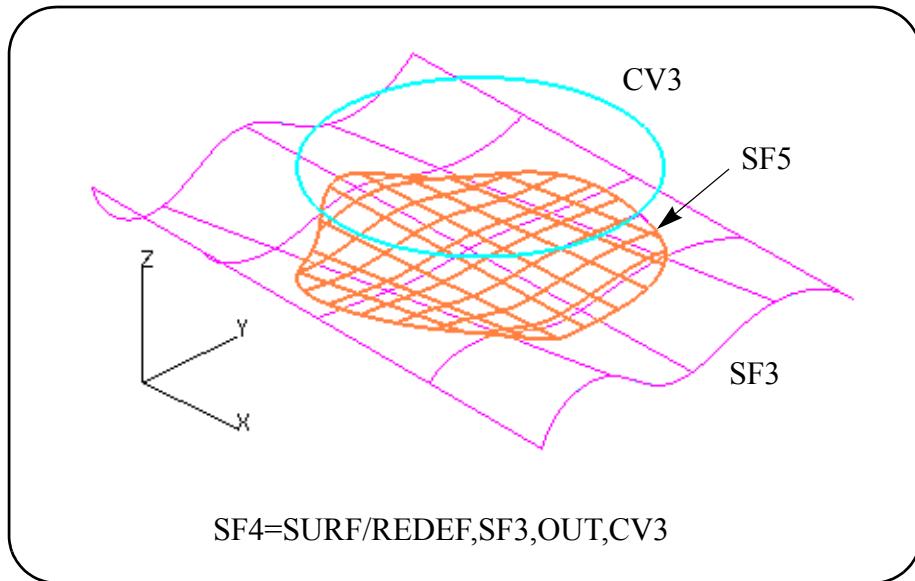
Command Syntax:

```
SURF/REDEF, [surface,]OUT,curve  
    plane
```

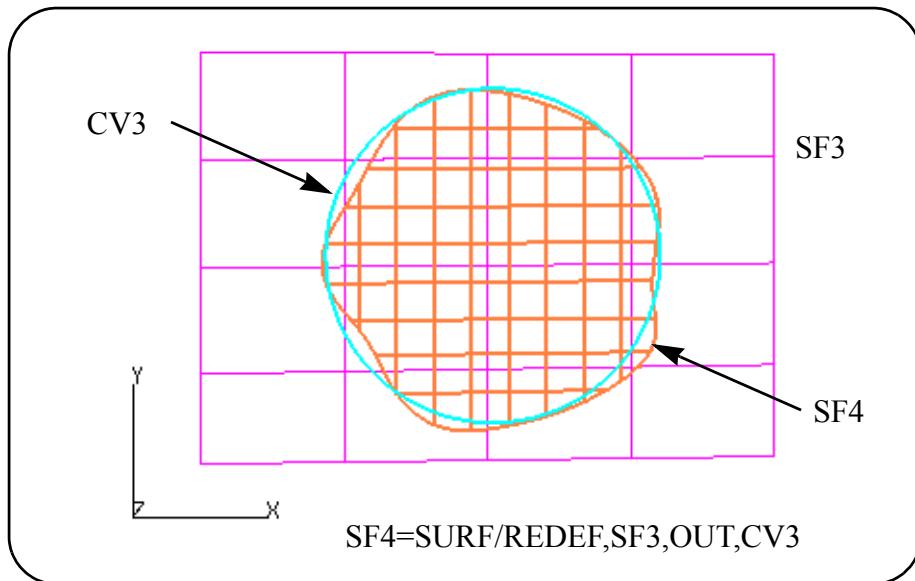
```
[,IN,curve[ [...] ,curve] ]
```

Icon Menu Sequence:

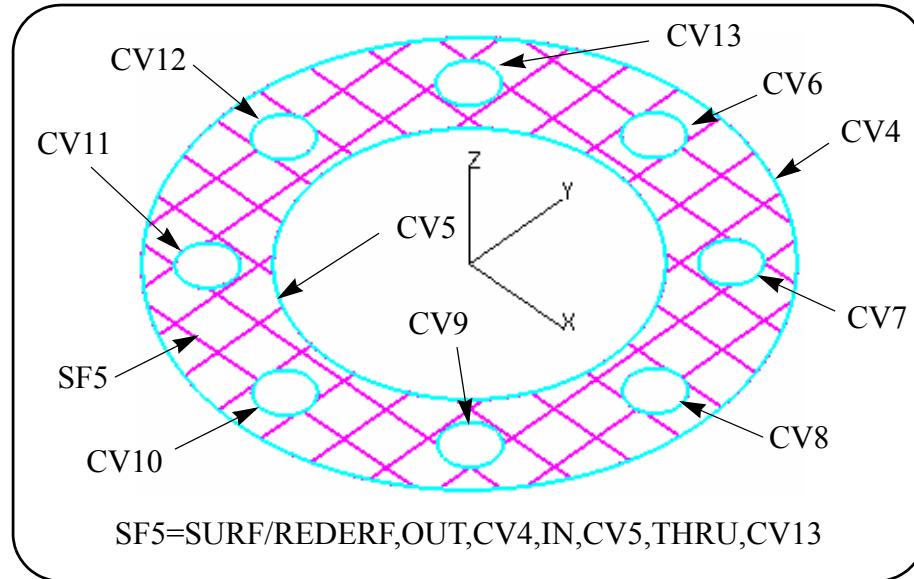




Isometric View



View looking downing the boundary curve plane



Calculation Method:

- This surface definition will create a trimmed surface out of the optional base surface/plane specified after the specifier: REDEF. If it is not specified, the outer and inner boundaries must be on a single planar surface or a plane and this single planar surface or the plane on which the boundaries lie will be used as the base to create the trimmed surface. The specified surface cannot be a **NET** surface.
- If the optional specified base surface is a “Trimmed Surface”, the underlying surface of this specified surface will be used as the base surface.
- The specifier **OUT** and the parameter after it specify the outer boundary of the trimmed surface. Only one outside boundary can be specified.
- The specifier **IN** and the parameters after it specify the inner boundary of the trimmed surface. More than one inner boundary can be specified.
- Input boundaries can be curves or circles.
- If the specified curve entities are not on the specified base surface/plane, they will be projected normal onto the specified surface/plane. The projected curves will be used as the boundaries of the trimmed surface.

Error conditions:

- The inner/outer boundaries or the projected inner/outer boundaries must be within the boundary of the specified base surface or its underlying surface if it is a “Trimmed Surface”, otherwise undesirable results might be obtained. This is not applicable if the specified entity is a plane.
- A NET surface is specified as the optional base surface.
- If the specified curve (or its projection if not on the specified surface) crosses over itself or overlaps a portion of itself.

Note: This command creates a new trimmed surface out of the specified surface. Use the [REDEF/surface,\[OUT\]...](#) command or the following icon sequence if you want to trim the specified surface only without creating a new surface:

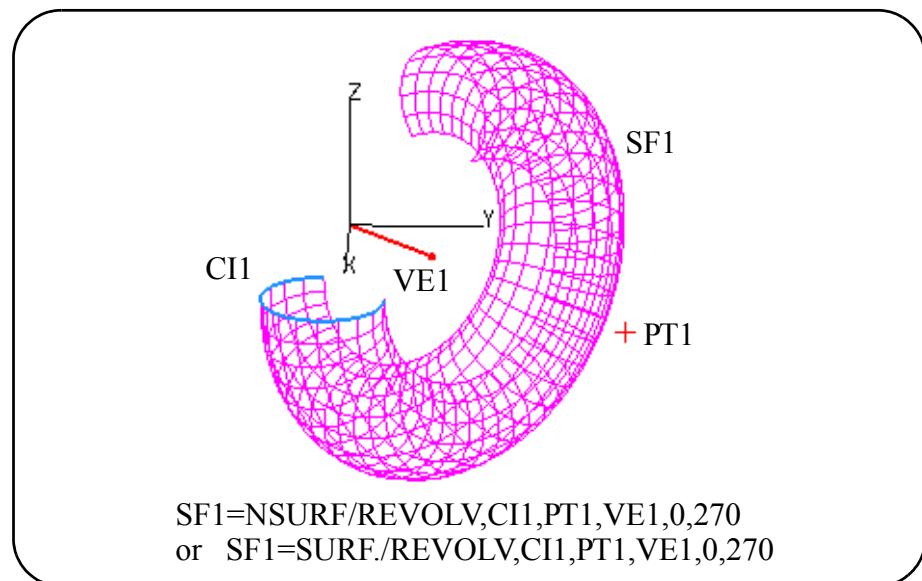


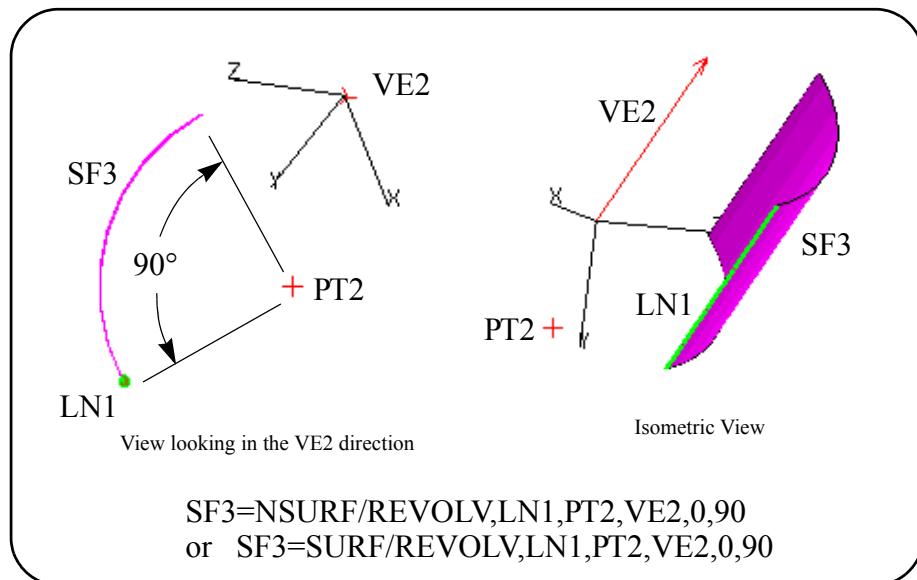
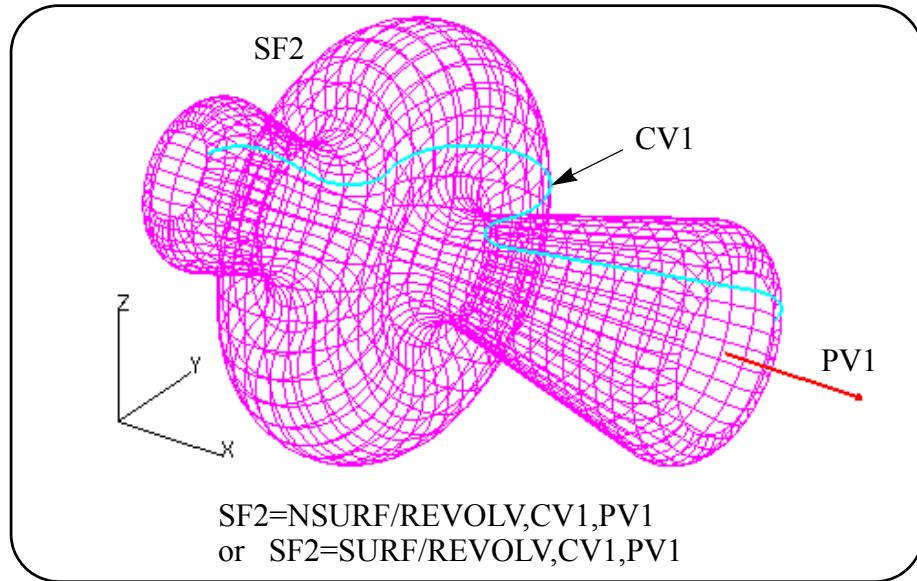
3.8.10 A Surface Of Revolution Constructed From A Single Wireframe Entity With Optional Starting And Ending Angle

Command Syntax:

```
NSURF/REVOLV, line , point-vector $  
SURF           circle point, vector  
                      curve  
  
[ ,start-angle,end-angle]
```

Icon Menu Sequence:





This surface is created by rotating the wire entity about the axis formed by the specified point-vector, or the specified point and along the specified vector direction as governed by the right hand rule. If the start and end angles are not specified, a full 360 degree surface is created. Start and/or end angles may be specified in decimal degrees or degrees, minutes and seconds (deg' min[^] sec; e.g. 10 degrees, 2 minutes and 30 seconds will be entered as 10'2[^]30).

Both the specifiers NSURF and SURF will create a surface with the primitive type “Surface of Revolution”.

3.8.11 A Surface Constructed From Four Edge Entities

Command Syntax:

```
NSURF/EDGE, interp, ent-U0, ent-V1, ent-U1, ent-V0
```

Where:

interp: Specifies the interpolation method used.

1 = Linear Interpolation

2 = Cubic Interpolation

Cubic interpolation will generally create a fuller, more natural looking surface

ent-U0: Specifies the edge entity at U=0. This can be a line, circle or curve.

ent-V1: Specifies the edge entity at V=1. This can be a line, circle or curve.

ent-U1: Specifies the edge entity at U=1. This can be a line, circle or curve.

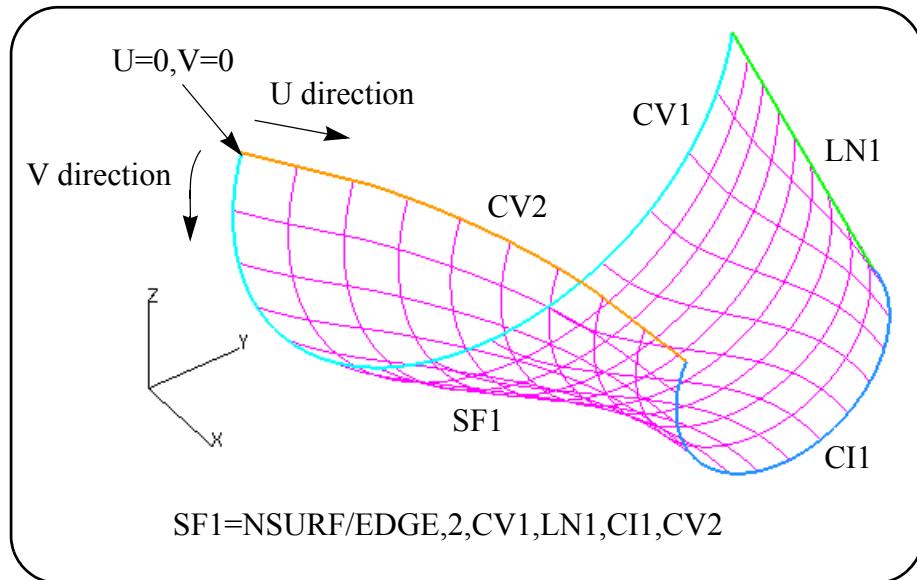
ent-V0: Specifies the edge entity at V=0. This can be a line, circle or curve.

Icon Menu Sequence:



This command creates a Non-Uniform Rational B-Spline surface out of the four specified edge entities. The edge entities can be any curve, line or circle.

The first and last entities' U-V origin must start in the same corner and the order in which the edge entities are specified is important. Otherwise, incorrect surface will be created.



VECTORS

The following list gives an abbreviated notation of all the valid VECTOR definition formats. See the individual sections for a complete explanation of each format.

1. `VECTOR/x, y, [z]`
2. `VECTOR/point , point
pntvec pntvec`
3. `VECTOR/pntvec`
4. `VECTOR/FWD`
5. `VECTOR/TLAXIS`
6. `VECTOR/UNIT,vector
pntvec`
7. `VECTOR/vector,TIMES,scalar
pntvec`
8. `VECTOR/vector,CROSS,vector
pntvec pntvec`
9. `VECTOR/vector,PLUS ,vector
MINUS`
10. `VECTOR/PERPTO,plane,POSX
NEGX
POSY
NEGY
POSZ
NEGZ`
11. `VECTOR/PARREL,INTOF,plane,plane,POSX
NEGX
POSY
NEGY
POSZ
NEGZ`
12. `VECTOR/point ,surface【u,v】]
pntvec`

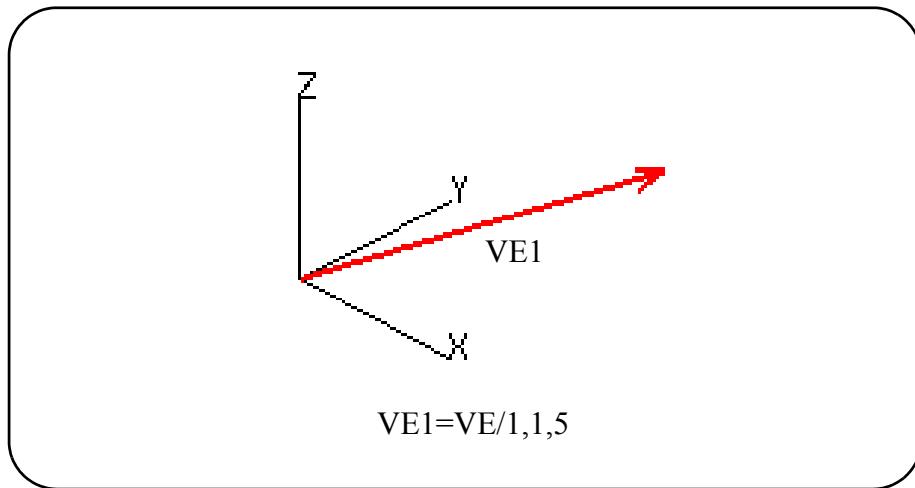
13. `VECTOR/TANTO,curve[[u]],point
pntvec`
or `VECTOR/TANTO,point ,curve[[u]]
pntvec`
14. `VECTOR/ATANGL,angle,line,POSX
XLARGE
POSY
YLARGE
NEGX
XSMALL
NEGY
YSMALL`

Example Vector Expressions:

```
VE/1,1,1  
VE/A,B  
VECTOR/FWD  
VE/TLAXIS  
VECTOR/PT1,PT2  
VECTOR/PV1,PV2  
VECTOR/PERPTO,PL1,POSZ  
VECTOR/UNIT,VE1  
VE/UNIT,PV1  
VECTOR/VE1,CROSS,VE2  
VE/PV1,CROSS,PV3  
VECTOR/PA,INTOF,PL1,PL2,NEGX  
VE/VE2,PLUS,VE3  
VE/PV1,MINUS,PV2  
VE/PT1,SF1  
VE/PV1,SF1  
VECTOR/VE1,TIMES,2  
VE/PV1,TIMES,3  
VE/TT,CV3,PT33  
VE/TT,CV1,PV2  
VE/TT,CV1,PV2
```

3.9 VECTOR Expressions

A vector is a geometric entity possessing two attributes - direction and magnitude.



Graphically, **NCL** displays a vector as a line starting at the origin and pointing in the direction indicated by the X, Y, Z components of the vector. The length of the line represents the magnitude of the vector. The magnitude, or length of a vector, is calculated as the square root of X squared plus Y squared plus Z squared.

$$\text{SQRT}((X^{**2})+(Y^{**2})+(Z^{**2}))$$

A unit vector is a vector of one unit magnitude calculated as follows:

$$\text{SQRT}((X^{**2})+(Y^{**2})+(Z^{**2}))=1$$

The X, Y, Z components of a unit vector are called direction cosines because they are equal to the cosines of the angles which the vector makes with the X, Y and Z coordinate axes.

A vector is not automatically unitized by the **NCL** system. If a unit vector is required, the "VECTOR/ UNIT" definition must be used.

Canonical Form:

$$\text{VECTOR/ I , J , K}$$

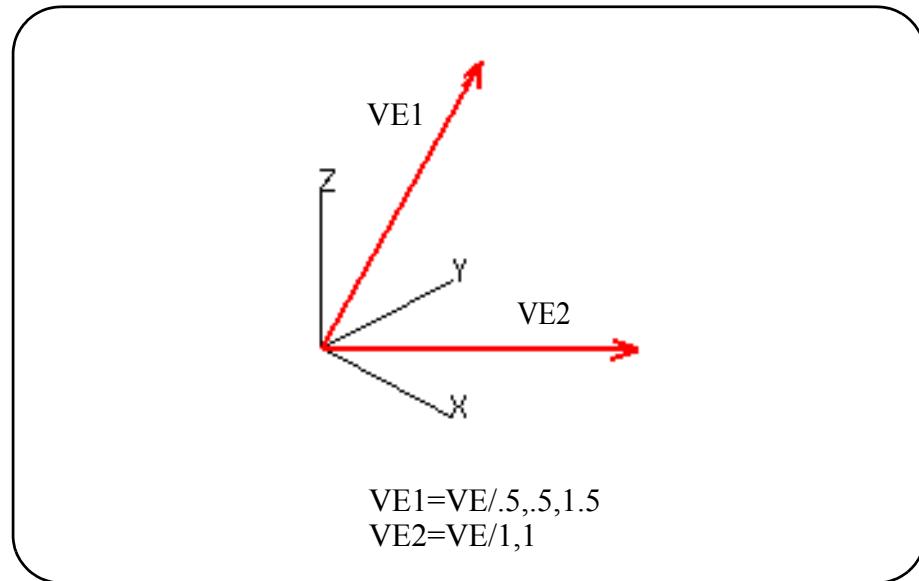
RED (pen 2) is the **NCL** System DEFAULT for CAM VECTORS.

3.9.1 A Vector By Its Components Along The X, Y And Z Axes

Command Syntax:

```
VECTOR/x-component,y-component[,z-component]
```

Icon Menu Sequence:



Calculation Method:

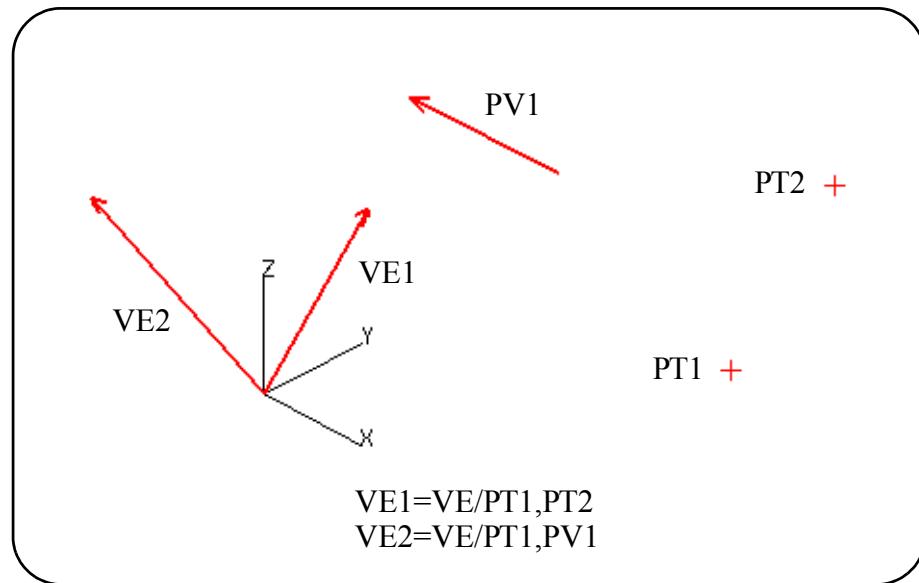
- Z-component will be defaulted to a value of "0" if it is not specified.
- The magnitude of the vector is calculated from the origin of the "current" REFERENCE SYSTEM to the "implied point" specified by the x, y and z components.
- The direction of the vector will be from the origin of the "current" REFERENCE SYSTEM to the "implied point" specified by the x, y and z components.

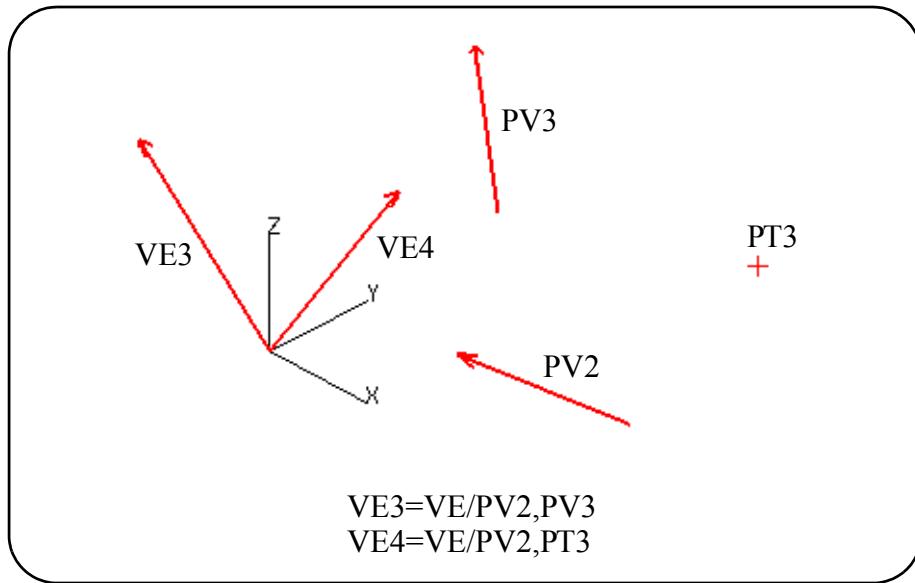
3.9.2 A Vector As The Direction From One Point/Point-Vector To Another Point/Point-Vector

Command Syntax:

```
VECTOR/point      , point  
          point-vector point-vector
```

Icon Menu Sequence:





Calculation Method:

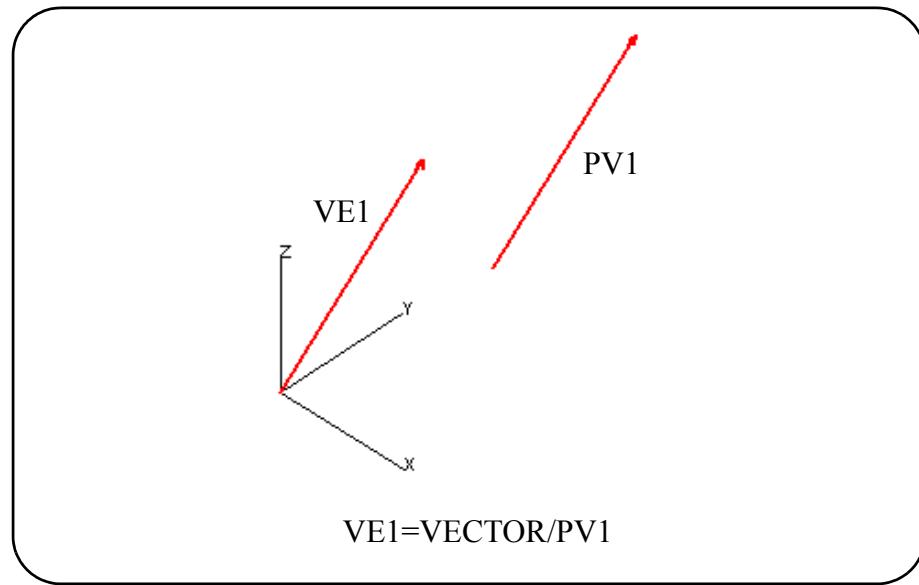
- The magnitude of the vector is calculated from the first referenced point (point-vector origin) to the second referenced point (point-vector origin).
- The direction of the vector will be from the first referenced point (point-vector origin) to the second referenced point (point-vector origin).

3.9.3 A Vector From A Point-Vector

Command Syntax:

VECTOR/point-vector

Icon Menu Sequence:



Calculation Method:

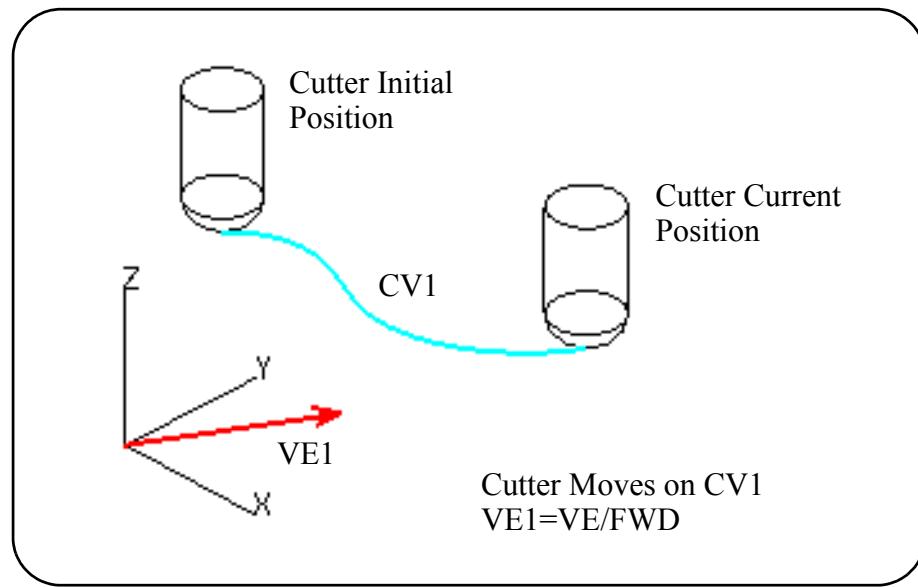
- The magnitude and the direction of the vector is equal to the point-vector.

3.9.4 A Vector In The Direction Of The Current Forward Sense Of The Tool

Command Syntax:

VECTOR / FWD

Icon Menu Sequence:



Calculation Method:

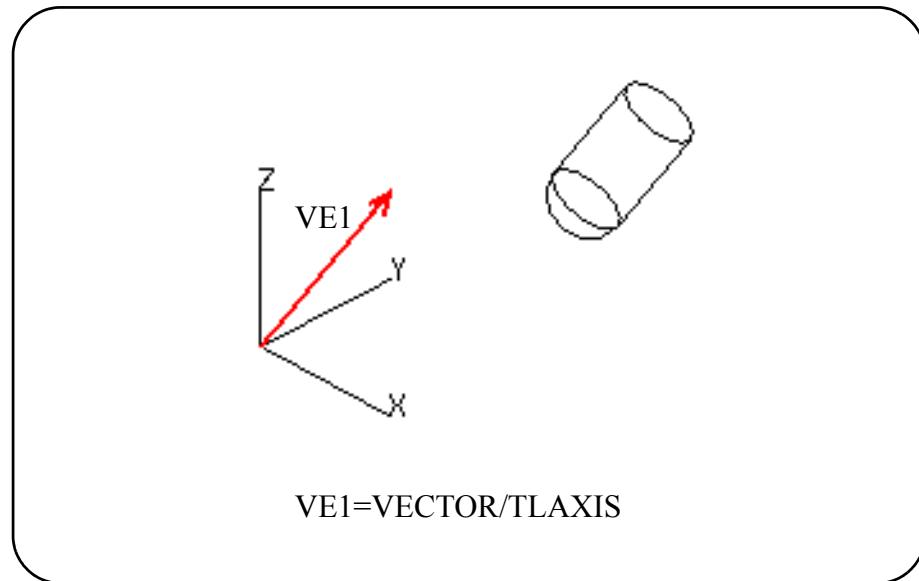
- The magnitude of the vector is one unit long.
- The direction of the vector is aligned with the current forward sense of the tool position.

3.9.5 A Vector In The Direction Up From The Current Tool Axis

Command Syntax:

VECTOR/TLAXIS

Icon Menu Sequence:



Calculation Method:

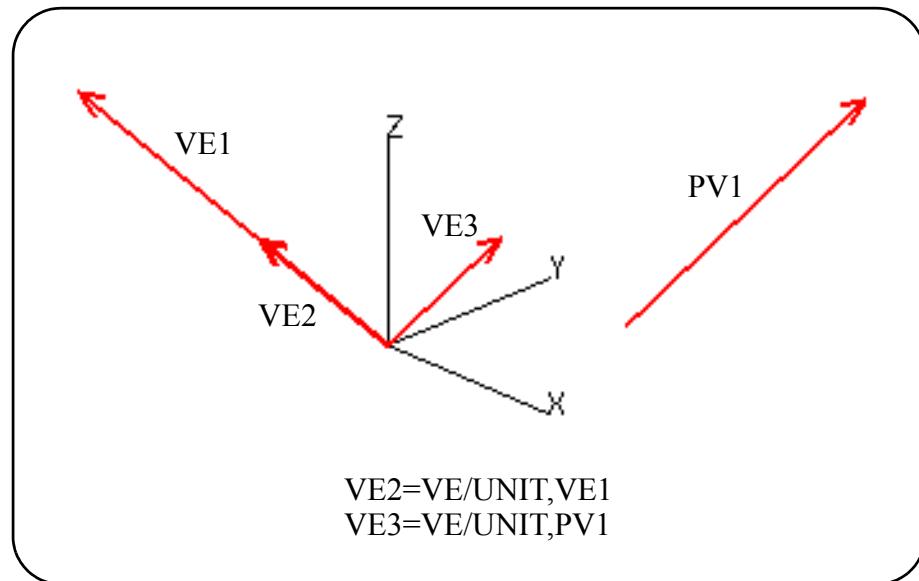
- The magnitude of the vector is one unit long,
- The direction of the vector points up the tool axis.
- If no **TLAXIS** statement has been specified prior to the use of this statement, **NCL** will use the System DEFAULT TLAXIS which is 0, 0, 1 (positive Z).

3.9.6 A Vector Which Is The Result Of Unitizing A Vector/Point-Vector

Command Syntax:

```
VECTOR/UNIT, vector
          point-vector
```

Icon Menu Sequence:



Calculation Method:

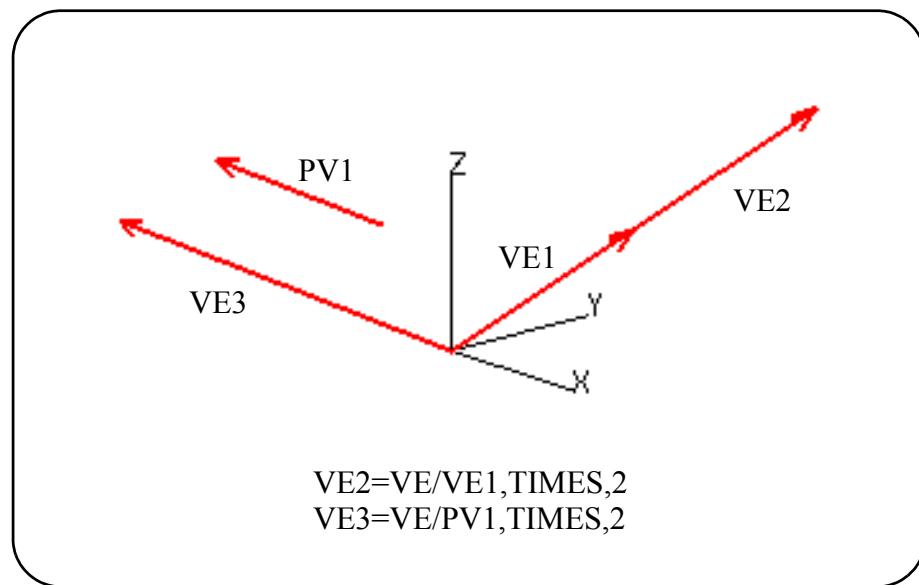
- The magnitude of the vector is one unit long.
- The direction of the vector is the same as the referenced vector/point-vector.

3.9.7 A Vector From A Vector/Point-Vector Multiplied By A Scalar

Command Syntax:

```
VECTOR/vector      , TIMES, magnitude-scalar
           point-vector
```

Icon Menu Sequence:



Calculation Method:

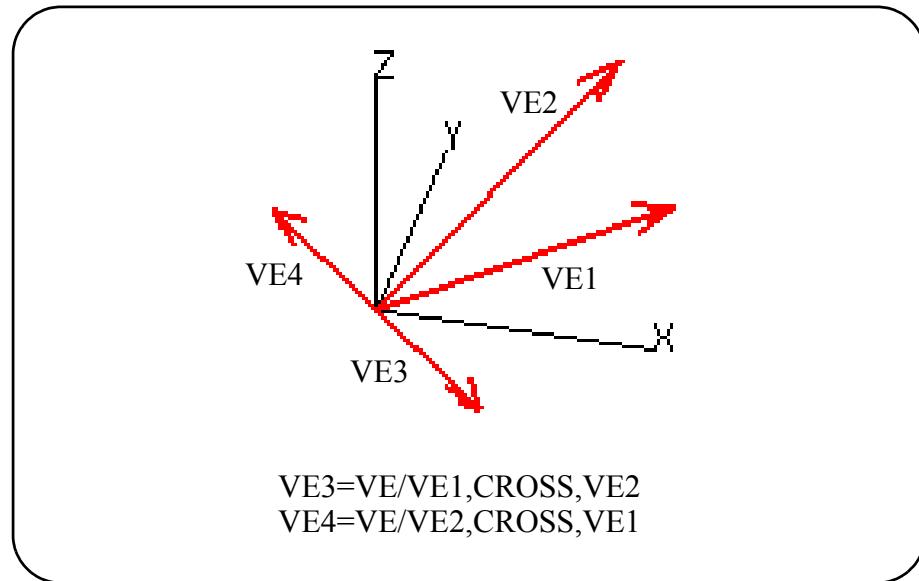
- The magnitude of the specified vector/point-vector will be multiplied by the magnitude-scalar to produce the defined vector's magnitude.
- The direction of the new vector will be the same as the specified vector/point-vector.

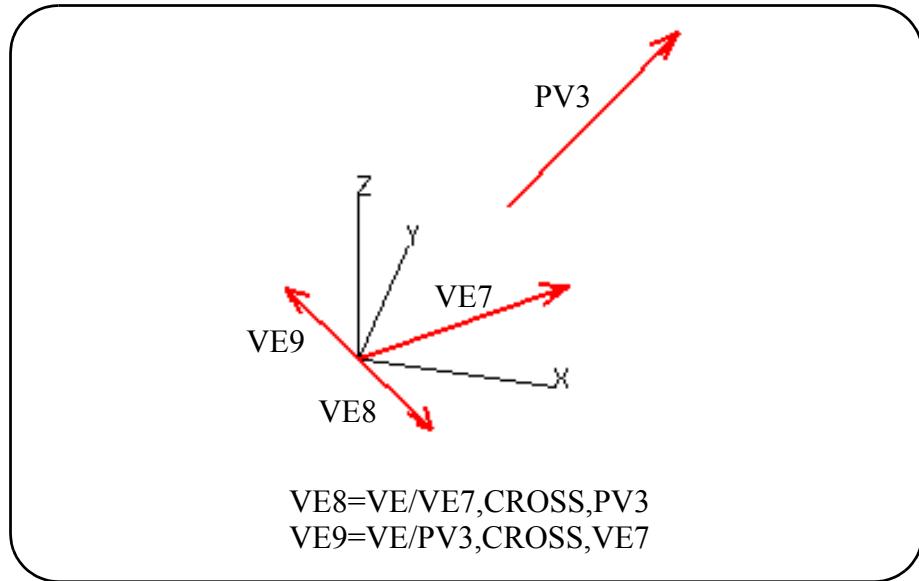
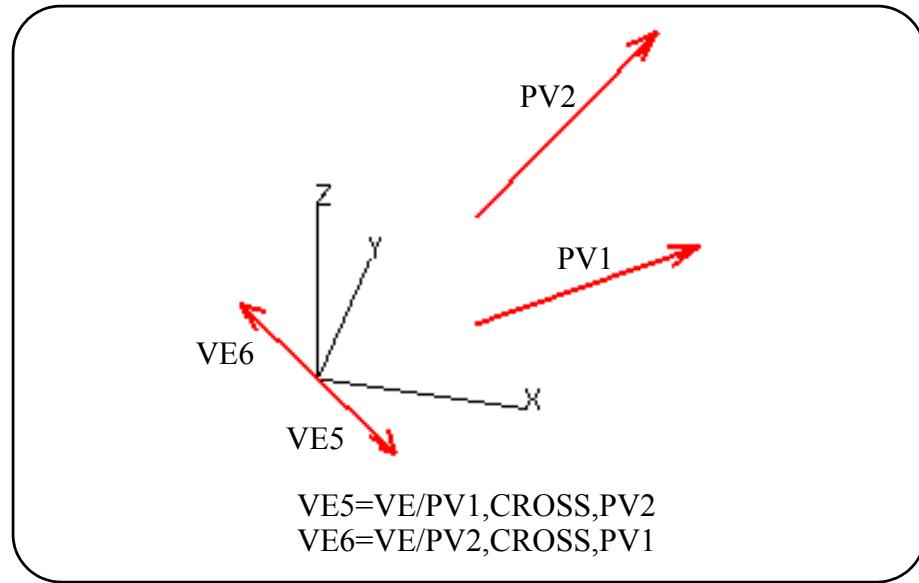
3.9.8 A Vector As The Cross Product (Normal) Of Any Combination Of Two Vectors/Point-Vectors

Command Syntax:

```
VECTOR/vector      ,CROSS,vector  
point-vector      point-vector
```

Icon Menu Sequence:





Calculation Method:

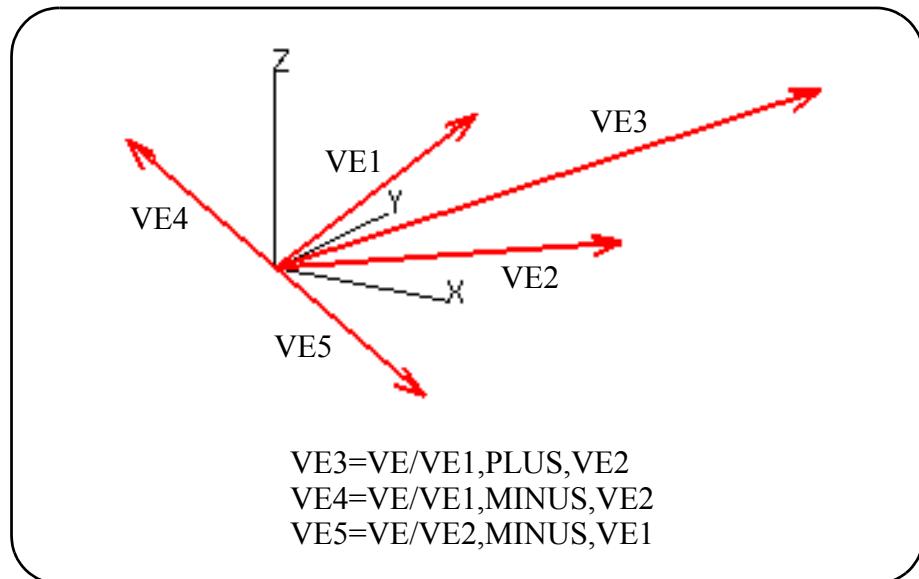
- The magnitude of the vector is equal to the product of the magnitudes of the referenced vectors/point-vectors times the sine of the angle between them.
- The direction of the vector is perpendicular to the plane of the two referenced vectors/point-vectors, measured from the first vector/point-vector to the second point/vector, using the "right hand" rule.

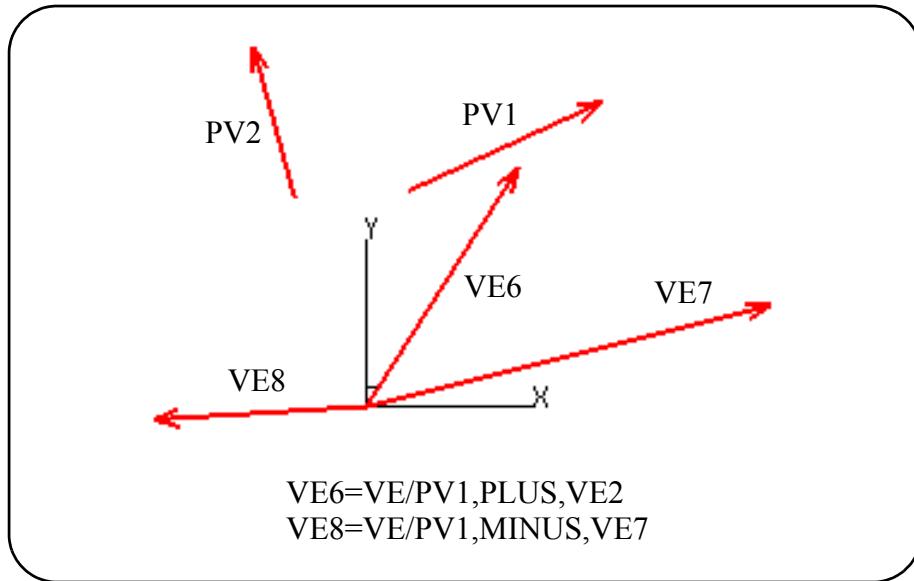
3.9.9 A Vector As The Sum or Difference Of Any Combinations Of Two Vectors Or Point-Vectors

Command Syntax:

```
VECTOR/vector      ,MINUS, vector
           point-vector PLUS  point-vector
```

Icon Menu Sequence:





Calculation Method:

- The magnitude of the new vector is calculated as the sum or difference of the two referenced vectors/point-vectors as follows:

$$\text{Sum} = \text{SQRT}(a^{**2} + b^{**2} + 2*a*b*\cos(\text{ang}))$$

$$\text{Difference} = \text{SQRT}(a^{**2} + b^{**2} - 2*a*b*\cos(\text{ang}))$$

Where "a" is the length of the first vector/point-vector and "b" is the length of the second vector/point-vector and "ang" is the true angle between them.

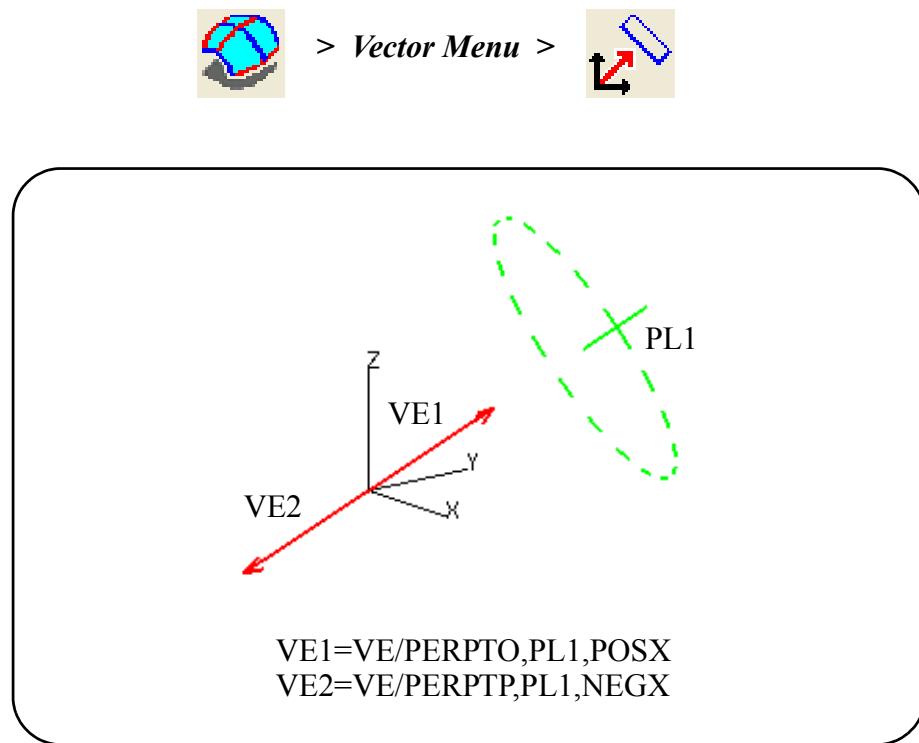
- The direction of the vector will be the sum or difference of the x, y and z-coordinates depending on the modifier used. Each coordinate is calculated separately; i.e. the directional length of the second vector/point-vector along the x-axis is added to or subtracted from the directional length of the first vector/point-vector along the x-axis to obtain the resultant vector's x-coordinate. Where directional length means if the vector/point vector portion points into the negative direction, the length have a negative value, if it points in the positive direction, it has a positive value.

3.9.10 A Vector Perpendicular To A Plane

Command Syntax:

```
POSX
NEGX
POSY
VECTOR/PERPTO,plane,NEGY
POSZ
NEGZ
```

Icon Menu Sequence:



NOTE: The orientation modifiers **POSX**, **NEGX**, **POSY**, **NEGY**, **POSZ** and **NEGZ** indicate the general direction of the vector to be defined.

Calculation Method:

- The magnitude of the vector is one unit long.
- The direction of the vector is determined by the specified orientation modifier.

3.9.11 A Vector As The Intersection Of Two Planes

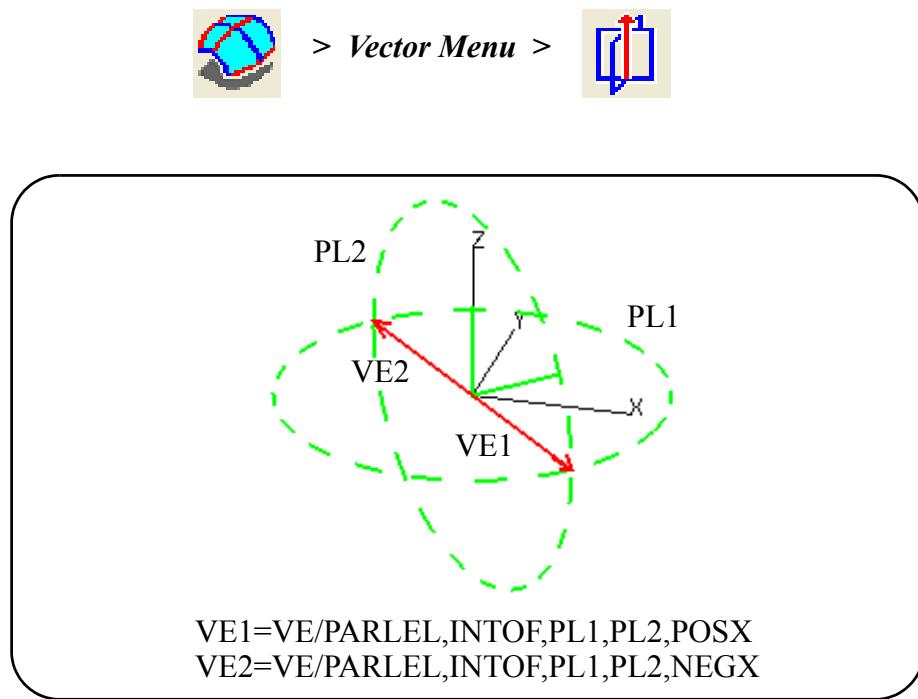
Command Syntax:

```

POSX
NEGX
POSY
VECTOR/PARREL,INTOF,plane,plane,NEGY
POSZ
NEGZ

```

Icon Menu Sequence:



Calculation Method:

- The magnitude of the vector is one unit long.
- The orientation modifiers **POSX**, **NEGX**, **POSY**, **NEGY**, **POSZ** and **NEGZ** indicate the general direction of the vector to be defined.

Error Condition:

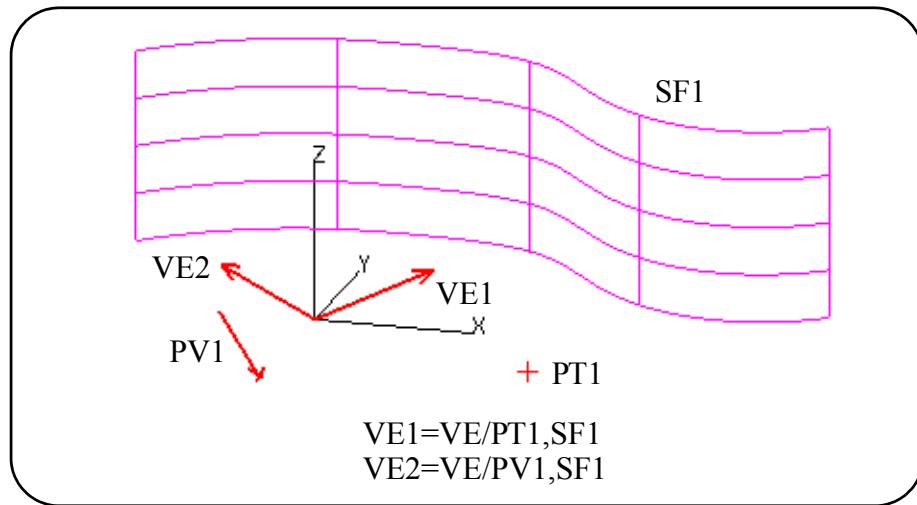
- The planes must intersect.

3.9.12 A Vector As The Directed Distance From A Point/Point-Vector Normal To A Surface

Command Syntax:

```
VECTOR/point      ,surface[[u,v]]  
    point-vector
```

Icon Menu Sequence:



The inner square brackets must be specified as part of the syntax if the optional [u,v] parameters are used.

The optional [u,v] values may be specified to tell **NCL** to “look” at the surface in the area specified by the UV values. This is sometimes necessary on round, U-shaped, or S-shaped surfaces as there may be multiple projection possibilities.

Calculation Method:

- The magnitude of the vector is one unit.
- The direction of the vector is from the specified point or the point-vector origin along a line normal to the surface.

3.9.13 A Vector Tangent To A Curve With A Near Point/Point-Vector Specified

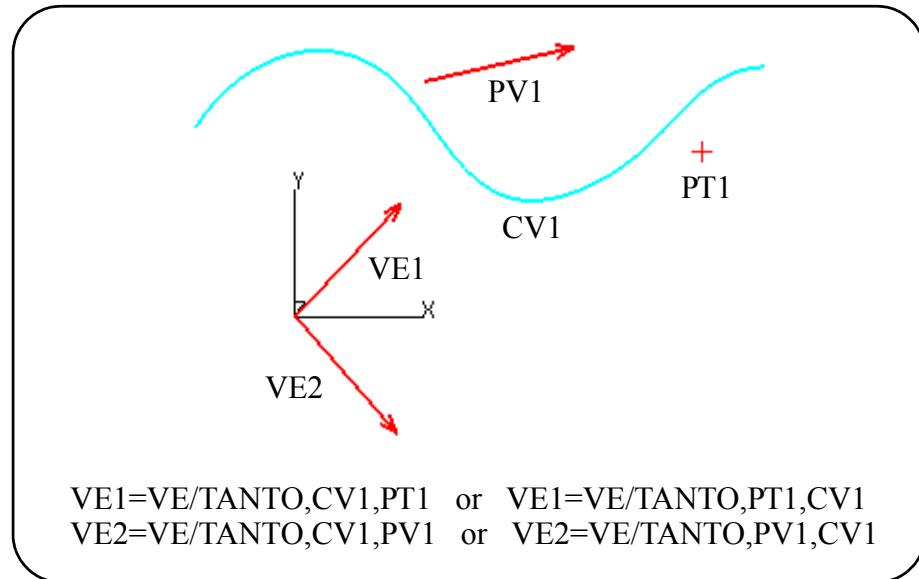
Command Syntax:

```
VECTOR/TANTO,curve[[u]],near-point  
near-point curve[[u]]
```

or

```
VECTOR/TANTO,curve[[u]] ,near-point-vector  
near-point-vector curve[[u]]
```

Icon Menu Sequence:



The inner square brackets must be specified as part of the syntax if the optional [u] parameters are used.

The optional [u] values may be specified to tell **NCL** to “look” at the curve in the area specified by the U values. This is sometimes necessary on round, U-shaped, or S-shaped curves as there may be multiple projection possibilities.

Calculation Method:

- If the point (point-vector origin) is on the curve, then the direction will be a function of the direction of the curve and the vector will have the slope of the curve at that point. The tangent vector will be one unit in magnitude along the forward sense of the curve.
- If the point (point-vector origin) is not on the curve, it is projected normal on the curve. The vector will be constructed through this projected point and tangent to the curve as described above.

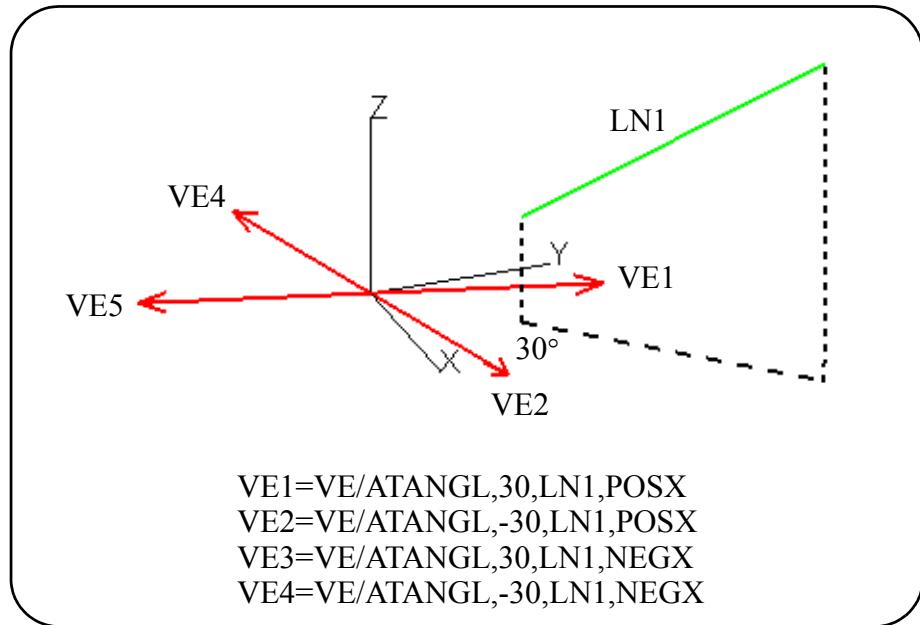
3.9.14 A Vector In The XY-Plane At An Angle With A Line

Command Syntax:

```
VECTOR/ATANGL,angle,line,POSX
          XLARGE
          POSY
          YLARGE
          NEGX
          XSMALL
          NEGY
          YSMALL
```

Icon Menu Sequence:

No corresponding Icon Menu



Calculation Method:

- The line is projected onto the XY-plane of the “current” REference SYstem.
- The plane of the vector will be the XY-plane of the ‘current’ REference SYstem.

- The magnitude of the vector is one unit.
- The modifiers POSX, XLARGE, POSY, YLARGE, NEGX, XSMALL, NEGY and YSMALL designate the desired direction of the desired vector in relation to the “current” REFERENCE SYSTEM. POSX and XLARGE have the same meaning, indicating that the desired vector has positive x-component.
- “angle” is the angle measured in degrees from the specified line toward the vector and is positive for counterclockwise and negative for clockwise.

MATRIX

The following list gives an abbreviated notation of all the valid MATRIX definition formats. See the individual sections for a complete explanation of each format.

1. **MATRIX**/Ix,Jx,Kx,Dx, \$
Iy,Jy,Ky,Dy, \$
Iz,Jz,Kz,Dz
2. **MATRIX**/TRANSL,Dx,Dy[,Dz]
3. **MATRIX**/XYROT,angle
YZROT
ZXROT
4. **MATRIX**/SCALE[,point],X-scale[,Y-scale[,Z-scale]]
pntvec
5. **MATRIX**/matrix,matrix
6. **MATRIX**/INVERS,matrix
7. **MATRIX**/MIRROR,plane
8. **MATRIX**/point,point-vector,point-vector
vector vector
9. **MATRIX**/point-vector,point-vector
vector
10. **MATRIX**/in-pt1,in-pt2,in-pt3,out-pt1,out-pt2,out-pt3

Example Matrix Expressions:

```
MATRIX/0,0,1,0,0,1,0,0,1,0,0,0  
MATRIX/TRANSL,3.5,2.75,75  
MATRIX/XYROT,15  
MATRIX/SCALE,.5  
MATRIX/MX1,MX2  
MATRIX/INVERS,MX1  
MATRIX/MIRROR,PL1  
MATRIX/PT1,VE1,VE2  
MATRIX/PT1,PV1,PV2  
MX/PV1,PV2  
MX/PV1,VE2
```

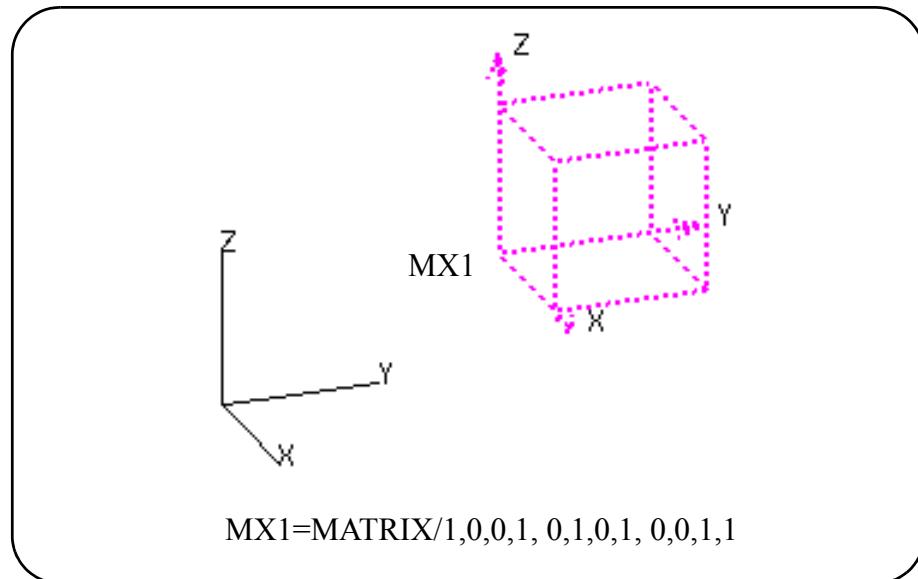
3.10 MATRIX Expressions

An **NCL** MATRIX is a rectangular array of numbers; it is not a piece of geometry. However, since it is used to perform transformation operations on geometry, we will treat it as such. An array of numbers is used to relate the *Secondary Coordinate System* to the *Primary Coordinate System*.

Canonical Form:

```
MATRIX/Ix,Jx,Kx,Dx,    $  
Iy,Jy,Ky,Dy,    $  
Iz,Jz,Kz,Dz
```

The displayed matrix appears as three vectors corresponding to the matrix axes and the wire frame of a parallelepiped (box). The default line style is dotted. The color of the matrix changes according to how it was used. By **default**, the color of matrix is **Magenta**. When it is used for **REFSYS**, the color is **Cyan**. When it is used for **MODSYS**, the color is **Red**. When it is used for **TRACUT**, the color is **Green**. For more information, see **DISPLAY/MATRIX** statement in Chapter 8 Graphic Display Control Statements.

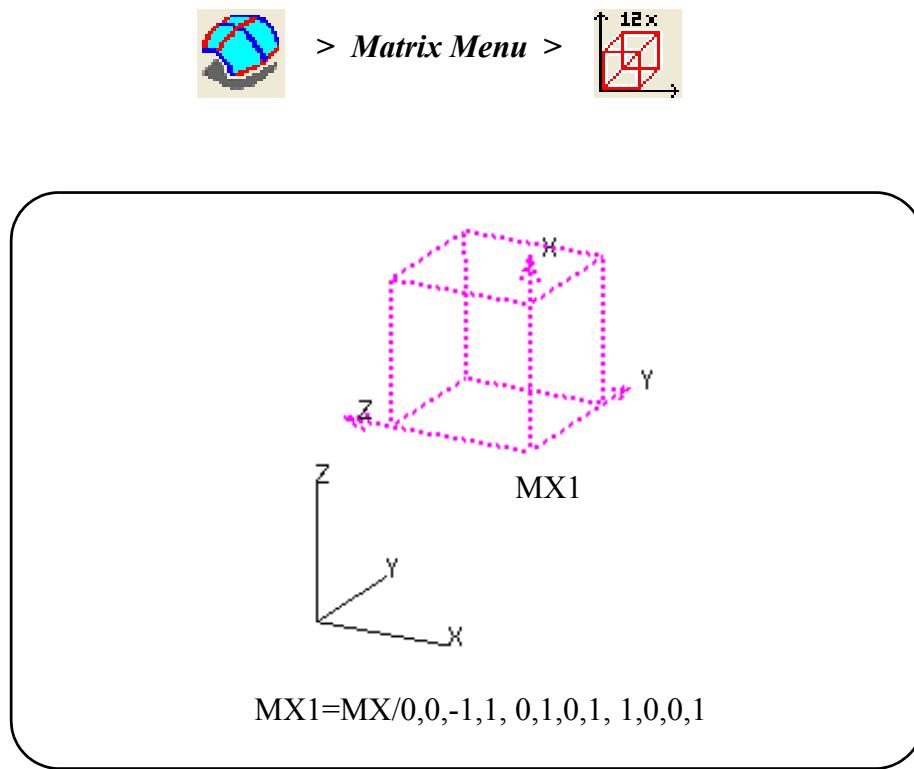


3.10.1 A Matrix In Terms Of Its Canonical Form

Command Syntax:

```
MATRIX/Ix,Jx,Kx,Dx, $  
Iy,Jy,Ky,Dy, $  
Iz,Jz,Kz,Dz
```

Icon Menu Sequence:



This matrix is defined in terms of 12 elements.

where:

Dx, Dy, Dz - the origin of the secondary coordinate system in terms of the primary coordinate system

Ix, Iy, Iz - the unit positive X-axis vector of the secondary coordinate system in terms of the primary coordinate system

Jx, Jy, Jz - the unit positive Y-axis vector of the secondary coordinate system in terms of the primary coordinate system

Kx, Ky, Kz - the unit positive Z-axis vector of the secondary coordinate system in terms of the primary coordinate system

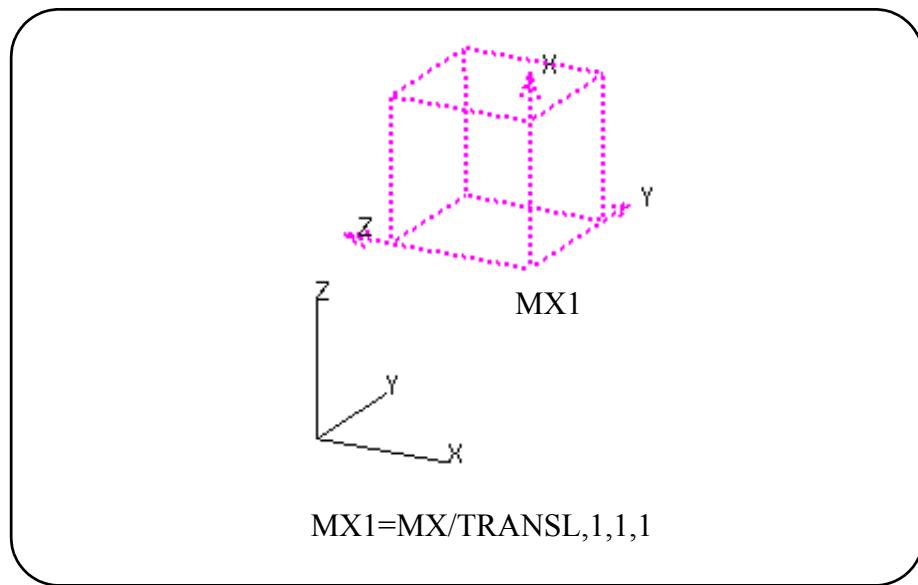
Note: **NCL** will not check if the secondary coordinate system is an orthogonal system or the unit vectors are unitized vectors in the primary coordinate system. Therefore if the 12 numbers are not specified correctly, an non-orthogonal coordinate system with/without scaling in any of the secondary axis will be created.

3.10.2 A Matrix As A Translation In X, Y And Z

Command Syntax:

```
MATRIX/TRANSL,delta-x,delta-y[,delta-z]
```

Icon Menu Sequence:



This matrix definition moves the Secondary Coordinate System a delta-x, delta-y and delta-z distance relative to the Primary Coordinate System. The z-coordinate may be omitted; it will then default to zero. In the example above, MX1 has moved the Secondary Coordinate System one unit in x, y and z from the Primary Coordinate System.

The translational matrix is corresponding to the following 12 parameters matrix.

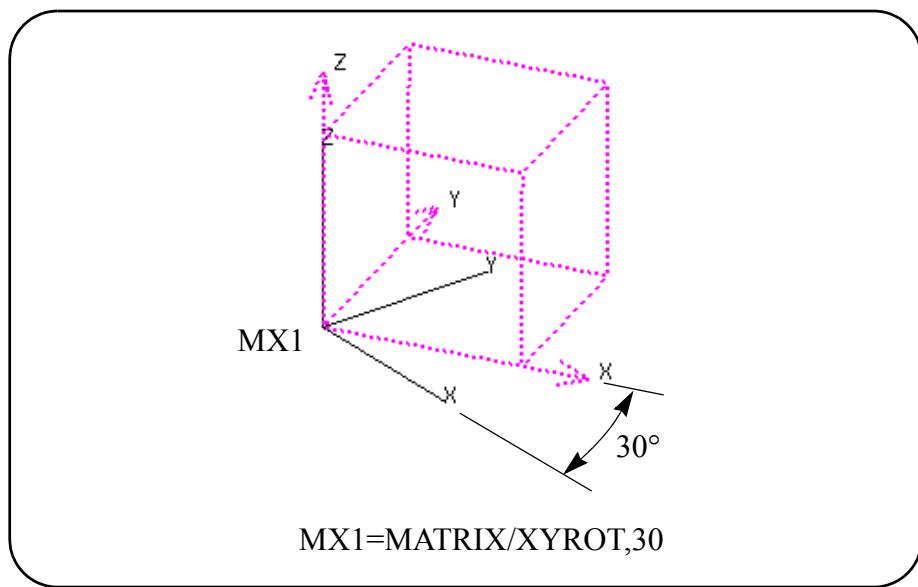
$$\begin{array}{cccc} 1, & 0, & 0, & \text{delta-x} \\ 0, & 1, & 0, & \text{delta-y} \\ 0, & 0, & 1, & \text{delta-z} \end{array}$$

3.10.3 A Matrix As A Rotation In A Specified Coordinate Plane

Command Syntax:

```
XYROT
MATRIX/YZROT, angle
ZXROT
```

Icon Menu Sequence:

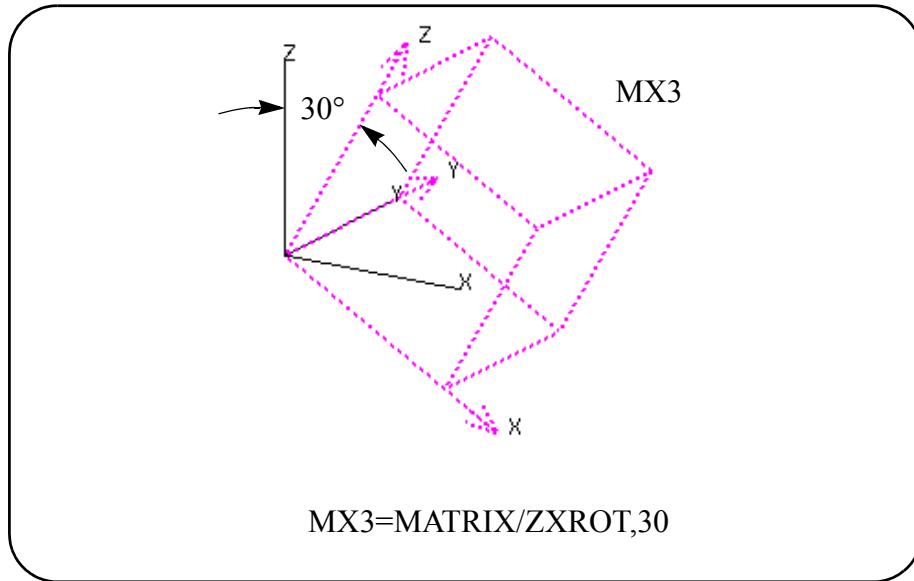
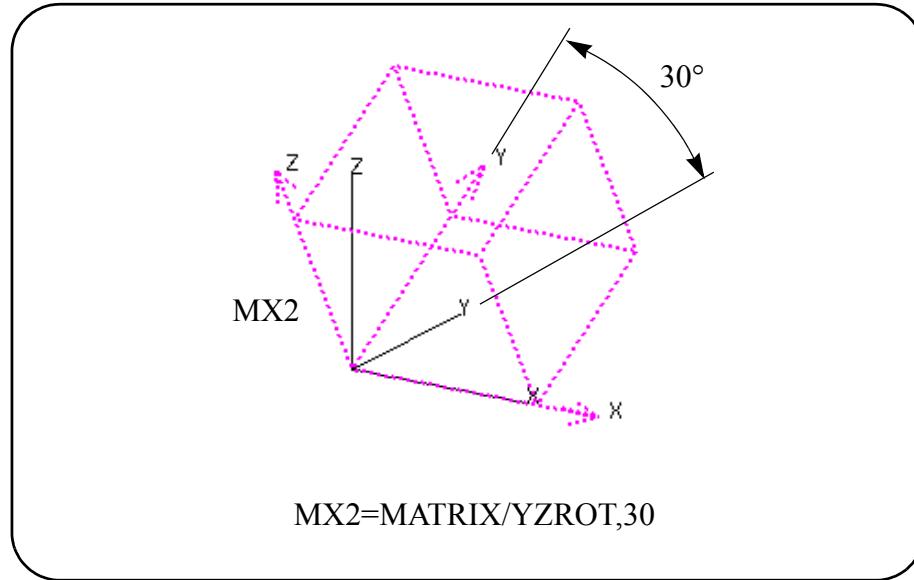


This matrix will rotate two of the Primary System's axes around the third.

XYROT rotates X and Y around the Z axis
YZROT rotates Y and Z around the X axis
ZXROT rotates Z and X around the Y axis

In above example, MX1 has rotated the Primary X and Y-axes 30 degrees using the Right Hand Rule, i.e. counterclockwise around the Z-axis. The angle of rotation, looking from the positive Z-axis toward the origin, is measured from the positive

X-axis to the positive Y-axis for a positive counterclockwise rotation. A negative measurement would result in a clockwise rotation from Y-axis toward the X-axis. Angles may be specified in decimal degrees or degrees, minutes and seconds (deg' min^ sec; e.g. 10 degrees, 2 minutes and 30 seconds will be entered as 10'2^30).



3.10.4 A Matrix As Scale Factors

Command Syntax:

```
MATRIX/SCALE[,point ],X-scale[,Y-scale[,Z-scale]]
          pntvec
```

Where:

point or pntvec: specifies the point location where the scaling will be performed from. If a pntvec is specified, the origin of the pntvec will be used. If not specified, it is defaulted to the origin of the Primary system.

X-scale, Y-scale, Z-scale : specify the individual scaling factors along each of the X, Y and Z-axis. If only one value is specified, all three axes have the same scaling factor. If Z-scale is not specified, the value specified for the Y-axis will be used for the Z-scale.

Icon Menu Sequence:



This Scaling Matrix definition is corresponding to the following 12 parameter matrix.

$$\begin{array}{cccc} \text{X-scale}, & 0, & 0, & -X1 \\ 0, & \text{Y-scale}, & 0, & -Y1 \\ 0, & 0, & \text{Z-scale}, & -Z1 \end{array}$$

or corresponding to the following combined matrix operation.

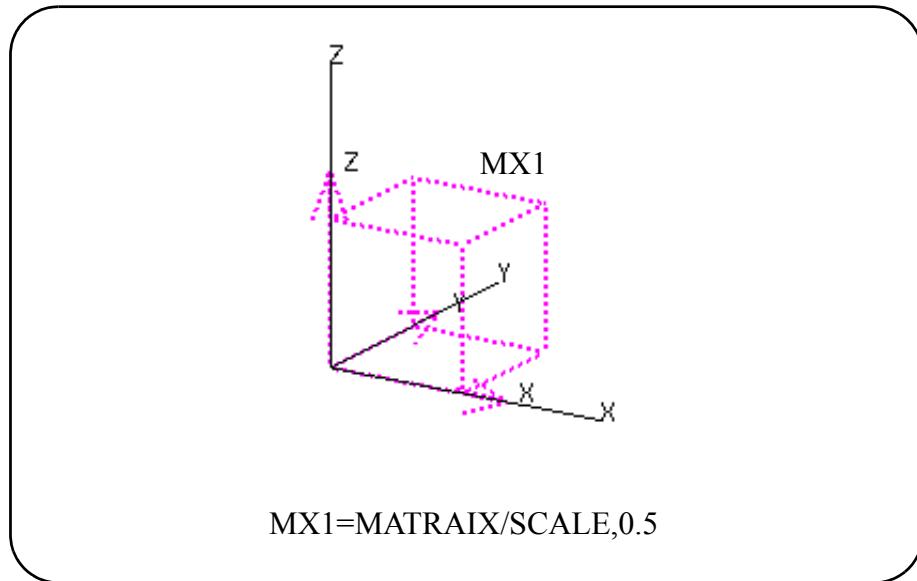
$\text{MX}/(\text{MX}/\text{INVERS},(\text{MX}/\text{TRANSL},X1,Y1,Z1)), (\text{MX}/\text{SCALE},\text{X-scale})$

where:

$$\begin{aligned} X1 &= x * (\text{X-scale} - 1) \\ Y1 &= y * (\text{Y-scale} - 1) \end{aligned}$$

$$Z1 = z * (\text{Z-scale} - 1)$$

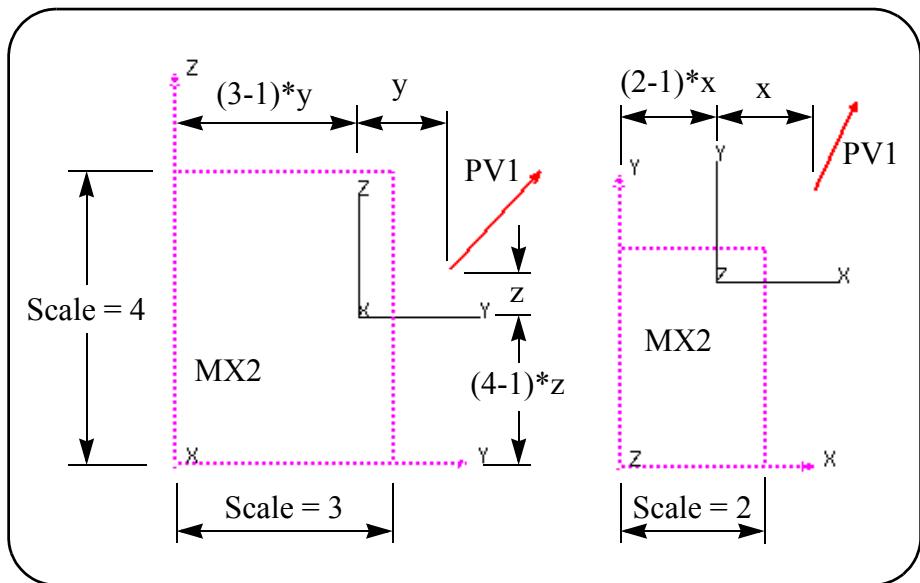
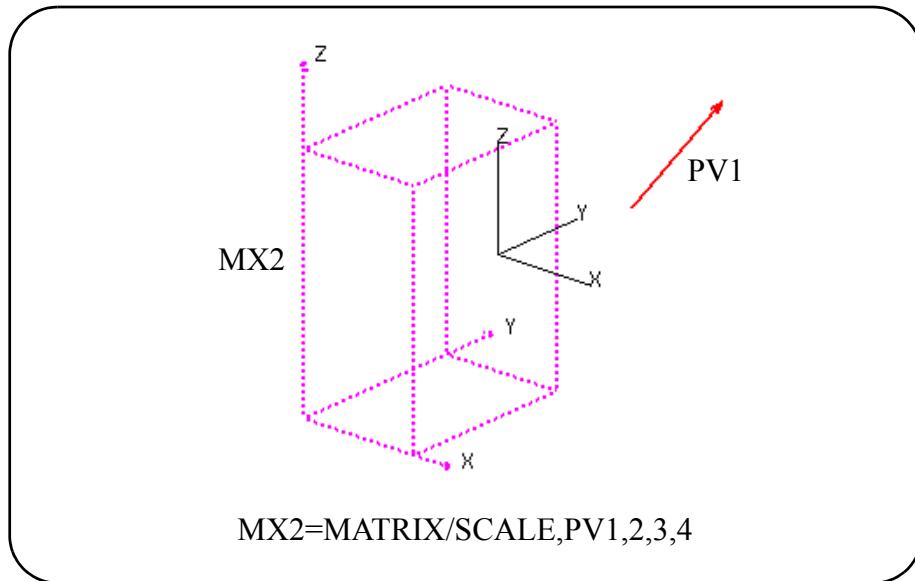
x, y, z is the X, Y, Z-coordinate of the optional specified point or the origin of the specified point-vector

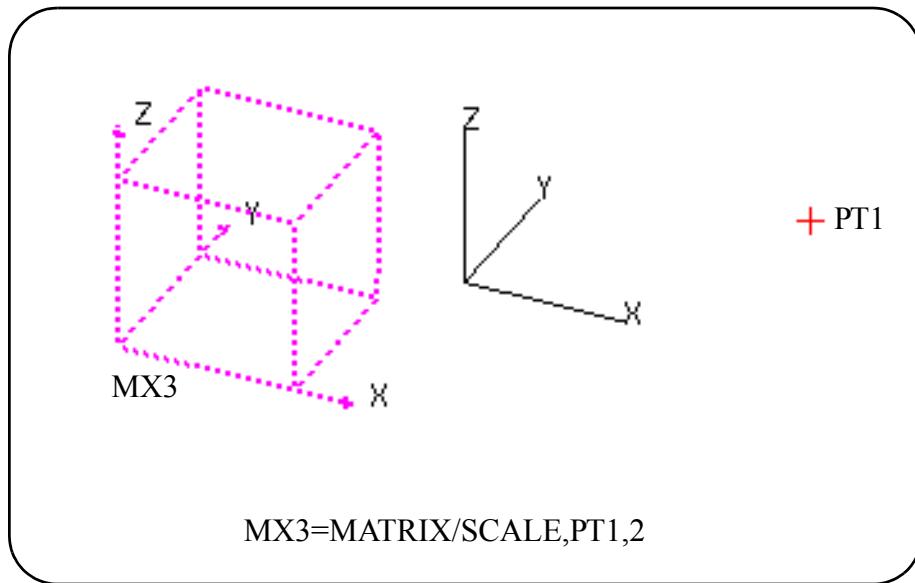


Above example defines a matrix with just one scale factor at the origin of the Primary System. MX1 generated the following matrix:

$$\begin{matrix} .5, & 0, & 0, & 0 \\ 0, & .5, & 0, & 0 \\ 0, & 0, & .5, & 0 \end{matrix}$$

The primary purpose of this matrix is for use with [TRACUT](#), [COPY](#) or [MODSYS](#) to scale cutter location points. MX1 would convert all point coordinates to half scale.



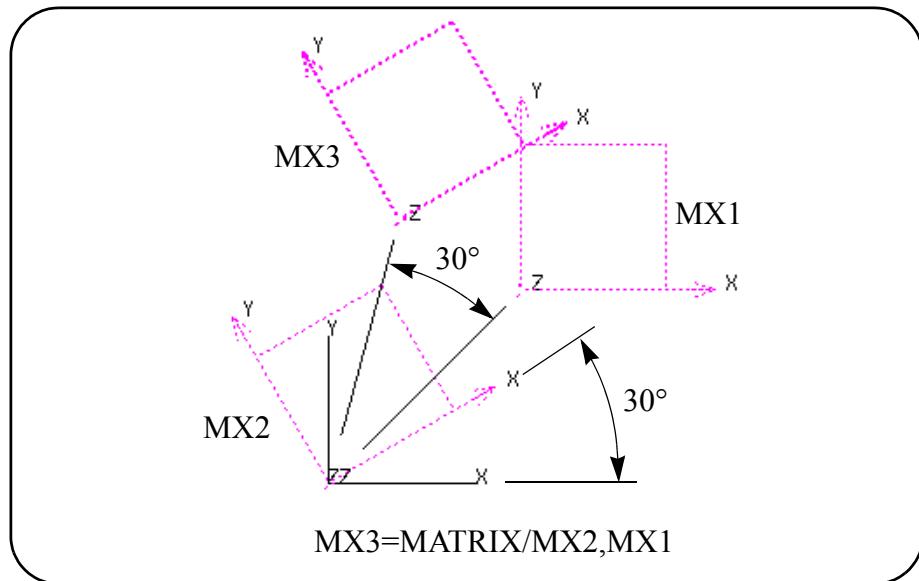


3.10.5 A Matrix As The Product Of Two Other Matrices

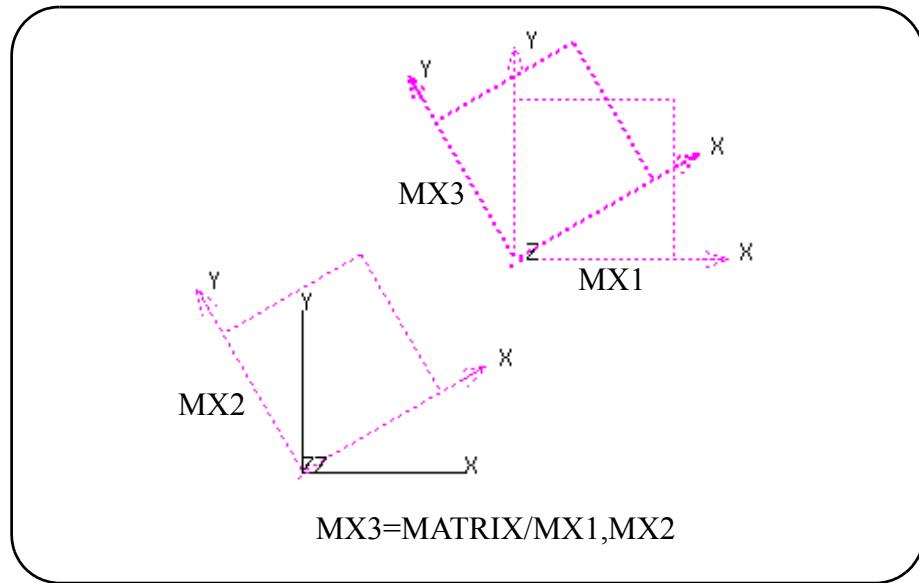
Command Syntax:

```
MATRIX/matrix,matrix
```

Icon Menu Sequence:



This matrix definition transforms the second specified matrix by the first specified matrix in the primary coordinate system. As shown in the example above, MX3 is obtained by rotating MX1 30 degrees (which is specified by MX2) in the primary coordinate system. While the example on next page, MX3 is obtained by translating MX2 with amount specified by MX1 in the primary coordinate system.

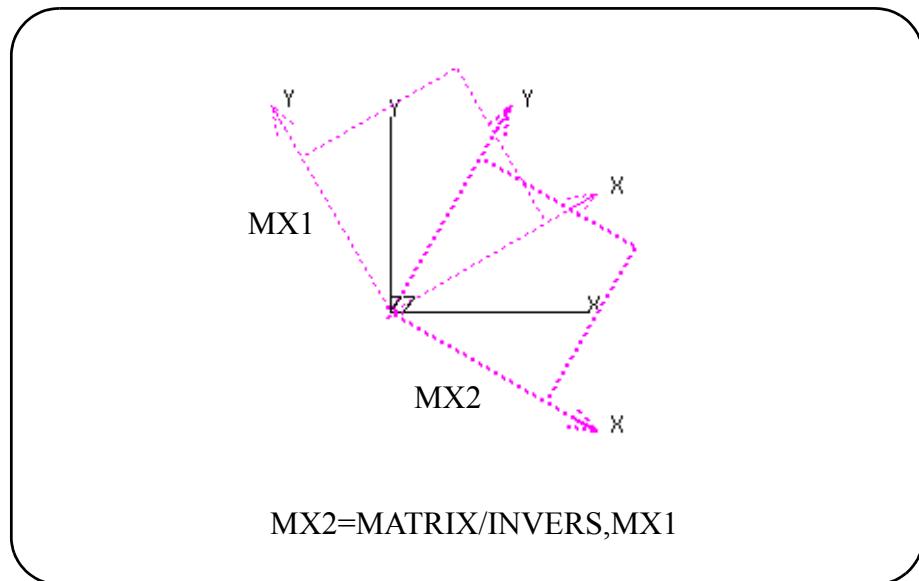


3.10.6 A Matrix As The Inverse Of A Matrix

Command Syntax:

```
MATRIX/INVERS,matrix
```

Icon Menu Sequence:



This matrix performs the inverse transformation of that performed by the given matrix.

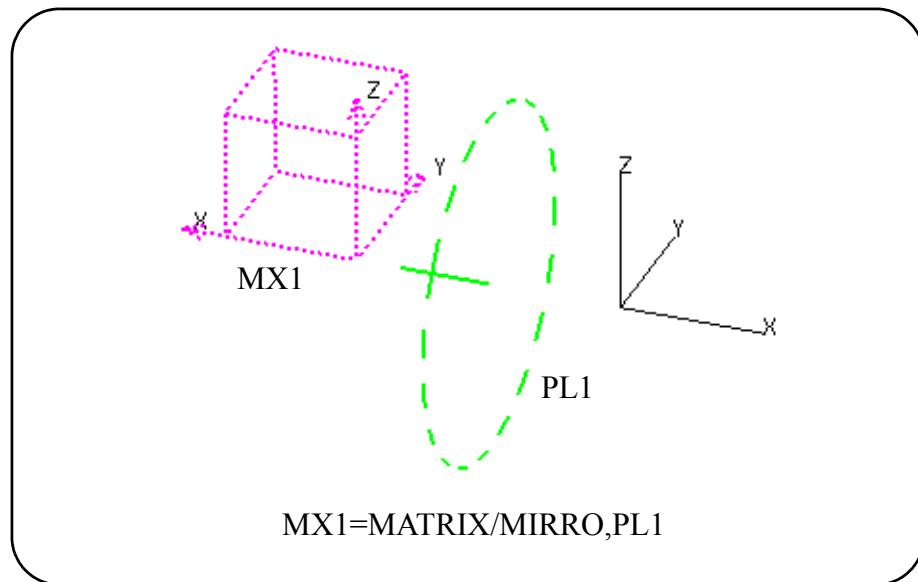
In the example MX2 performs the inverse or opposite transformation that MX1 did. MX1 did an **XYROT** of 30 degrees; MX2 does an **XYROT** (of the original coordinate system) of -30.

3.10.7 A Matrix As A Mirror Image Relative To A Specified Plane

Command Syntax:

```
MATRIX/MIRROR, plane
```

Icon Menu Sequence:



This matrix mirrors the coordinates about a specified plane.

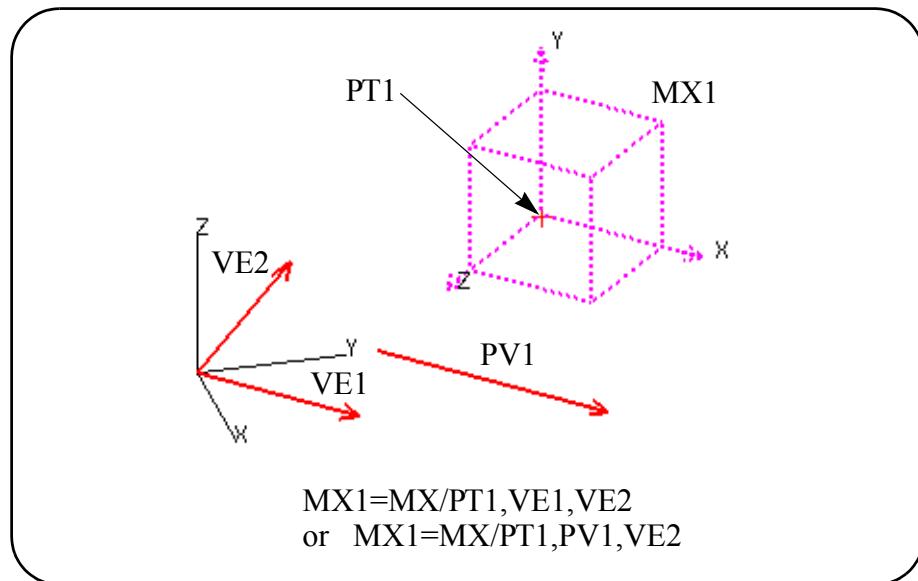
In the above example, the matrix MX1 creates a Secondary Coordinate System as the mirror image of the Primary Coordinate System about the plane PL1.

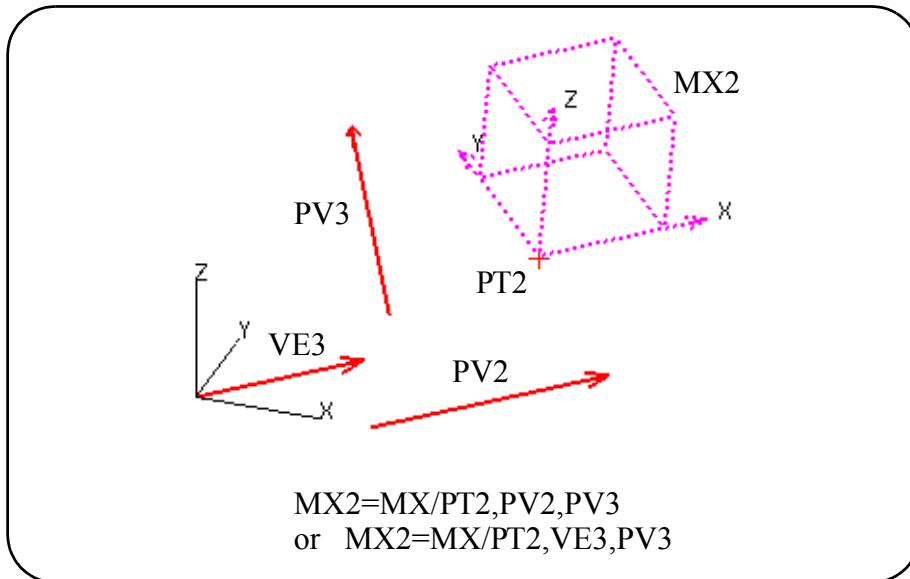
3.10.8 A Matrix In Terms Of A Point And A Combination Of Two Entities Which Can Be Vector Or Point-Vector

Command Syntax:

```
MATRIX/point, vector      , vector
          point-vector point-vector
```

Icon Menu Sequence:





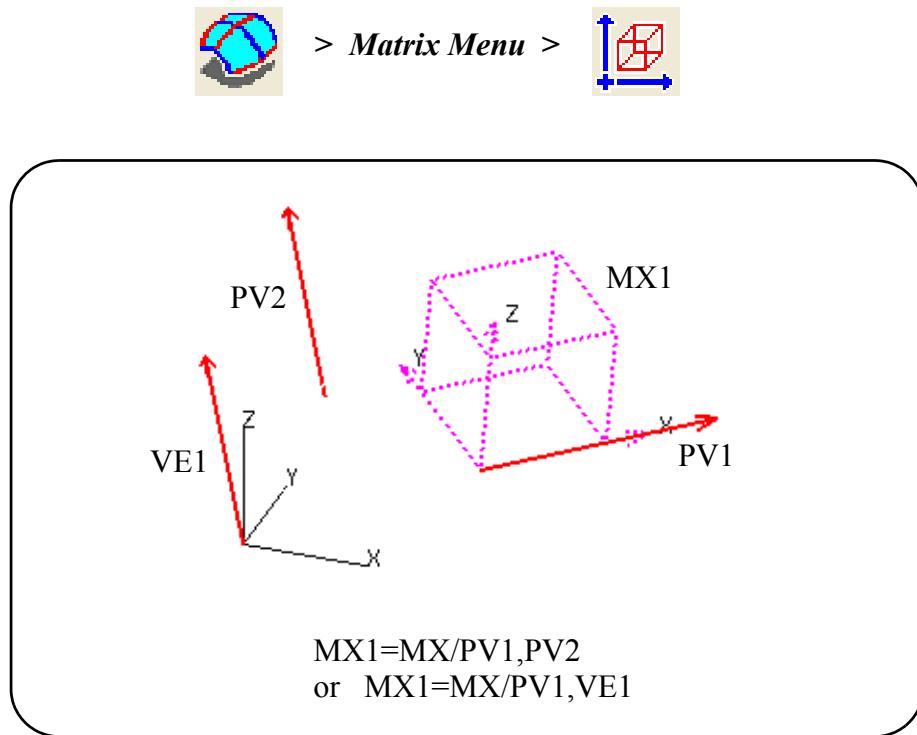
This matrix definition transforms the primary coordinate system to the specified point with a combination of two specified entities (which can be vector or point-vector). The specified point becomes the origin of the Secondary Coordinate System, the first specified entity (a vector or the vector portion of a point-vector) represents the Secondary positive X-axis and the second specified entity (the vector or the vector portion of a point-vector) represents the general direction of the Secondary positive Y-axis. The first specified entity must be pointed in the exact direction of the Secondary positive X-axis. The second entity only needs to point in the general direction of the Secondary positive Y-axis. The exact positive direction of the Secondary Y- and Z-axis will be determined by the right hand rule of an orthogonal coordinate system.

3.10.9 A Matrix In Terms Of A Point-Vector And Another Point-Vector Or Vector

Command Syntax:

```
MATRIX/point-vector,point-vector
      vector
```

Icon Menu Sequence:



This matrix definition transforms the primary coordinate system to the origin of the first specified point-vector with another point-vector or vector. The origin of the first specified point-vector becomes the origin of the Secondary Coordinate System, the vector portion of the first specified point-vector represents the Secondary positive X-axis and the second specified entity (the vector or the vector portion of a point-vector) represents the general direction of the Secondary positive Y-axis. The first specified point-vector must be pointed in the exact direction of the Secondary positive X-axis. The second entity only needs to point in the general direction of the Secondary positive Y-axis. The exact positive direction of the Secondary Y- and Z-axis will be determined by the right hand rule of an orthogonal coordinate system.

3.10.10 A Matrix Converts One Coordinate System To Another Coordinate System

Command Syntax:

```
MATRIX/in-pt1,in-pt2,in-pt3,out-pt1,out-pt2,out-pt3
```

Where:

in-pt1, in-pt2,in-pt3 Three points in the input coordinate system

out-pt1,out-pt2,out-pt3 Corresponding three points in the output coordinate system

Icon Menu Sequence:



Note: The definition of this matrix is the same as the matrix definition used by **PostWorks** for coordinate transformation which means the in-points are three points in the programming system and the out-points are the corresponding physical points on the machine system.

MULTIPLE ENTITIES TYPE GEOMETRY STATEMENTS

This section deals with the statements which can create several geometry entities with different types in one statement.

3.11.1 Creating Boundary Contours Or Their Components At The Intersection Of Groups Of Surfaces And A Plane, A Surface Or A Z-value.

NCL has the ability to create boundary contours or their components at the intersection of groups of surfaces and a plane, a planar-surface or a Z-level value with the following syntax:

```
SPLINE/INTOF, ALL      ,surface-list,          $
CURVE           COMPOS

AT,zlev          $
plane           [,modifier,offset]
planar-surf

[,num-cv-scalar]
```

Where:

ALL Means all the possible intersection will be created as individual components.

COMPOS Means only closed boundary contours will be created and all of them will be created as a composite curve.

surface-list List of surfaces that the intersection contours or components would be created from. The list can be specified as follows:

- A standard sequence of labels, e.g., sf2, sf3, etc.
- A sequence indicated by a “THRU” clause.

Example:

sf1, THRU, sf5

or

1, THRU, 5

3 GEOMETRY EXPRESSIONS MULTIPLE ENTITIES TYPE GEOMETRY

- A net surface - it is treated internally as a collection of individual surfaces.
- A layer number, e.g. LAYER=4.
- Any combination of above, separated by commas.

Note: Surfaces with same label are treated as one surface even they are specified more than once either explicitly or implied, e.g. a net surface is specified and any of the components are specified explicitly in the surface list.

zlev, plane or planar-surf Specified the Z-level value, a plane or a surface that the 2-D contour curve will be created. The specified plane or planar-surface does not need to be normal to the Z-axis of the current reference system.

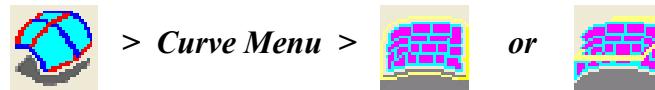
modifier,offset To specify an option offset value for the intersection plane or surface in the direction specified by XLARGE, YLARGE, ZLARGE, XSMALL, YSMALL, or ZSMALL.

num-cv-scalar An Optional scalar holds the total number of entities created with this command.

Note: Even lines, circles or curves might be created, all of them will be identified with curve labels by default. For example, if one of the components is a line and the default labelling system is utilized, this line will be identified as CV# (# is the next number available for curve assignment), displayed with the color GREEN and has the LINE entity property instead of the CURVE property. See “[Default Identifier Naming Convention of Multi-Entities Geometry Creation Commands](#)” of Chapter 2 for details.

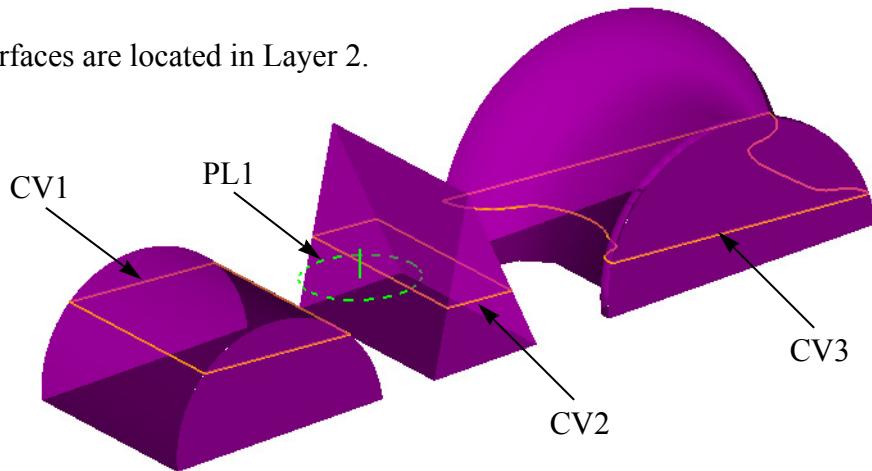
***SET/VER,9.5 must be specified for all programs written in **NCL95** that used this statement with the “ALL” option due to changes in the order of entity labelling.**

It should be noted that while the command can be entered as shown, this statement is best produced by using the on screen menu icon sequence

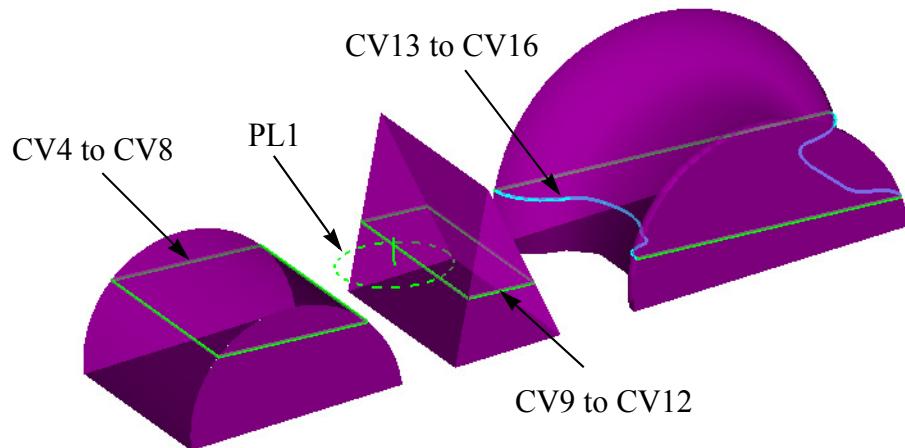


MULTIPLE ENTITIES TYPE GEOMETRY 3 GEOMETRY EXPRESSIONS

All surfaces are located in Layer 2.



CV1=SPLINE/INTOF,COMPOS,LAYER=2,AT,PL1,CNUM
“CNUM=3 after process the command.”



CV4=SPLINE/INTOF,ALL,LAYER=2,AT,PL1,CNUM

All surfaces are located in Layer 2.
CV5 thru CV12: all are LINE entities
CV13 thru CV16: 2 are LINE entities, 2 are SPLINE entities
CNUM = 12 after process the command

3.11.2 Extracting Component(s) From A Composite Curve

The command

```
CURVE/OUT, curve, ALL      [ , NUM, val ]
                           cv_list
```

allows you to extract a composite curve components as separate (labeled) geometric entities. The components are extracted as they are defined, so lines, circles, curves, splines, and composite curves could be obtained as a result.

Where:

curve	the curve that the components to be extracted
cv-list	specifies a list of components to extract from the composite curve. The word ALL can be used to extract all components of the composite curve, or a list of individual component numbers (1,3,5) can be specified, or a THRU clause (2,THRU,5) can be used.
NUM,val	“val” is a scalar variable which stores the number of component cures extracted.

Note: Even lines, circles or curves might be extracted, all of them will be identified with curve labels by default. For example, if one of the components is a line and the default labelling system is utilized, this line will be identified as CV# (# is the next number available for curve assignment), displayed with the color GREEN and has the LINE entity property instead of the CURVE property. See “[Default Identifier Naming Convention of Multi-Entities Geometry Creation Commands](#)” of Chapter 2 for details.

It should be noted that while the command can be entered as shown, this statement is best produced by using the on screen menu icon sequence



Note:

- When using this icon menu with the Thru or All options there will be a prompt for the variable to be used for the scalar variable. To use this icon

menu without providing a variable press the <**ENTER**> key or the **Done** hot key to skip providing a variable when prompted.

- If the option **ALL** is selected, **NCL** will prompt once for a scalar variable and the variable will be subscripted if more than one curve was selected during picking to have its component curves extracted. If the scalar is already subscripted, the subscript will be incremented for each curve starting at the initial subscript.
- **NCL** will prompt for a scalar variable at the end of each pick if the option **Thru** is selected.

Example:

A composite curve composed of a line, circle, curve is created with the following command.

```
CV1=CURVE/COMPOS, (LINE/PT1, PT2),    $  
                  (CIRCLE/PT1, PT2, PT3),    $  
                  (CURVE/PT3, PT4, PT5, PT6)
```

- The command CURVE/CV1,OUT,ALL generates the entities CV2, CV3 and CV4. CV2 is a *LINE* entity, CV3 is a *CIRCLE* entity and CV4 is a *CURVE* entity.
- The command CURVE/CV1,OUT,1,3 generates the entities CV2 and CV3. CV2 is a *LINE* entity and CV3 is a *CURVE* entity.
- The command CURVE/CV1,OUT,2 generates only CV2 which is a *CIRCLE* entity.

3.12 [label] = SOLID/OUT,solid,ALL [, NUM,ncomp] id-list

This command is used to extract component solids from a composite solid (not composite solid formed by a group of surfaces).

“solid” is the label of the composite solids that the components would be extracted.

“ALL” means all components solid will be extracted.

：“id-list” is a single value or a range of values that specify which components will be extracted. A “THRU” clause is allowed.

3 GEOMETRY EXPRESSIONS MULTIPLE ENTITIES TYPE GEOMETRY

“NUM,ncomp” specifies a scalar that will receive the number of components extracted.

It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence



**3.13 [label] = SURF/OUT,geo, ALL [, NUM,ncomp]
id-list**

This command is used to extract component surfaces from a net surface or a composite solid formed by a group of surfaces.

“geo” is the label of the net surface or the composite solid formed by a group of surfaces would be extracted.

“ALL” means all components surfaces will be extracted.

“id-list” is a single value or a range of values that specify which components will be extracted. A “THRU” clause is allowed.

“NUM,ncomp” specifies a scalar that will receive the number of components extracted.

It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence



SOLID

The following list gives an abbreviated notation of all the valid SOLID definition formats. See the individual sections for a complete explanation of each format.

1. **SOLID/BOX**,point,point[,minz,maxz]
x,y,z x,y,z
2. **SOLID/BOX**,CENTER,point,length,width,height
x,y,z
3. **SOLID/BOX**,BOUND[,xexp[,yexp,zexp]],goe-list
LAYER=n
ALL
MOTION
4. **SOLID/COMPOS**[,CLOSE][,INVIS],solid-list
OPEN RETAIN surf-list
REMOVE LAYER=n
ALL
5. **SOLID/CONE**,point,point,RADIUS, radius-1, radius-2
x,y,z x,y,z
6. **SOLID/CONE**,CENTER,point-vector,height,
point,vector
x,y,z,i,j,k

RADIUS,radius-1,radius-2
7. **SOLID/CYLNDR**,circle,height
8. **SOLID/CYLNDR**,point,point,RADIUS, radius
x,y,z x,y,z
9. **SOLID/CYLNDR**,CENTER,point-vector,height,
point,vector
x,y,z,i,j,k

RADIUS,radius
10. **SOLID/EXTRUD**,curve,vector,length
i,j,k

-
11. **SOLID/PART**,expansion,surface-list \$
 AYER=n
 ALL
- [,at,zlev,height] \$
 plane,height
 LEVEL[,bofs[,tofs]]
- [,OFFSET,vector][,PROFIL]
 i,j,k BOX
12. **SOLID/REVOLV**,curve,point-vector[,str-ang,end-ang]
 point,vector
 x,y,z,i,j,k
13. **SOLID/SPHERE**,circle
14. **SOLID/SPHERE**,CENTER,point,RADIUS, radius
 x,y,z
15. **SOLID/STL**[,INCHES][,EDGE,ON],"file-name"
 MM OFF
16. **SOLID/TORUS**,circle,circle
17. **SOLID/TORUS**,circle,RADIUS, radius
18. **SOLID/TORUS**,CENTER,point-vector,RADIUS,rd-1,rd-2
 point,vector
 x,y,z,i,j,k
19. **SOLID/LOAD**,"file-name"[,nent][,TRFORM,matrix]
20. **SAVE/SOLID**,STOCK , "file-name",solid-list
 FIXTUR LAYER=n
 ALL
21. **SAVE/STL**[,1][,tol],"file_name",solid-list
 2

Example Solid Expressions:

```
SOLID/BOX,PT1,PT2
SOLID/CONE,PT1,PT2,RADIUS,0.5,1
SOLID/CYLNDR,CI1,2
SOLID/EXTRUD,CV1,VE1,3
SOLID/PART,0.1,SF1,SF2,SF3
SOLID/REVOLV,CV2,PV2,45,180
SOLID/SHPERE,CI2
SOLID/TORUS,CI4,CI5
SOLID/STL,"file.stl"
SOLID/LOAD,"file.stk"
SAVE/SOLID,STOCK,"out.stk",SO2,SO4
SAVE/STL,"out.stl",SO1,SO3
```

3.14 Solids Expressions

An **NCL** SOLID is used for display purpose only and cannot be used for geometry or motion development, except for Waterline Roughing and curve intersection creation. It is incorporated as an easy way to define fixtures, clamps, stocks, etc.

Solids can be displayed either as SHADED or UNSHADED. Only the UV lines of the solids will be displayed if displayed as UNSHADED. The default condition can be changed by setting the parameter “UU_SHADING” in the file: \NCCS\NCL100\interface\ncl.init. If this parameter is set to ON, the default display of solids will be SHADED. If it is set to OFF, the default solid display will be UNSHADED. If the default condition is set to SHADED, the shading attribute of each individual solid can be modified by clicking the interface icons as follows: *TOOLS > CAM > SURFACE > SHADED SF* or *UNSHADE SF* or *MOD ATTRIB*

SEAGRN is the **NCL** System DEFAULT color for solids.

Canonical Form:

The canonical form for each type of solid is different and is not available to the user.

The only parameters which can be accessed by users with the **OBTAIN** or **CAN** statements are:

For a Box Solid:

param(1)	Type of solid, always has a value of 1.
param(2) to param(4)	x, y, z values of the originating corner point of box.
param(5) to param(7)	x, y, z values of the opposite corner point of box.
param(8) to param(10)	i, j, k values of the X-axis (length) vector with distance.
param(11) to param(13)	i, j, k values of the Y-axis (width) vector with distance.
param(14) to param(16)	i, j, k values of the Z-axis (height) vector with distance.

For a Cylinder Solid:

param(1)	Type of solid, always has a value of 2.
param(2) to param(4)	x, y, z values of the cylinder solid bottom center.
param(5) to param(7)	i, j, k values of the cylinder solid axis.
param(8)	The height of the cylinder solid.
param(9)	The radius of the cylinder solid.

For a Torus Solid:

param(1)	Type of solid, always has a value of 3.
param(2) to param(4)	x, y, z values of the torus solid center.
param(5) to param(7)	i, j, k values of the torus solid axis.
param(8)	The axial radius of the torus solid.
param(9)	The circle radius of the torus solid.

For a Sphere Solid:

param(1)	Type of solid, always has a value of 4.
param(2) to param(4)	x, y, z values of the sphere solid center.
param(5)	The radius of the sphere solid.

For a Cone Solid:

param(1)	Type of solid, always has a value of 5.
param(2) to param(4)	x, y, z values the cone solid bottom center.
param(5) to param(7)	i, j, k values of the cone solid axis.
param(8)	The height of the cone solid.
param(9)	The bottom radius of the cone solid.
param(10)	The top radius of the cone solid.

For an Extruded Solid:

param(1)	Type of solid, always has a value of 6.
param(2) to param(4)	i, j, k values of the direction of extrusion.
param(5)	The length of the extrusion.
param(6)	The number of points in the extruded curve.
param(7) to param(9)	x, y, z values for the first point of the extruded curve.

For a Contour Solid”

param(1)	Type of solid, always has a value of 7
param(2)	The value of the contour expansion.
param(3)	Type of contour. A value of “0” means a Profile Solid. A value of “1” means a Box Solid.
param(4) to param(6)	i, j, k values of the direction of extrusion.
param(7)	The length of extrusion.
param(8)	The number of points in the extruded curve.
param(9) to param(11)	x, y, z values for the first point of the extruded curve.

For a Revolved Solid:

param(1)	Type of solid, always has a value of 8
param(2) to param(4)	x, y, z values of the revolved solid center point.
param(5) to param(7)	i,j,k values of the revolved solid axis.

param(8)	The starting angle of the revolved solid.
param(9)	the ending angle of the revolved solid.
param(10)	The number of points in the revolved curve.
param(11) to param(13)	x, y, z values for the first point of the revolved curve.

STL Solid:

Param(1)	Type of solid, always has a value of 9.
----------	---

3.14.1 A Solid Box By Two Opposite Corner Points

Command Syntax:

```
SOLID/BOX,point,point[,minz,maxz]
      x,y,z x,y,z
```

Icon Menu Sequence:



This command creates a box shaped solid through two points in space.

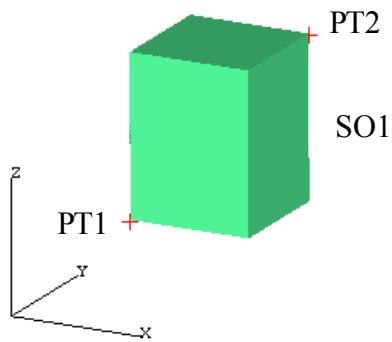
Where:

point or “x,y,z” = Defines the opposite corners of the solid box to be created.

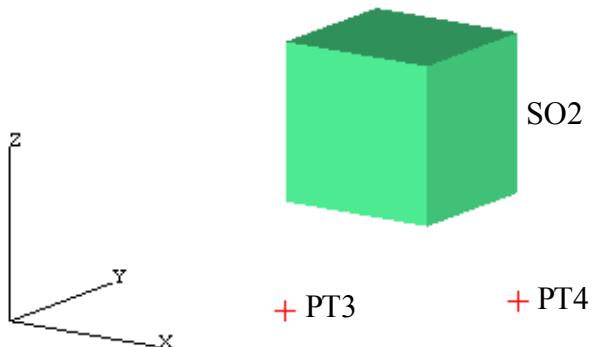
minz, maxz = Optional parameters to override the Z-values of the specified points and are required if the two points defining the opposite corners of the solid are in the same Z-level. "minz" defined the Z-level value of the bottom corner and "maxz" defined the Z-level value of the top corner of the box to be created.

Note:

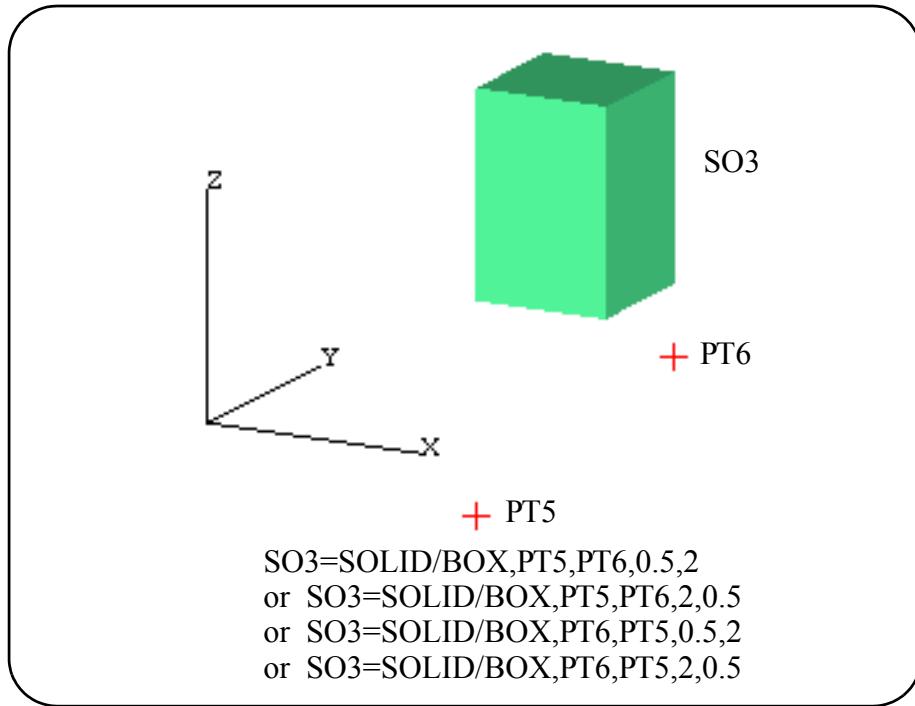
- The edges of the created box shape solid are always aligned with the current MODel SYStem, not the active REference SYSem.
- The “minz” and “maxz” are always specified in the current MODel SYSem, not the active REference SYSystem.
- The “x, y, z” are always specified in the current REference SYSystem.



SO1=SOLID/BOX,PT1,PT2
or SO1=SOLID/BOX,PT2,PT1



SO2=SOLID/BOX,PT3,PT4,0.5,1.25
or SO2=SOLID/BOX,PT3,PT4,1.25,0.5
or SO2=SOLID/BOX,PT4,PT3,0.5,1.25
or SO2=SOLID/BOX,PT4,PT3,1.25,0.5



3.14.2 A Solid Box By A Center Point, Length, Width And Height

Command Syntax:

```
SOLID/BOX,CENTER,point,length,width,height  
x,y,z
```

Icon Menu Sequence:



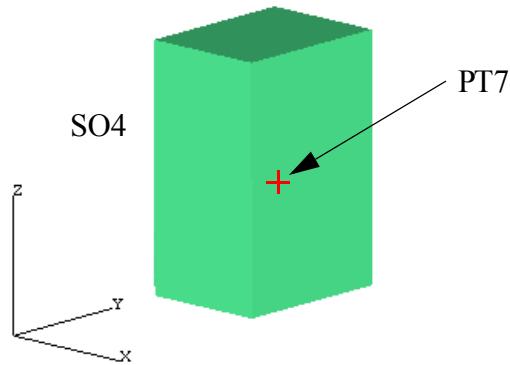
This command creates a box shaped solid centered at a point with a specified width, length and height of the box.

Where:

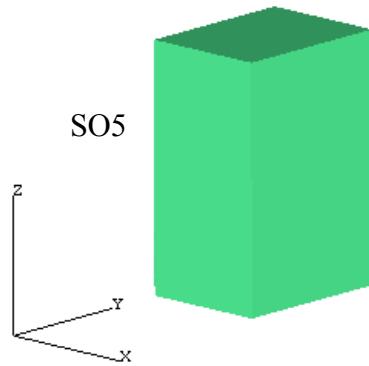
- | | |
|------------------|---|
| point or “x,y,z” | = Defines the center point of the solid box to be created. |
| width | = A numerical value defines the width of the solid box along the current MODSYS system X-axis. |
| length | = A numerical value defines the length of the solid box along the current MODSYS system Y-axis. |
| height | = A numerical value defines the height of the solid box along the current MODSYS system Z-axis. |

Note:

- The edges of the created box shape solid are always aligned with the current MODEl SYStem, not the active REFerence SYStem.
- The “x, y, z” are always specified in the current REFerence SYStem.



SO4=SOLID/BOX,CENTER,PT7,0.75,1,1.5



SO5=SOLID/BOX,CENTER,1,1,1,0.75,1,1.5

3.14.3 A Solid Box As An XYZ Bounding Box Around A List Of Geometry Or Generated Motion

Command Syntax:

```
SOLID/BOX, BOUND [,xexp[,yexp,zexp]], geo-list
    LAYER=n
    ALL
    MOTION
```

Icon Menu Sequence:



This command creates a box shaped solid as the bounding box of defined geometric entities or previously generated motion.

Where:

`xexp[,yexp,zexp]` = Defines the box expansion in the current MODel SYStem X-axis, Y-axis, and Z-axis. If only “`xexp`” is specified, then “`xcep`” defines the expansion for all three axes. If no expansion variables are specified, then an expansion will not be applied to the calculated box.

`ge-list` or `LAYER=n` or `ALL`

= A list of geometry to use to calculate the bounding box. Valid geometry types are: Point, Point-Vector, Pattern, Lines, Circle, Curve, Surface, Solid, Annotation, Symbol Master and Symbol Instance. A layer that contains the geometry to use can be used by specifying `LAYER=n`, or the word `ALL` can be used to specify all defined geometry.

Symbol Masters and Symbol Instances will not be included when either `LAYER=n` or `ALL` is specified.

MOTION = Generate a bounding box around the previously generated motion.

Note:

- MOTION bounding box with TRACUT on.

If “DRAFT/CUTTER,TRACUT=ON” is not specified, the rectangular shaped bounding box solid will be created in the current MODel SYStem disregard of the TRACUT output position. The edges of the rectangular shaped bounding box solid are always aligned with the current MODel SYStem, not the active REFerence SYStem.

If “DRAFT/CUTTER,TRACUT=ON” is specified, the rectangular shaped bounding box solid will be created in the TRACUT output position. However, the edges of the rectangular shaped bounding box solid are always aligned with the current MODel SYStem, not the active REFerence SYStem.

- Geometry bounding box.

The current MODel SYStem is used to create the rectangular shaped bounding box solid, not the active REFerence SYStem.

The edges of the rectangular shaped bounding box solid are always aligned with the current MODel SYStem, not the active REFerence SYStem.

3.14.4 A Composite Solid Created By A List Of Solids/Surfaces

Command Syntax:

```
SOLID/COMPOS [ ,CLOSE] [ ,INVIS ] ,entity-list  
OPEN      RETAIN    LAYER=n  
REMOVE    ALL
```

Icon Menu Sequence:



This command creates a composite solid from a group of solids or surfaces

Where:

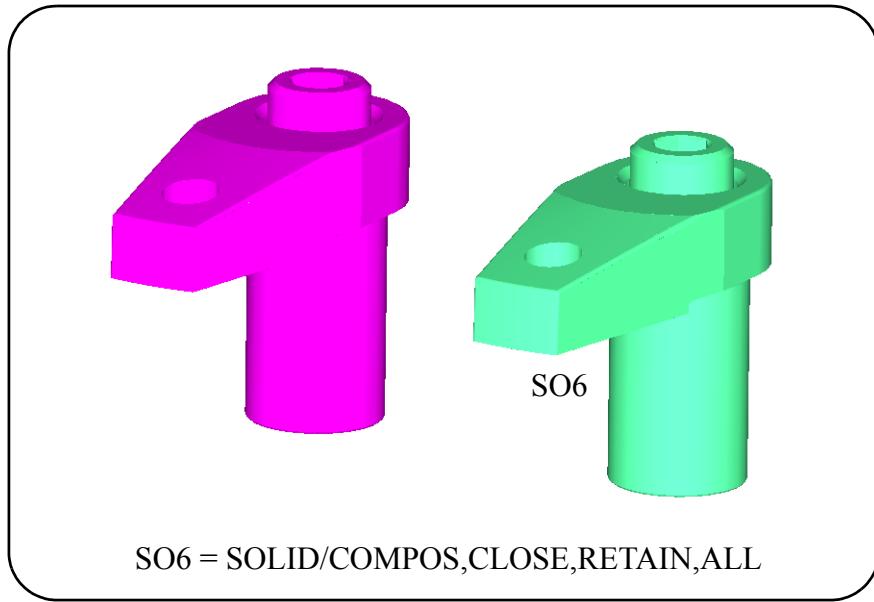
CLOSE/OPEN = “CLOSE” marks the solid as a closed solid. This setting currently has no effect on the solid, but will be used when exporting the Unibase using the STEP convertor. “OPEN” marks the solid as an open solid.

INVIS/RETRIN/REMOVE =

“INVIS” will automatically invisible all of the surfaces/solids used to create the composite solid. This is the default setting. “RETAIN” will maintains the input surfaces/solids as they are. “REMOVE” will automatically remove the input surfaces/solids after the composite solid is created.

entity-list/LAYER=n/ALL=

A list of surfaces/solids used to create the composite solid. It can be a list of geometry, a layer (LAYER,n), or the word ALL. “LAYER” and “ALL” can only be used with surfaces.



Note: SO6 shown above has been translated linear to the right
for clarification after creation.

3.14.5 A Solid Cone By Two Points And Two Radius

Command Syntax:

```
SOLID/CONE,point,point,RADIUS,radius
      x,y,z x,y,z
```

Icon Menu Sequence:

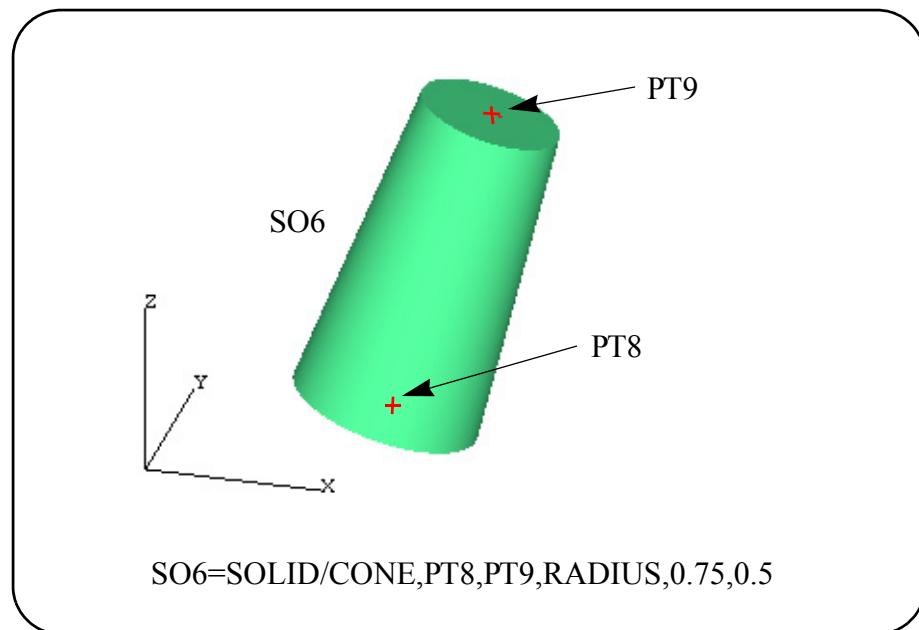


This command creates a cone shaped solid through two points with radius specified.

Where:

point or “x,y,z” = Defines the two opposite ends of the cone shape solid. This also defines the cone axis.

radius = A numerical value defines the radius of the cone solid to be created at each end.



3.14.6 A Solid Cone By A Point-Vector, Height And Two Radius

Command Syntax:

```
SOLID/CONE,CENTER,pntvec      ,height,           $  
                           point,vector  
                           x,y,z,i,j,k  
  
                           RADIUS,radius,radius
```

Icon Menu Sequence:

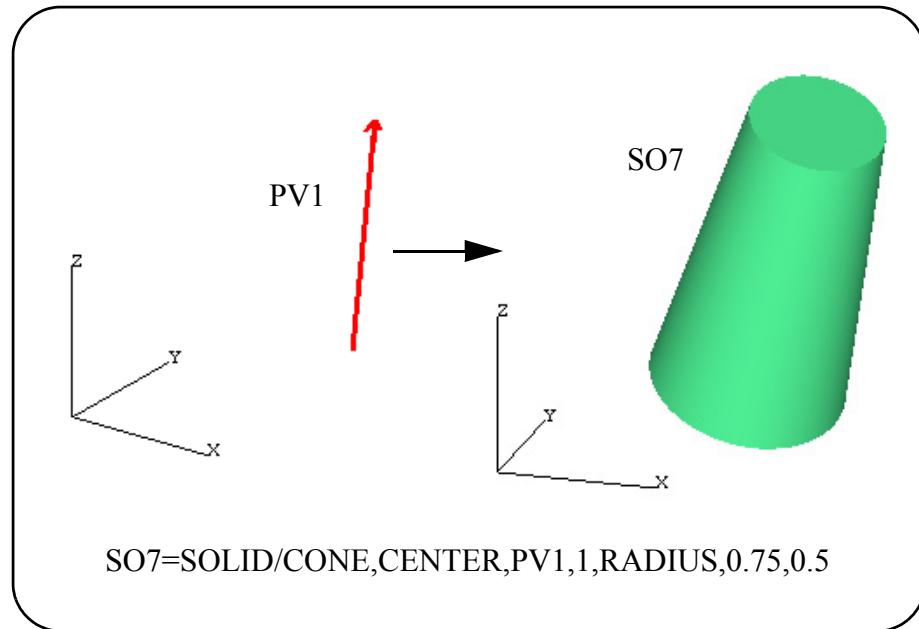


This command creates a cone shaped solid through a point, along a vector with height and the upper and lower radius specified.

Where:

pntvec or “point, vector” or “x, y, z, i, j, k”

- = Defines the one end and the axis of the cone shaped solid.
- height = A numerical value defines the height of the cone shaped solid. A negative value means the axis direction will be opposite in direction to the specified “vector”. A positive value means the axis direction will be in the same direction of the specified “vector”.
- radius = A numerical value defines the radius of the cone solid to be created at each end.



3.14.7 A Solid Cylinder By A Circle And Height

Command Syntax:

```
SOLID/CYLNDR,cirlce,height
```

Icon Menu Sequence:

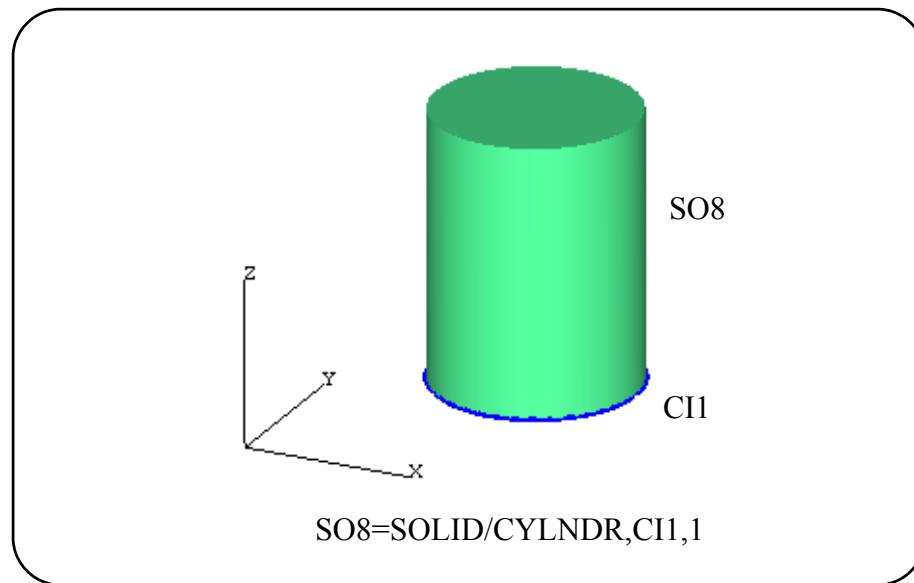


This command creates a cylindrical shaped solid through a circle with a specified height along the axis normal to the specified circle.

Where:

circle = A circle which defined one end of the circular shaped solid to be created.

height = A numerical value defines the height of the circular shaped solid. A negative value means the extrusion will be along a vector opposite in direction to the vector normal to the specified circle. A positive value means the extrusion will be in the same direction of the vector normal to the specified circle.



3.14.8 A Solid Cylinder By Two End Points And Radius

Command Syntax:

```
SOLID/CYLNDR,point,point,RADIUS, radius
x,y,z x,y,z
```

Icon Menu Sequence:

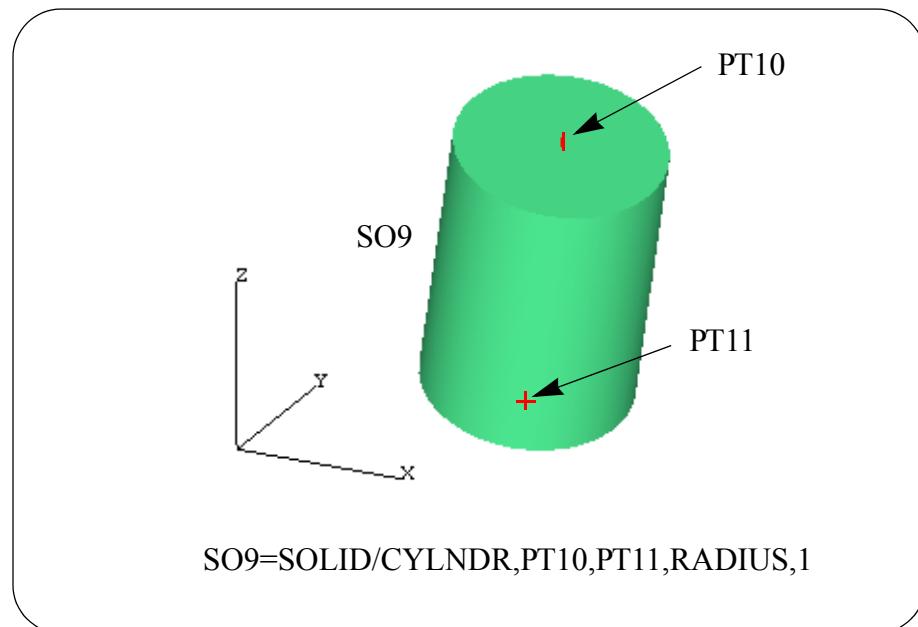


This command creates a cylindrical shaped solid with its axis going through two points with radius specified.

Where:

point or “x,y,z” = Defines the two opposite ends of the cylindrical shape solid. This also defines the cylindrical axis.

radius = A numerical value defines the radius of the cylindrical solid to be created.



3.14.9 A Solid Cylinder By A Point-Vector, Height And Radius

Command Syntax:

```
SOLID/CYLNDR,pntvec      ,height,RADIUS, radius
                  point, vector
                  x,y,z,x,y,z
```

Icon Menu Sequence:



This command creates a cylindrical shaped solid through a point with the axis along a vector, height and radius specified.

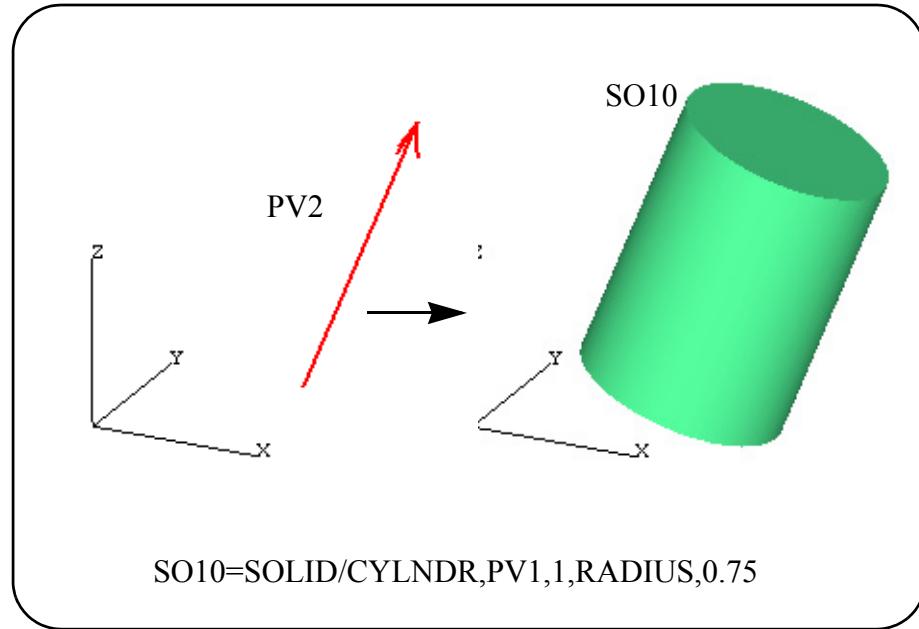
Where:

pntvec or “point, vector” or “x, y, z, i, j, k”

= Defines the end and the axis of the cylindrical shape solid.

height = A numerical value defines the height of the circular shaped solid. A negative value means the extrusion will be along a vector opposite in direction to the specified “vector”. A positive value means the extrusion will be in the same direction of the specified “vector”.

radius = A numerical value defines the radius of the cylindrical solid to be created.



3.14.10 A Solid Extrusion By Lifting A Closed Planar Curve With Distance Along A Vector

Command Syntax:

```
SOLID/EXTRUD,curve ,vector,length  
circle i,j,k
```

Icon Menu Sequence:



This command creates a solid extrusion by lifting a curve along a vector with distance specified.

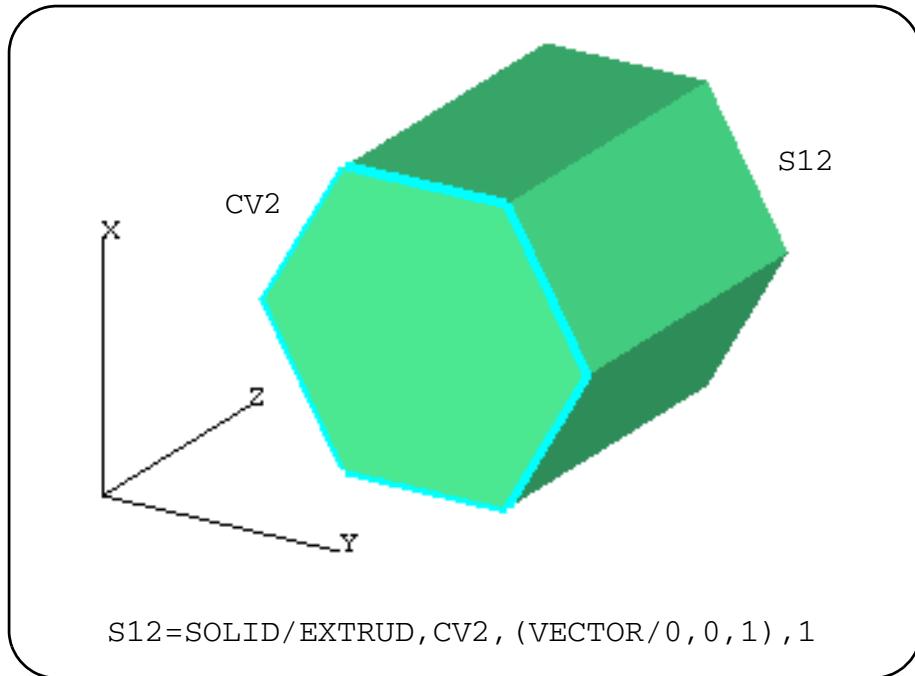
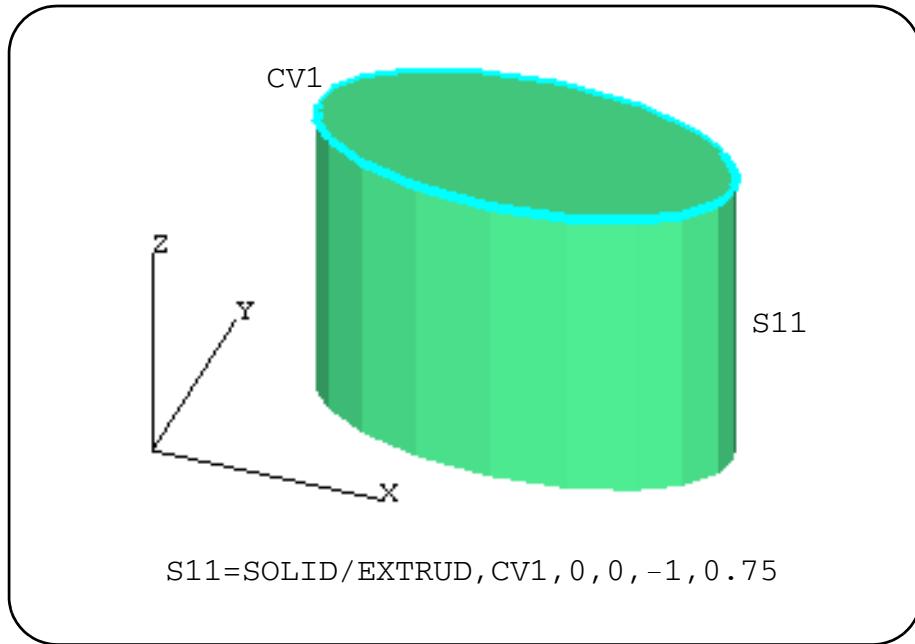
Where:

curve or circle

= The curve or circle to be lifted. The curve must be a closed planar curve. The circle must be a full circle.

vector = Defines the direction of the extrusion and cannot parallel to plane of the curve or circle.

length = Defines the length of the extrusion.



3.14.11 A Solid Extrusion By Following the Contour Of The Part

Command Syntax:

```
SOLID/PART, expansion, surface-list $  
          LAYER=n  
          ALL  
  
          [, AT, zlevel, height ] $  
          plane, height  
          LEVEL[, bofs, [, tofs]]  
  
          [, OFFSET, vector] [, PROFIL]  
          i, j, k     BOX
```

Icon Menu Sequence:



This command creates a solid extrusion that follows the contour of the part.

Where:

expansion = Specifies the horizontal expansion of the calculated part contour.

surface-list or LAYER=n or ALL

= A list of surfaces to use to create the part contour. A standard surface list can be specified, a layer that contains the surface to use can be used by specifying LAYER=n, or the word ALL can be used to specify all surfaces.

AT = Specifies the lower and upper bounds of the contour solid.

zlevel, height

= Specifies the bottom Z-level and the height of the solid.

plane, height

- = Specifies a plane to use as the bottom of the solid (which does not have to be a Z-level plane) and the height of the solid.

LEVEL, bofs, tofs

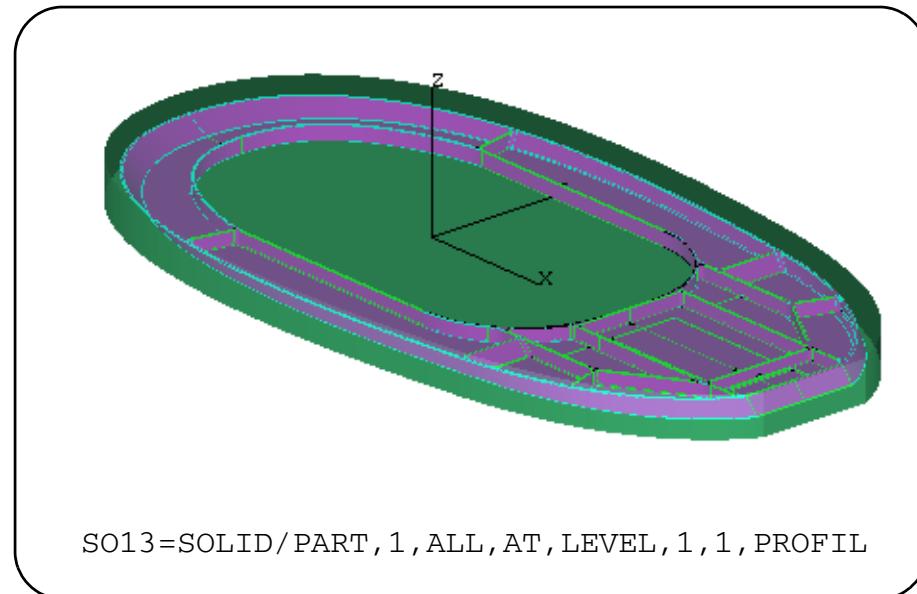
- = Specifies that the calculated bottom and top of the part contour be used with optional bottom and top vertical offsets.

“OFFSET, vector” or “OFFSET, i, j, k”

- = Specifies the directional vector to use to extrude the contour. This vector defaults to the plane normal used as the bottom of the contour.

PROFIL = Generates a solid that follows the profile of the calculated part contour.

BOX = Generates a box style solid that fully encloses the calculated part contour.



3.14.12 A Solid By Revolving A Planar Curve Around A Point-Vector

Command Syntax:

```
SOLID/REVOLV,curve,pntvec $  
point,vector  
x,y,z,i,j,k  
  
[,start-angle,end-angle]
```

Icon Menu Sequence:



This command creates a solid by revolving a planar curve around a point-vector.

Where:

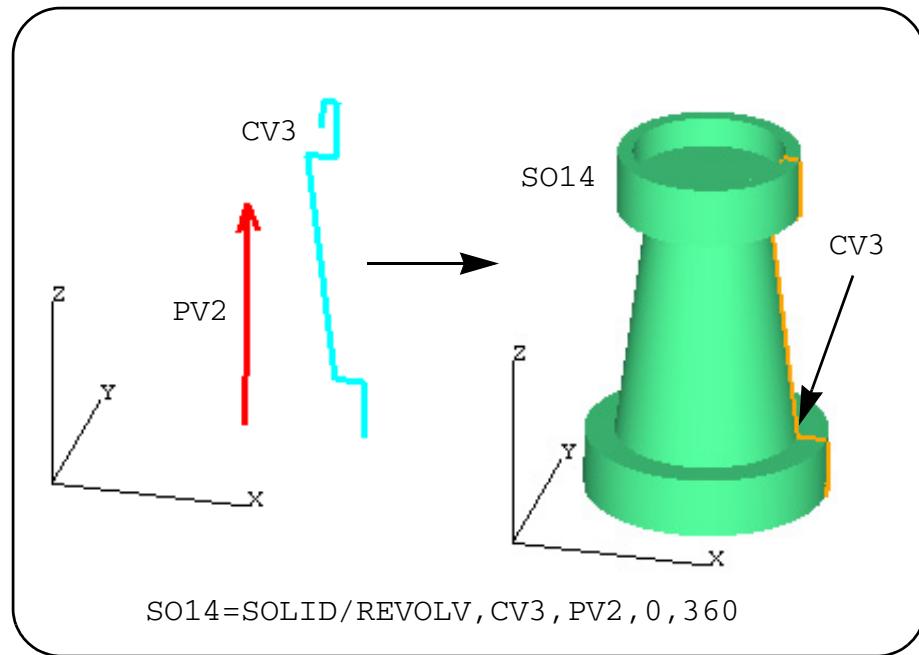
curve = The curve to be revolved. The curve must be planar. It can be a curve, spline, line or circle.

pntvec or “point, vector” or “x, y, z, i, j, k”

= Defines the revolving origin and axis.

start-angle and **end-angle**

= Optional parameters to define the starting angle and the ending angle in degrees. The solid will be defined in a counter-clockwise (CCLW) direction from “start-angle” to “end-angle”. The default angles are 0 and 360, specifying an untrimmed solid.



3.14.13 A Solid Sphere By A Circle

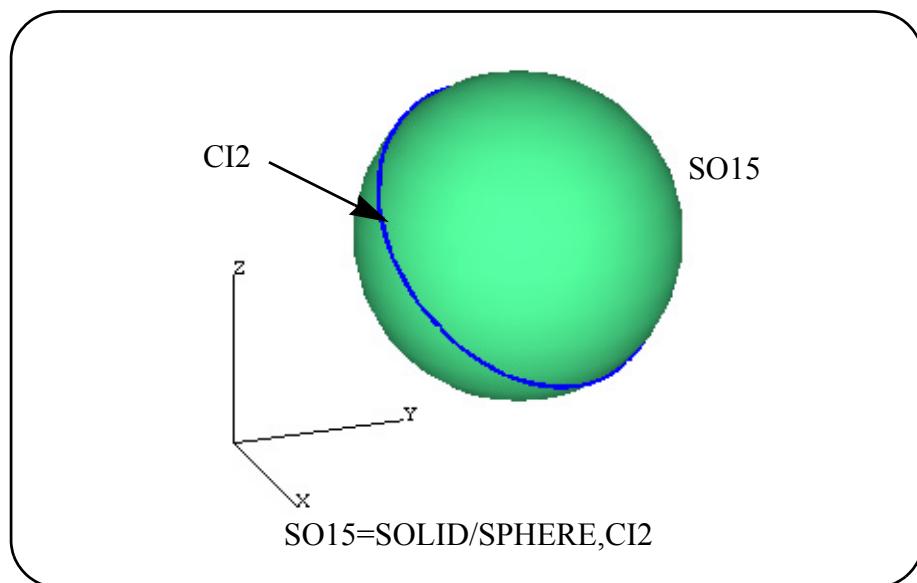
Command Syntax:

`SOLID/SPHERE,circle`

Icon Menu Sequence:



This command creates a spherical shaped solid with the center point and radius of a circle.



3.14.14 A Solid Sphere By A Center Point And Radius

Command Syntax:

```
SOLID/SPHERE,CENTER,point,RADIUS,radius  
x,y,z
```

Icon Menu Sequence:

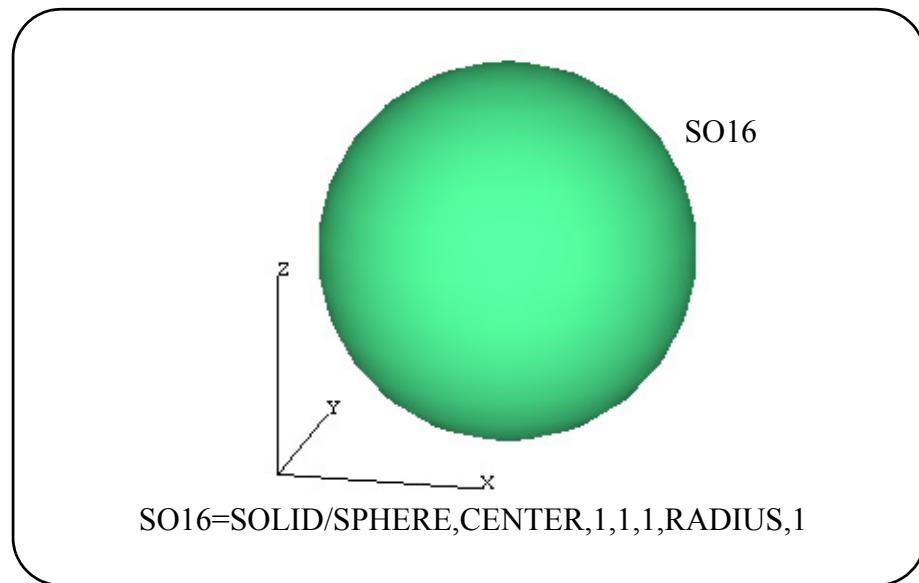


This command creates a spherical shaped solid with a center point and a radius.

Where:

point or “x,y,z” = Defines the center of the spherical solid to be created.

radius = A numerical value defines the radius of the spherical solid to be created



3.14.15 A Solid By Importing An External STL File

Command Syntax:

```
SOLID/STL [ , INCHES ] [ , EDGE, ON ] , "file"
          MM                  OFF
```

Icon Menu Sequence:



This command will load an external STL file and creates solids out of it.

Where:

“file” = Specifies the name of the stl file. The file name extension must be specified and all of them must be enclosed by a pair of double quotes. **NCL** will first look in the current directory for the file and if it is not found there, it will look in the system directory which is defined by the "NCL_INCDIR" parameter inside the ncl.init file. If there are multiple solids associated with this STL file, then **NCL** will continue numbering them in ascending order.

EDGE,ON or OFF

= Specifies if the STL solid edge display will be optimized to remove the inner edges of a planar face (EDGE,ON) or if the solid edge display will contain all defined edges within the STL model (EDGE,OFF).

INCHES or MM

= An optional parameter to specify the unit that the STL file is actually stored in. The default is the current modeling units.

Notes:

The STL models are always loaded in with the current MODSYS system, not the active REFSYS system.

STL models are stored as a series of triangles that are used to describe the faces of the STL solid. Many triangles may be necessary to define a planar face of the solid depending on the complexity of the boundary of the planar face.

When optimization of the STL solid edge display is enabled **NCL** will remove the inner triangle edges of a planar face leaving only the boundary edge of the face.

Optimizing the edge display of an STL solid can be time consuming depending on the complexity of the solid. Disabling this feature will leave the edge display as it is defined in the STL file and will load complex solids much quicker, but the edge or wireframe display will be quite busy (a lot of extraneous edges will be displayed).

On complex models, disabling the edge display optimization can improve the loading speed of an STL file by up to 3x faster.

As a general rule, STL models that do not have planar faces, or a small number compared to the rest of the model, can be imported with the edge display optimization disabled. STL models that contain a number of planar faces should have the edge display optimization enabled.

STL models once stored in a Unibase can be retrieved rather quickly no matter if the solid edge display optimization has been enabled or not. Loading STL models into a standalone Unibase and then retrieving these solid models in the main program may be a more desirable method to improve the speed of loading large STL models rather than disabling the edge display optimization.

3.14.16 A Solid Torus By Two Circles

Command Syntax:

```
SOLID/TORUS,circle_1,circle_2
```

Icon Menu Sequence:

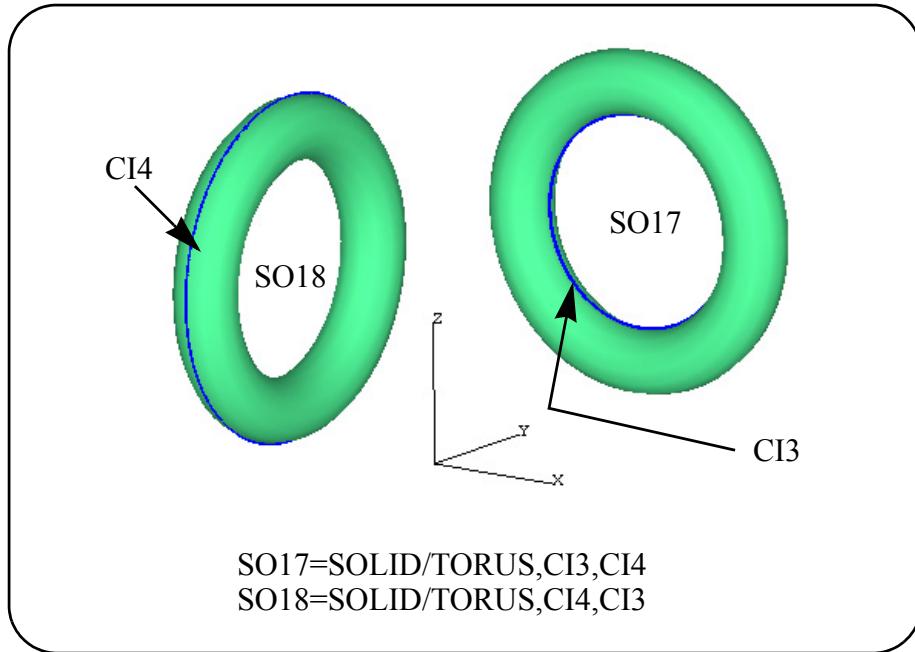


This command creates a torus shaped solid using a first circle to define its center, axis, inner/outer radius and a second circle to define the outer/inner radius.

Where:

circle_1 = A circle which defines the center, axis and inner/outer radius of the torus.

circle_2 = A circle which defines the outer/inner radius of the torus. This circle does not require to be parallel or concentric with the first circle. Only the radius will be utilized. If the radius of this circle is smaller than the radius of the first circle, this circle specified the inner radius of the torus and the first circle specifies the outer radius of the torus.



3.14.17 A Solid Torus By A Circle And A Radius

Command Syntax:

```
SOLID/TORUS,circle,RADIUS,radius
```

Icon Menu Sequence:

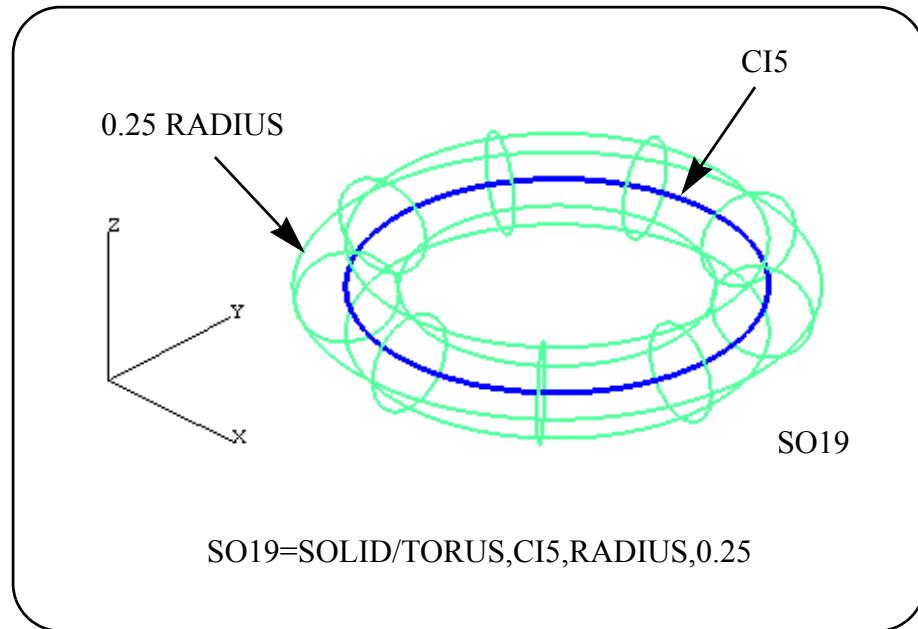


This command creates a torus shaped solid using a circle to define the center, axis, axial radius and a value to specify the circular radius.

Where:

circle = A circle which defined the center, axis and axial radius of the torus.

radius = A numerical value defines the torus circular radius. This value cannot be bigger than the radius of the specified circle.



3.14.18 A Solid Torus By A Point-Vector And Two Radius

Command Syntax:

```
SOLID/TORUS,CENTER,pntvec      ,RADIUS,rad1,rad2  
          point,vector  
          x,y,z,i,j,k
```

Icon Menu Sequence:

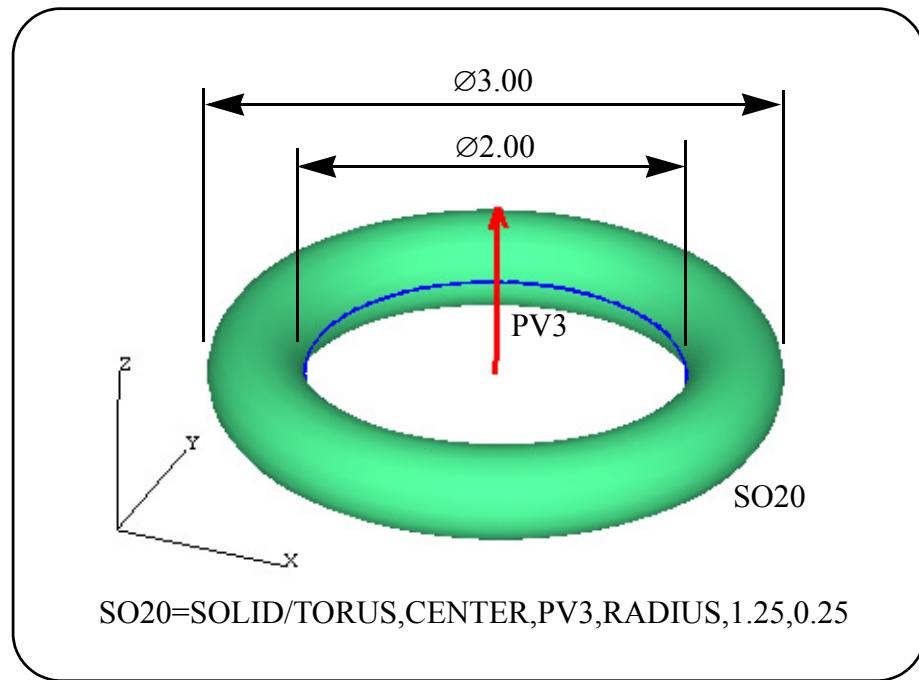


This command creates a torus shaped solid using a center point, axis, axial radius, and circular radius.

Where:

pntvec or “point , vector” or or “x, y, z, i, j, k”

- = Defines the center and the axis of the torus shape solid
- rad_1 = A numerical value defines the axial radius of the torus solid to be created.
- rad_2 = A numerical value defines the circular radius of the torus solid to be created. This value cannot be larger than “rad_1”.



3.14.19 A Solid By Loading An External Stock File

Command Syntax:

```
SOLID/LOAD, "file" [,nent] [,TRFORM,matrix]
```

Icon Menu Sequence:



This command will load an **NCL/IPV** primitives file and creates solid(s) from the stock commands.

Where:

“file” = Specifies the name of the stock file to be loaded. The file extension must be specified and all of them must be enclosed by a pair of double quotes. **NCL** will first look in the current directory for the file and if it is not found there, it will look in the system directory which is defined by the "NCL_INCDIR" parameter inside the ncl.init file. If there are multiple stocks/fixtures in the primitive file, then **NCL** will continue numbering them in ascending order.

nent = An optional parameter to specifies a scalar to receive the number of solids actually defined from the file.

TRFORM, matrix

= Specifies a matrix to modify the position and/or orientation of the solid upon loading it.

3.14.20 Save Defined Solids As An External Stock File

Command Syntax:

```
SAVE/SOLID, STOCK , "file", solid-list
          FIXTUR           LAYER=n
                      ALL
```

Icon Menu Sequence:



This command will save the specified solids as an **NCL/IPV** primitives file.

Where:

STOCK or FIXTUR

- = STOCK specifies an **NCL/IPV** primitive stock file will be saved. FIXTUR specifies an **NCL/IPV** primitive fixture file will be saved.
- "file" = Specifies the name of the stock file to be saved. The file extension must be specified and all of them must be enclosed by a pair of double quotes.

solid-list or LAYER=n or ALL

- = A list of solids to be saved to the external **NCL/IPV** primitives file. A standard solid list can be specified, a layer that contains the solid to use can be used by specifying "LAYER=n", or the word ALL can be used to specify all solid.

3.14.21 Save Defined Solids As An External STL File

Command Syntax:

```
SAVE/STL [,1] [,tolerance], "file", solid-list  
2
```

Icon Menu Sequence:



This command will save the listed solids or surfaces as an external STL file.

Where:

1 or 2 = “1” specifies the STL file will be saved in ASCII format. “2” specifies the STL file will be saved in binary format. “1” is the default setting.

tolerance = Specifies the tolerance to use to generate the STL model.

“file” = Specifies the name of the STL file to be saved. The file extension must be specified and all of them must be enclosed by a pair of double quotes.

solid-list = A list of solids or surfaces to be saved to the external STL file.

ANNOTATIONS

3.15 ANOTE

The ANOTE command is used to add annotation (lettering) to a part.

The valid syntax of the ANOTE command is:

```
ANOTE/text [ ,AT,LETTER] [ ,LETTER, pos2] $  

          LINE  

          pos1  

[ , LETTER, pos3] [ , CURVE, cv1] $  

[ , PROJCT [ , ve1] , sf1 [ , NOWRAP] [ , START ] ]  

          WRAP      MIDDLE  

          REVOLV   END  

          RADIAL    CENTER  

          AT, pt1
```

Where:

- “text” can either be a text string enclosed in double quotes (“xxx”) or a text variable and designates the annotation text to create.
- Multiple lines of text can be created using a single ANOTE command by specifying the end of line escape sequence (\n) in the text. For example:

```
AN1=ANOTE/"First Line\nSecond Line\nThird Line"
```

- Superscripts and subscripts can be contained in the text by enclosing the superscript/subscript text in brackets using the following format.

```
ANOTE/"text{ [fmt] super/subscript }more text"
```

The braces ({{}}) in the above example designate the optional superscript/subscript clauses and should not be included in the text. The opening bracket and both closing brackets are mandatory when specifying a superscript/subscript. fmt can be one of the following values.

U{ofs} = Superscript text. ofs = Height offset to position subscript at as a percentage of the character height. The default value is .6.

L{ofs} = Subscript text. ofs is the same as for superscript text. The default value is .5.

H{hgt} = Overrides the default superscript/subscript character height.

Example:

ANOTE./”[H.75U]Superscript]Text[H.75I]subscript”

will generate the following annotation.

SuperscriptTextSubscript

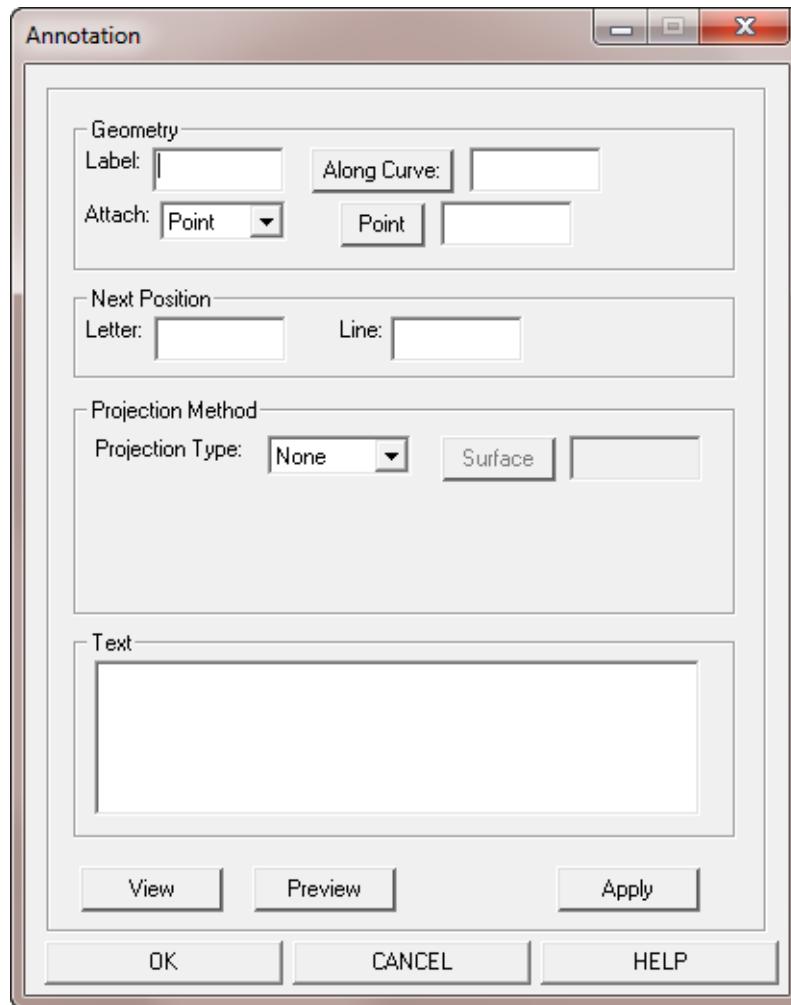
In order to actually include a left bracket “[“ as part of the text, it must be preceded by the backslash “\” character.

- A maximum of 255 characters is allowed in “text” with all format characters included if existed.
- “AT” specifies the position of the annotation text.
 - “LETTER” will position the text at the next text position of the previously defined annotation. It is the same as if this text was appended to the previous annotation, but it will be a different entity and can have different attributes (font,color, etc.).
 - “LINE” positions the text at the beginning of the next line of the previous defined annotation.
 - “pos1” specifies an exact location to position the text at. It can be a point or a point-vector. If a point-vector is specified, only the origin point of the point-vector will be used.
 - The default setting is LINE.
- “Letter, pos2” defines a point-vector or point label that will receive the defined location for the character immediately following the annotation text.

- “LINE, pos3” defines a point-vector or point label that will receive the defined location for the character that will start the next line following the annotation text.
- “CURVE, cv1” defines an optional arc or a line that the annotation text will follow.
- “PROJCT” projects the text onto a surface.
 - “Ve1” is an optional vector used for projecting all text polylines onto the surface when NOWRAP is specified, or for projecting the attach point of the text onto the surface for other projection types.
 - “sfl” is the surface to project to.
 - “NOWRAP”, “WARP”, “REVOLV”, and “RADIAL” specify the type of projection to perform and are the same as the standard projection parameters used for curves and patterns.
 - “START” begins the projection at the annotation text attach point.
 - “MIDDLE” begins the projection half way along the text width at the same height as the attached point.
 - “END” begins the projection at the end of the line at the same height as the attach point.
 - “CENTER” uses the center of the calculated text box as the start of the projection.
 - “AT, pt1” specifies a specific location for the start of the projection.

It should be noted that while this command can be entered as shown, it can also be produced from the graphical interactive interface as shown below by using the following on screen menu sequences:





This form composed of the following sections: Geometry, Next Position, Projection Method, Text, and Function Buttons.

Geometry:

This section composed of four items: Label, Along Curve, Attach and Point. They are described as follows:

Label:

Input the optional geometry label name for the annotation. If a label is not entered, then the standard NCL naming convention will be used (AN1, AN2, etc).

Along Curve:

Press this button to select a line or circle for the annotation text to follow. The curve name may also be typed into the field to the right of this button.

Attach:

The annotation can be positioned at a point or point-vector (Point), after the last character of the previously define annotation (Letter), or on the line following the last annotation (Line).

Point:

Press this button to select the point or location to position the annotation at when the Attach field is set to Point. The point or location may also be typed into the field to the right of this button.

Next Position:

This section composed of four items: Letter and Line. They are described as follows:

Letter:

Enter a point-vector or point label that will receive the defined location for the character immediately following the previously defined annotation text. This field can be left blank.

Line:

Enter a point-vector or point label that will receive the defined location for the character that will start the next line following the previously defined annotation text. This field can be left blank.

Projection Method:

This section composed of up to five items: Projection Type, Surface, Vector, Attach Point and Point. They are described as follows:

Projection Type:

Select the type of projection that is desired from the following choices.

- None - The annotation is not projected onto a surface.
- Normal - The projection will occur normal to the surface.
- Vector - The projection will be along a fixed vector.
- Wrap - Wraps the annotation onto the surface using an attach point as reference and using the tangencies and distances of each geometry position to maintain the original shape of the geometry.
- Revolve - Revolves the annotation onto a surface of revolution by using an attach point as reference and by travelling vertically and horizontally along the axis of rotation. The size of the geometry will be approximately maintained, but the shape may not.
- Radial - Revolves the annotation onto a surface of revolution by using an attach point as reference and by travelling vertically and horizontally along the axis of rotation. This type of projection is similar to Revolve except that the shape of the geometry will be approximately maintained, but not necessarily the size.

Surface:

Press this button to select the surface or plane to project the annotation to. The surface/plane name may also be typed into the field to the right of this button. This button is deactivated if the projection type is NONE.

Vector

- This field is only active when the projection type is not NONE or NORMAL.
- Selects a vector to be used to project the geometry onto the surface/plane when the projection type is Vector. For continuous projection types, this vector will be used to project the attach point onto the surface. Leaving this field blank will use a vector normal to the surface.
- A vector may be selected from the screen, a label typed in, or the i,j,k components typed in.

Attach Point:

- This field is only active when a continuous projection type is WARP, REVOLV or RADIAL.

- This selects the initial projection point that will be used to control the projection of the annotation in order to maintain its shape and size.
- The following types of attach points can be used.

Start	- Use the starting point of the annotation.
Middle	- Use a point half way along the length of the annotation.
End	- Use the ending point of the annotation.
Center	- Use a point calculated as the center of the bounding box of the annotation.
Point	- Use a user defined point.

Point:

- This field is only active if the Attached Point is of the type POINT.
- Selects the user defined attach point to be used for continuos projections. A point may be selected from the screen, a label typed in, or the x,y,z components typed in.

Text

Enter the text of the annotation. Multiple lines of text can be entered.

Note: If the text entered into the Annotation form is longer than 40 characters, then the text will be defined in variables prior to the ANOTE command using the following syntax.

```

@TX1 = "This is the first line of a multiple lin"
@TX2 = "e\nAnnotation that is longer than 40 cha"
@TX3 = "racters"
@TX4 = @TX1 & @TX2 & @TX3
AN1=ANOTE/@TX4

```

```

@TX1="This is a single line annotation with mo"
@TX2="re than 40 characters."
@TX3=@TX1&@TX2
AN2=ANOTE/@TX3

```

Function Buttons:

This section composed of three items: View, Preview and Apply. They are described as follows:

View:

Click to activate dynamic viewing.

Preview:

Click to preview the annotation without creating permanent geometry. The annotation can be viewed while the form is active, but it will be deleted when the form is existed or another annotation is created.

Apply:

Creates the annotation without taking down the form so that other annotations can be created.

OK:

Click this button to accept the entries, generate the annotation, output the ANOTE command and close the Annotation form.

CANCEL:

Click this button to cancel all the changes and close the Annotation form.

HELP:

Click this button to display a brief description of the Annotation form.

3.16 Annotation Attributes

The default annotation attributes can be defined or existing annotation's attributes can be modified by the following command.

```
DRAFT/ANOTE [ , MODIFY=anote-list] [ , COLOR=col]           $
               ALL

               [ , SIZE=hgt, exp] [ , ATANGL=ang]           $
               [ , FONT="name"] [ , STYLE=STROKE]           $
               STRING

               [ , START=pos] [ , LETDIR=LEFT ]           $
               RIGHT
               UP
               DOWN

               [ , SPACE=hor, ver] [ , LINWGT=dens]       $
               [ , LAYER=lay] [ , PEN=pn]
```

Where:

- “MODIFY” states that the annotation entities will be modified with the parameters in the command.
 - A single entity, a list of entities, or “ALL” entities can be modified.
 - If the MODIFY clause is not specified in this command, then the default annotation attributes will be set.
- “COLOR” specifies the new annotation color. It can be one of the standard color specifications or corresponding numeric values: BLACK(0), WHITE(1), BLUE (2), RED(3), GREEN(4), MAGNTA(5), YELLOW(6), CYAN(7), BROWN(8), LTTAN(9), LTBLUE(10), SEAGRN(11), ORANGE(12), PINK(13), PURPLE(14) and GREY(15).
- “SIZE” defines the character height hgt) and character expansion (exp). The character expansion is a ratio of the character width to the character height.
- “ATANGL” defines the angle in degrees of the annotation direction. A positive value will rotate the text in the clockwise direction.
- “FONT” specifies the font to use for stroke style annotation. This font must exist and be defined as an environmental variable in the format FONT name.
- “STYLE” specifies the type of the annotation to create. Either drawn geometry text (STROKE) or graphics text (STRING) can be created.
 - “STROKE” style text is drawn as a series of polylines in 3-D and can be projected onto surfaces and machined using the PROFIL command. This text will also be rotated with the part.
 - “STRING” text is the same as the graphics hardware text used to display geometry labels. It is 2-D in nature and cannot be projected onto a surface nor machined with the PROFIL command. It will always be displayed on the graphics screen plane, only its position will be rotated with the part.
- “START” specifies the attach point for annotation.

- “pos” is a value from 0 to 8 that specifies a site location on the bounding box of the annotation of where it should be positioned at the text origin point. The following diagram and corresponding defined values illustrate where the attach points of the annotation are located.

0 = Top Left	3 = Top Center	6 = Top Right
1 = Middle Left	4 = Middle Center	7 = Middle Right
2 = Bottom Left	5 = Bottom Center	8 = Bottom Right

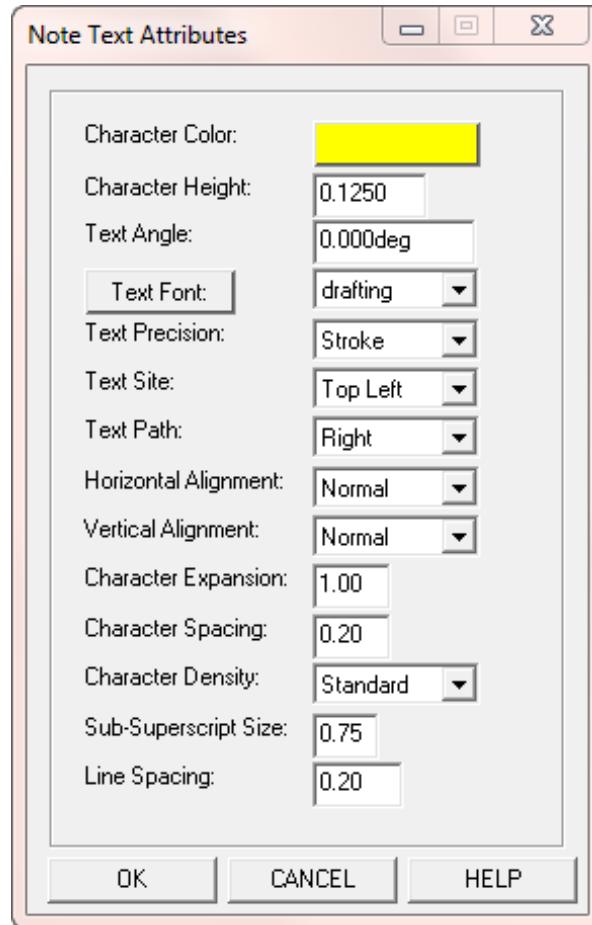
- “LETDIR” controls the text path of the annotation. The path direction can be RIGHT, LEFT, UP or DOWN.
- “ALIGN” specifies both the horizontal and vertically alignment of the annotation.
 - “hor” specifies the horizontal text alignment and can be NORMAL, LEFT, RIGHT, or CENTER.
 - “ver” specifies the vertical text alignment and can be NORMAL, UP, CENTER, BASE, or DOWN.
 - “NORMAL” selects the proper alignment based on the text path.
 - “BASE” offsets the vertical text position by the size of the descending portion of the font.
 - The horizontal and vertical alignments are dependant on the attach point.
- “SPACE” defines the character (hor) and line (ver) spacing for annotation.
- “LINWGT” specifies the character density (line weight) of the annotation. It can have one of the standard line weight specifications or corresponding numeric values: STD(1), MEDIUM(2), HEAVY(3), EXHVY(4).
- “LAYER” is only valid when the MODIFY clause is used and specifies the new layer number for existing annotation. To set the default layer for annotation along with all other geometric entities, use the standard DRAFT/MODIFY command.
- “PEN” is only valid when the MODIFY clause is used and specifies the new pen number to use when plotting for the existing annotation. To set the default pen for annotation along with all other geometric entities, use the standard DRAFT/MODIFY command.

It should be noted that while this command can be entered as shown, it can also be produced from the graphical interactive interface as shown on next page by using the following on screen menu sequences for setting the default attributes:



or

using the following on screen sequences for modifying the annotations attributes.



Character Color:

Defines the color of the annotation text.

Character Height:

Specifies the character height. The character width will be automatically adjusted using its predefined ratio to the character height.

Text Angle:

Specifies the angle in degrees at which the annotation text is displayed. A positive value will rotate the text in the clockwise direction.

Text Font:

This is a toggle field that displays all defined fonts that can be used for annotation. Select the name of the font to use to display stroked text.

Text Precision:

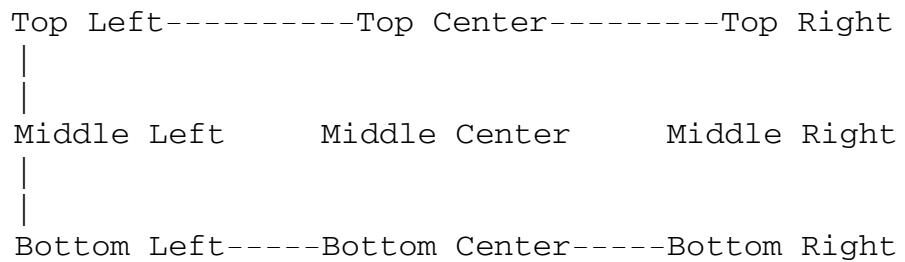
Either hardware or stroked (Hershey) text can be displayed for annotation. Select either "String" for hardware text or "Stroke" for stroked text. "Char" currently is treated the same as hardware text.

Hardware text is displayed using a graphics font and is the same as the style used when displaying geometry labels. When the view is rotated, hardware text will remain displayed in the XY-plane of the graphics screen.

Stroked text is actually drawn using a set of polylines and can be considered 3-dimensional text as it will rotate with the view, can be projected onto surfaces, and can be used for engraving.

Text Site:

Controls where the text is located. Imagine an invisible box enclosing all text associated with the annotation. This toggle field determines where on this box will be attached to the origin location of the annotation. The following diagram displays the available positions.

**Text Path:**

Defines the text path of the annotation. The path direction can be Right, Left Up, or Down.

Horizontal Alignment:

Specifies the horizontal alignment of the annotation and can be Normal, Left, Right, or Center. Normal selects the proper alignment based on the text path.

The horizontal alignment is dependant on the attach point.

Vertical Alignment:

Specifies the vertical alignment of the annotation and can be Normal, Top, Half, Base, or Bottom. Normal selects the proper alignment based on the text path. Base offsets the vertical text position by the size of the descending portion of the font. The vertical alignment is dependant on the attach point.

Character Expansion:

The character expansion is a ratio of the character width to the character height. A value of 1.0 defines a width that is directly proportional to the character height.

Character Spacing:

Defines an extra space between characters to add. For historical fonts (ansi, duplex, italic, etc.) this value is typically .02. For the newer fonts (romans, scriptc, timesi, etc.) this value is typically 0.

Character Density:

Specifies the character density (line weight) of the annotation.

Sub-Superscript Size:

The subscript/superscript size is specified as a percentage of the character height of the annotation and is usually smaller (less than 1.) than the character height.

Line Spacing:

Defines the vertical spacing between lines of the annotation text.

OK:

Click this button to accept the entries and output the DRAFT/ANOTE command and close the Annotation Attribute form.

CANCEL:

Click this button to cancel all the changes and close the Annotation Attribute form.

HELP:

Click this button to display a brief description of the Annotation Attribute form

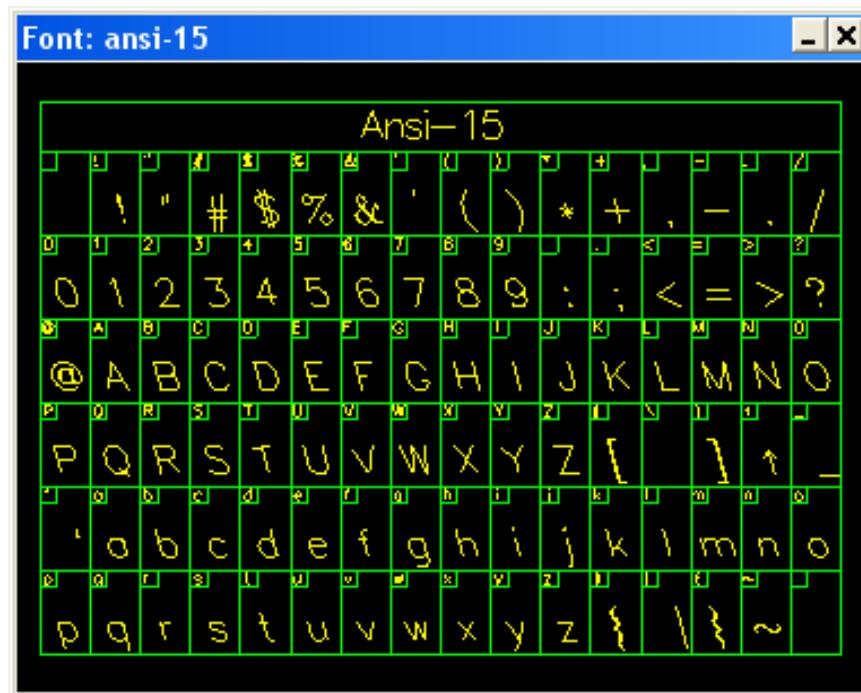
3.17 Miscellaneous.**3.17.1 Fonts Support By Annotation**

The following fonts as shown in following pages are supported by Annotations.

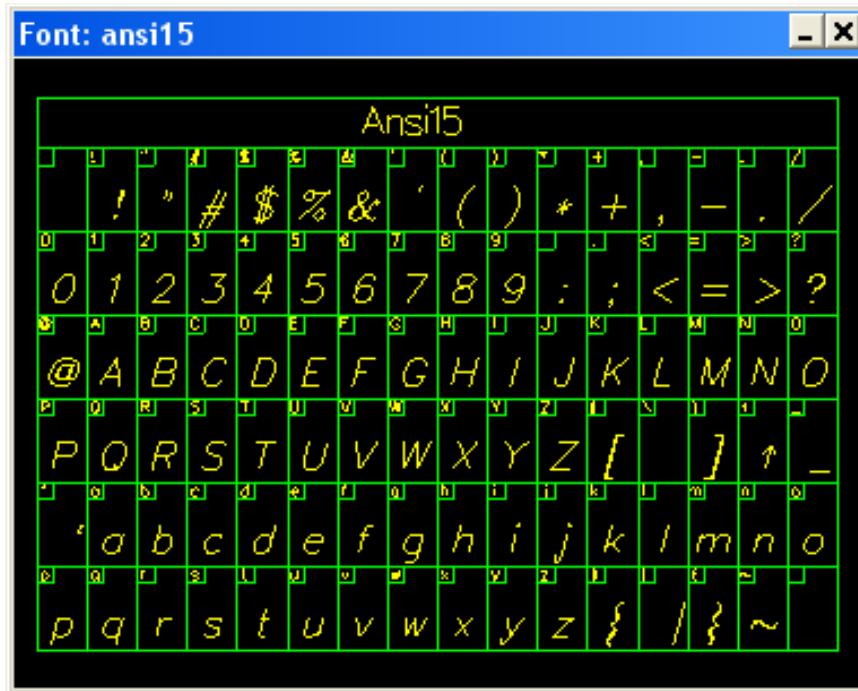
- Ansi, Iso, DraftingI



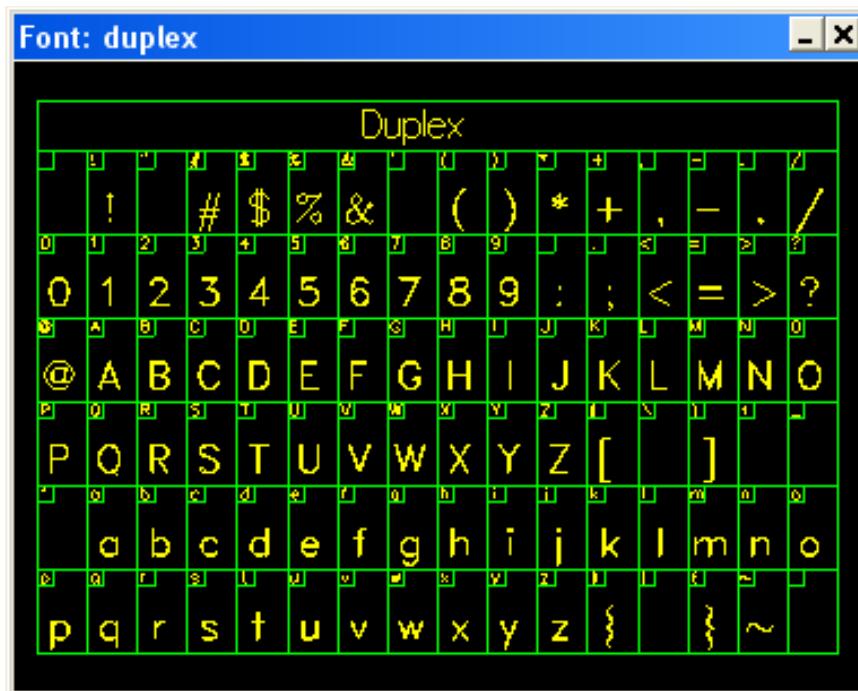
- Ansi-15, Iso-15, Ansileft



- Ansi15, Iso15, Ansiright



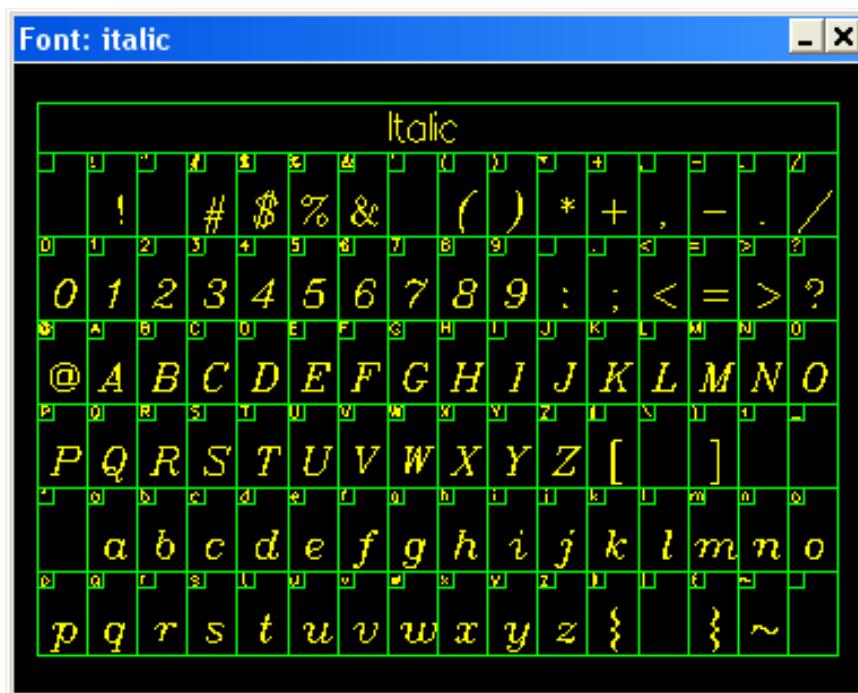
- Duplex



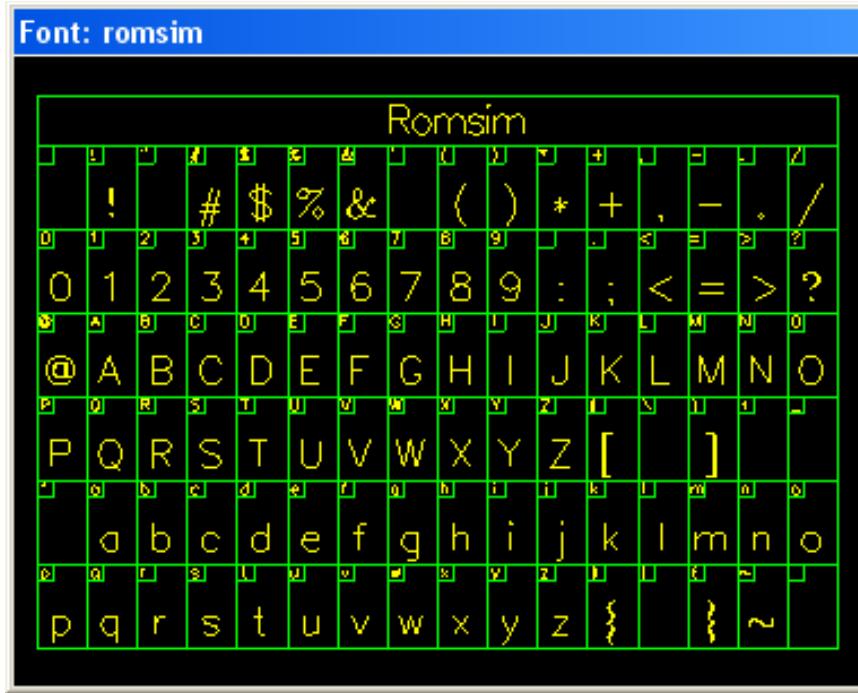
- Grksim



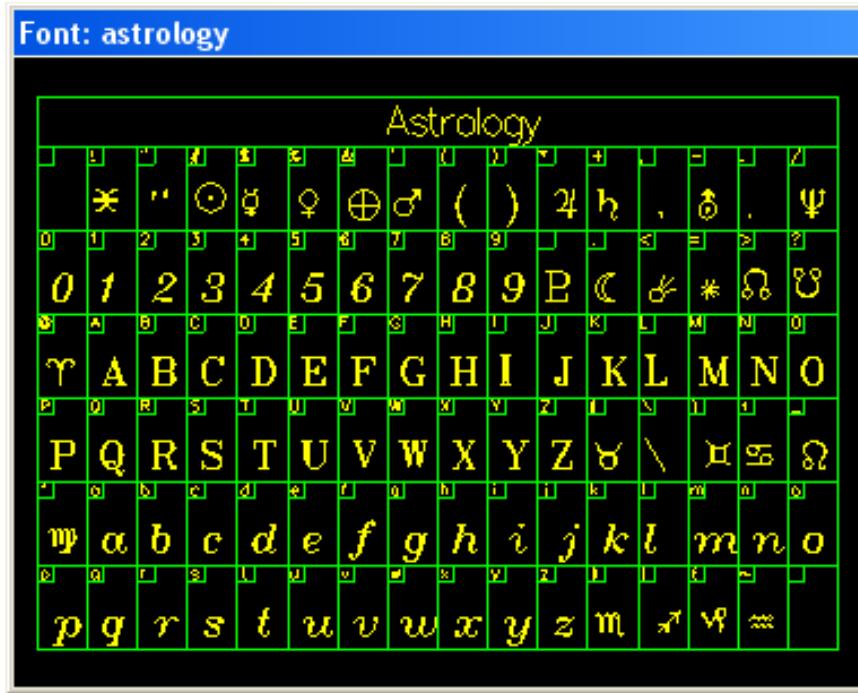
- Italic



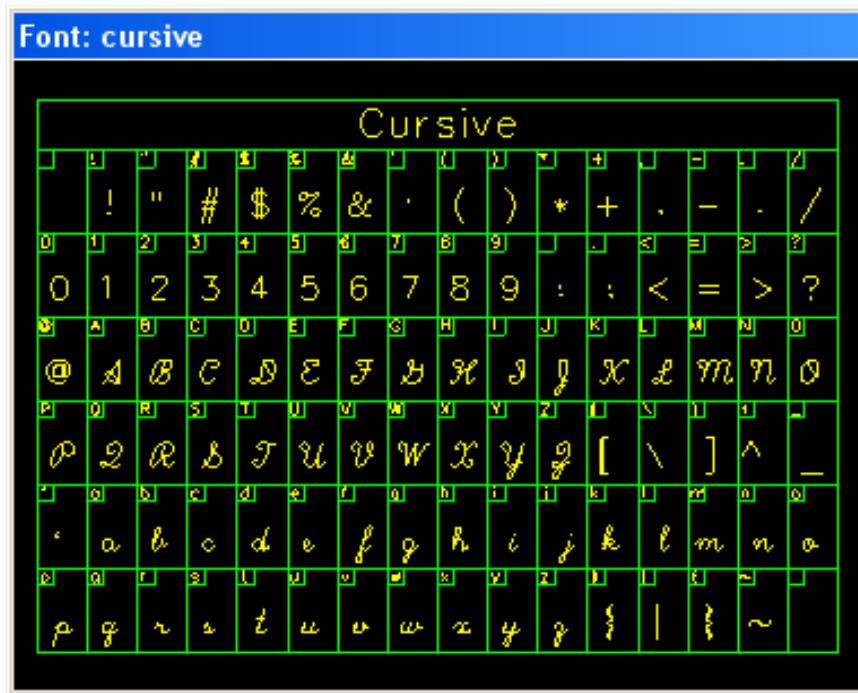
- Romsim



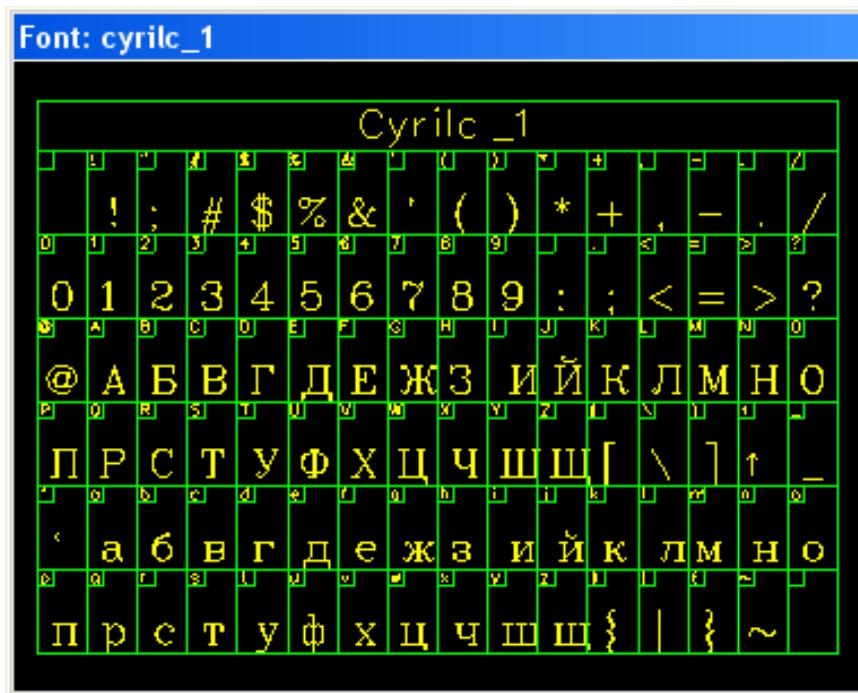
- Astrology



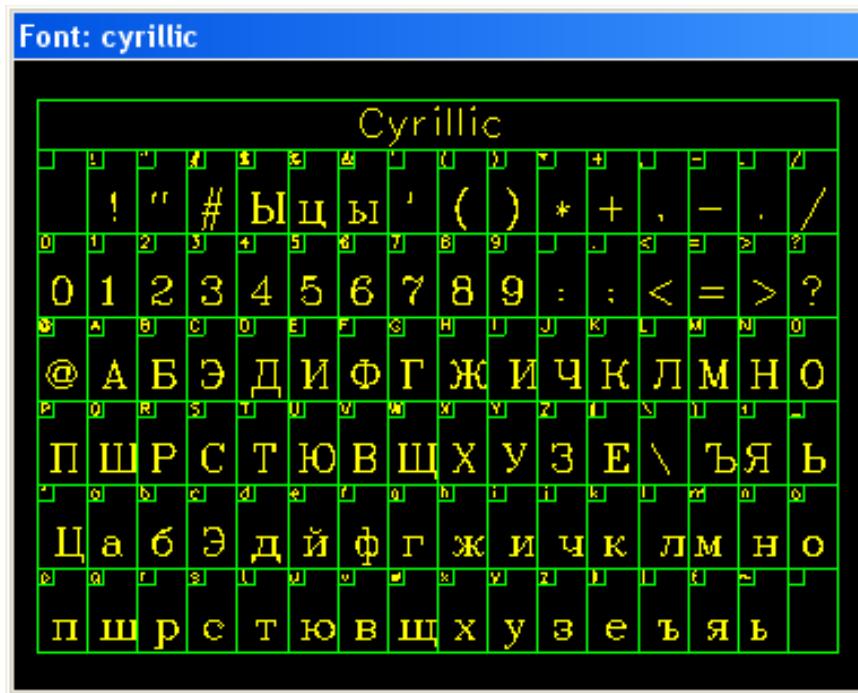
- Cursive



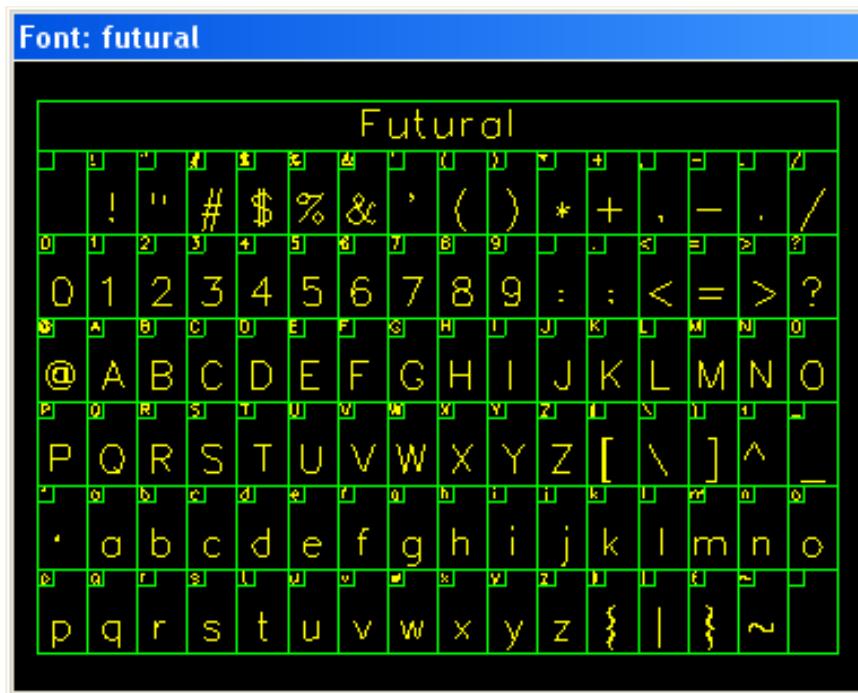
- Cyrilc_1



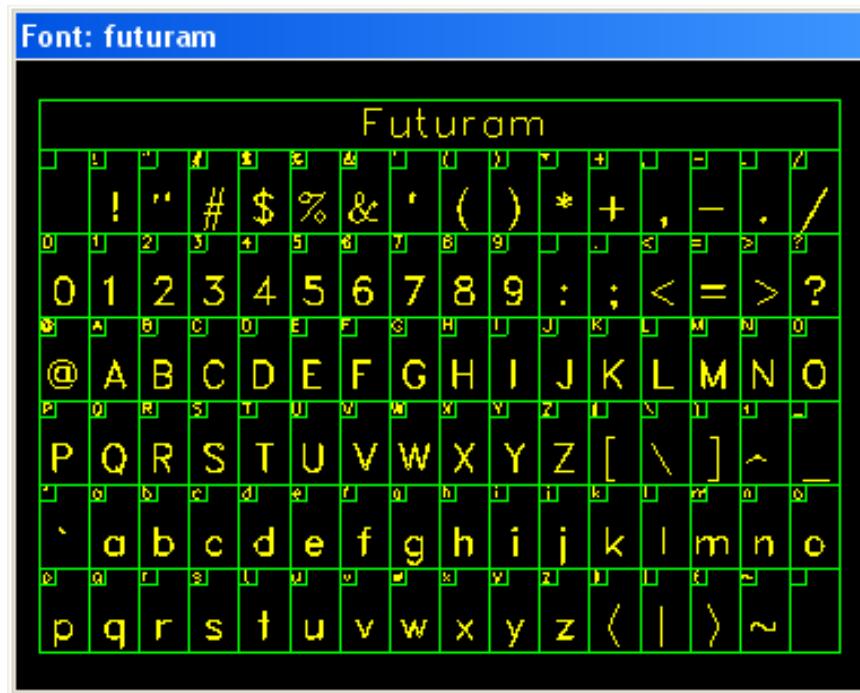
- Cyrillic



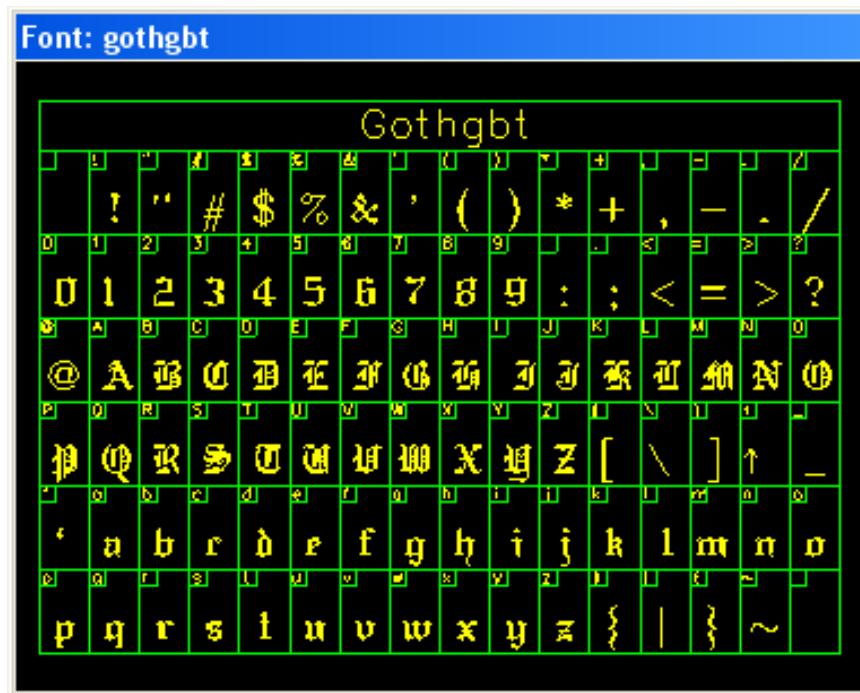
- Futural



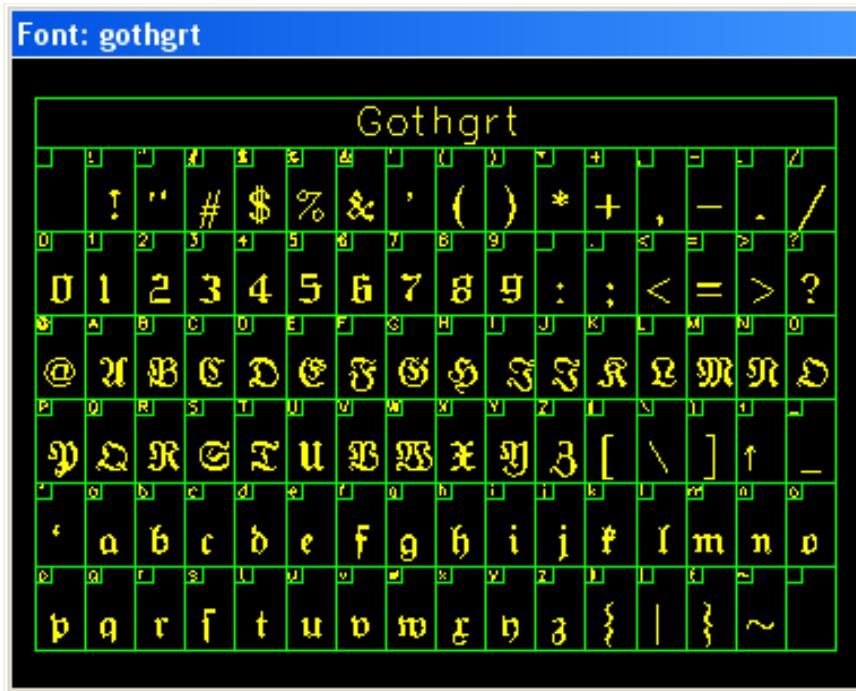
- Futuram



- Gothgbt



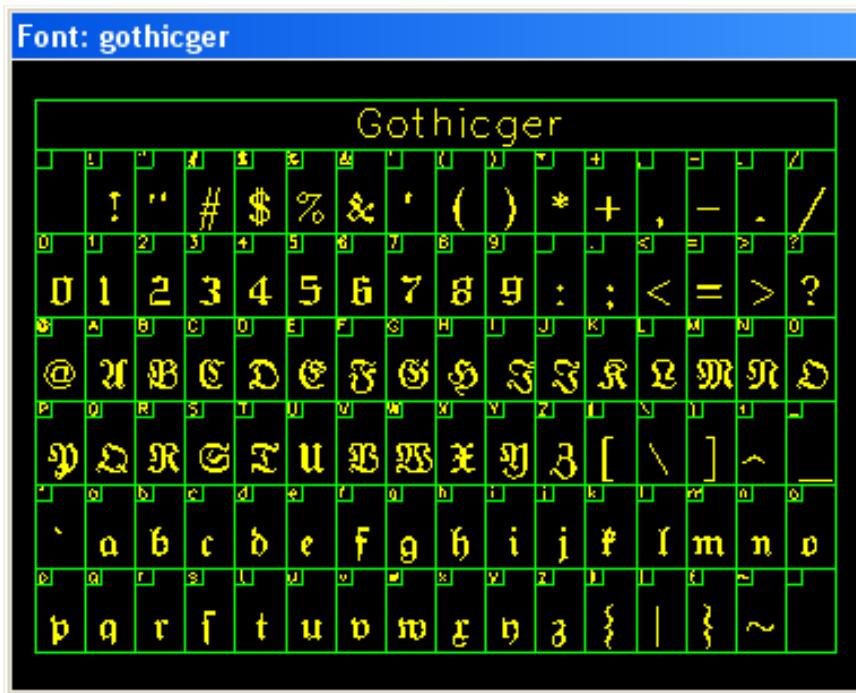
- Gothgrt



- Gothiceng



- Gothicger



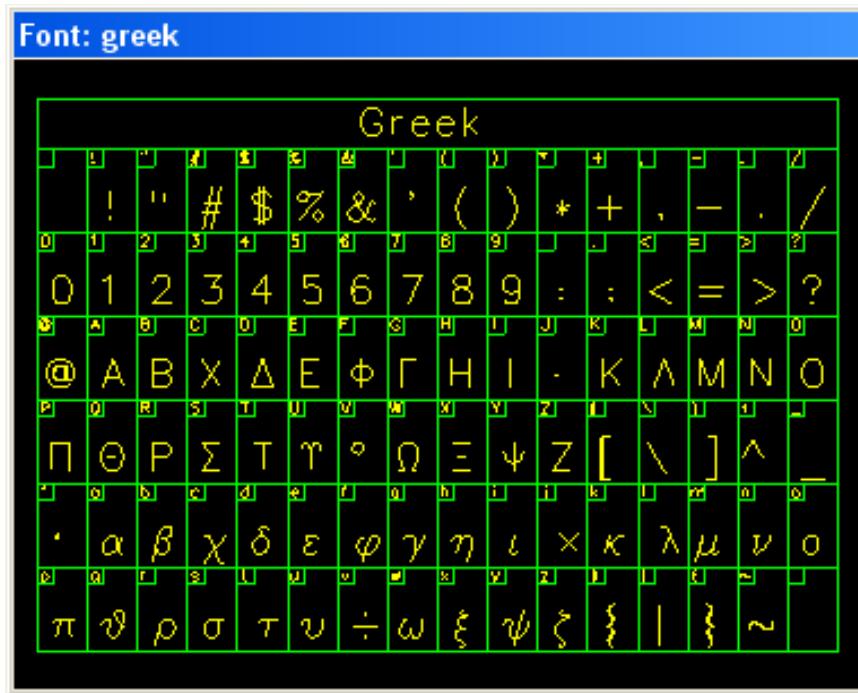
- Gothicita



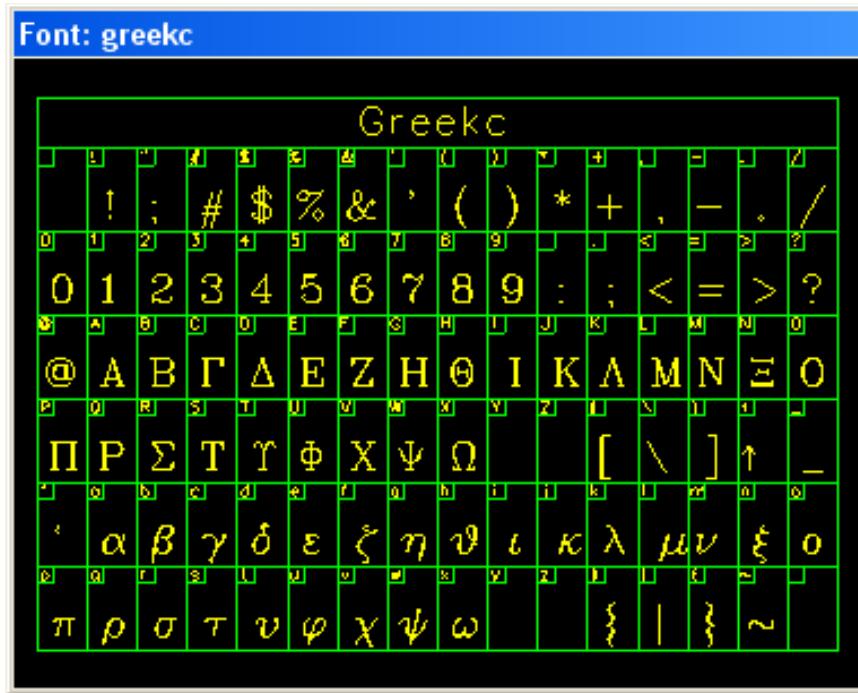
- Gothitt



- Greek



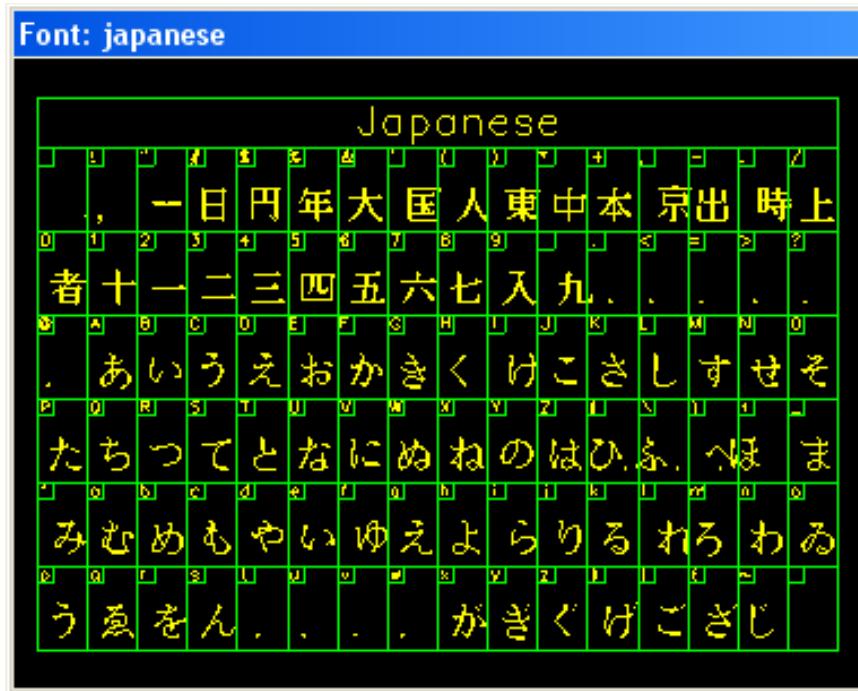
- Greekc



- Greeks



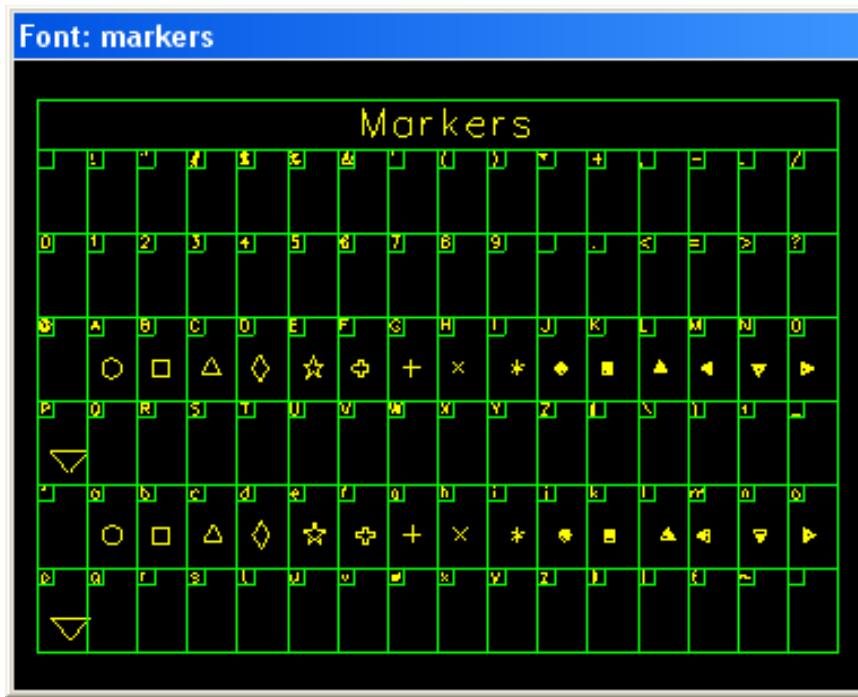
- Japanese



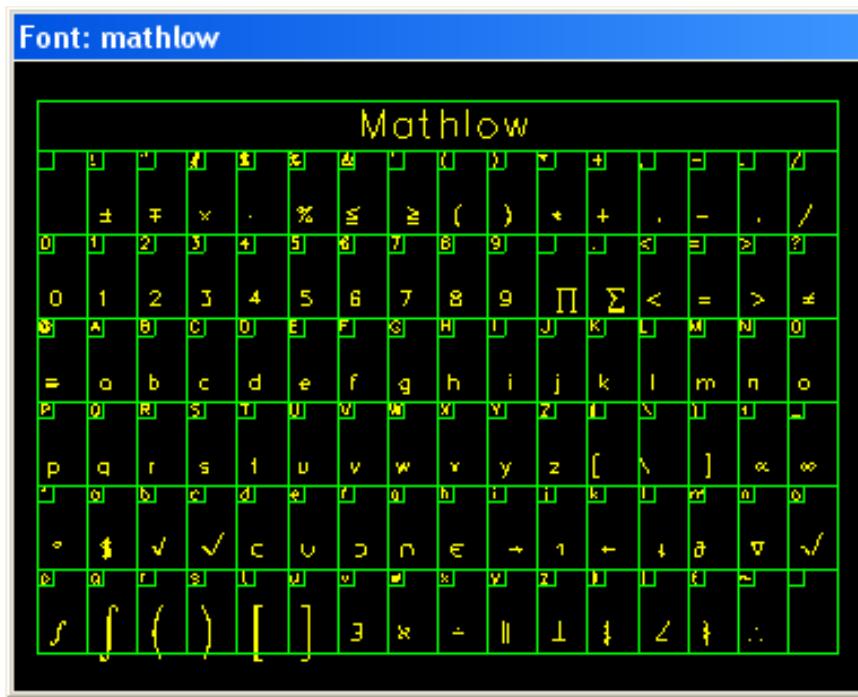
- Japanese2



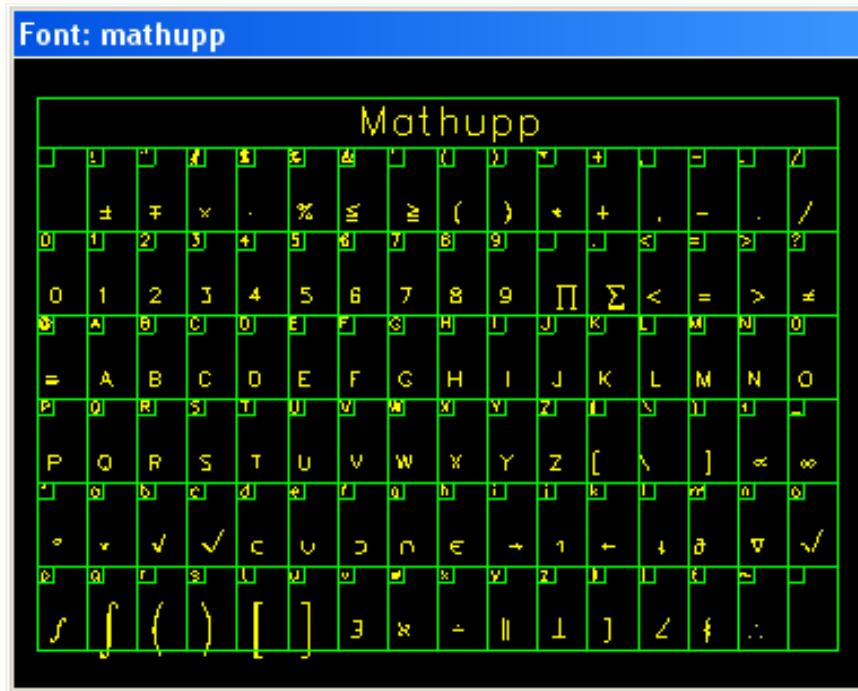
- Markers



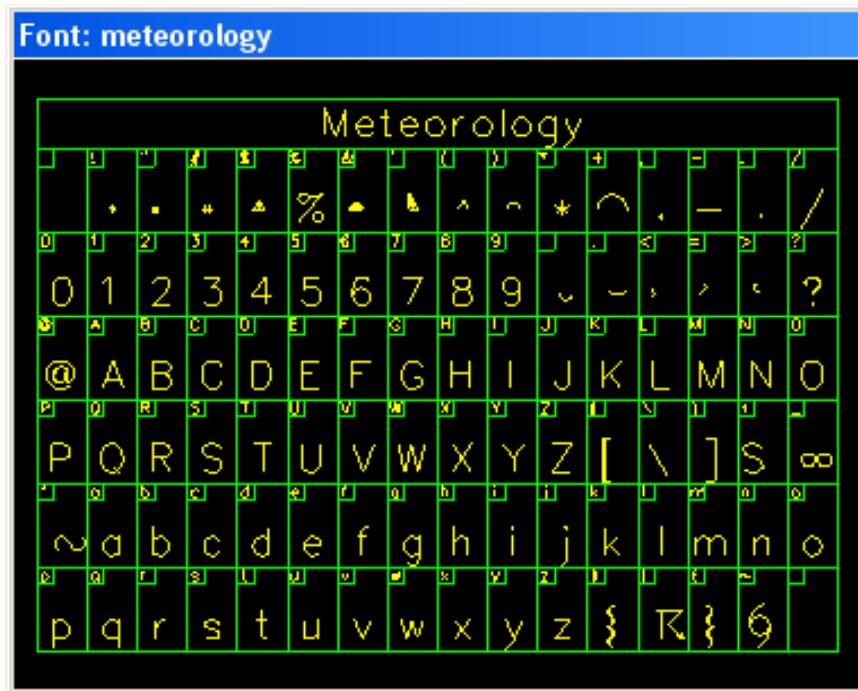
- Mathlow



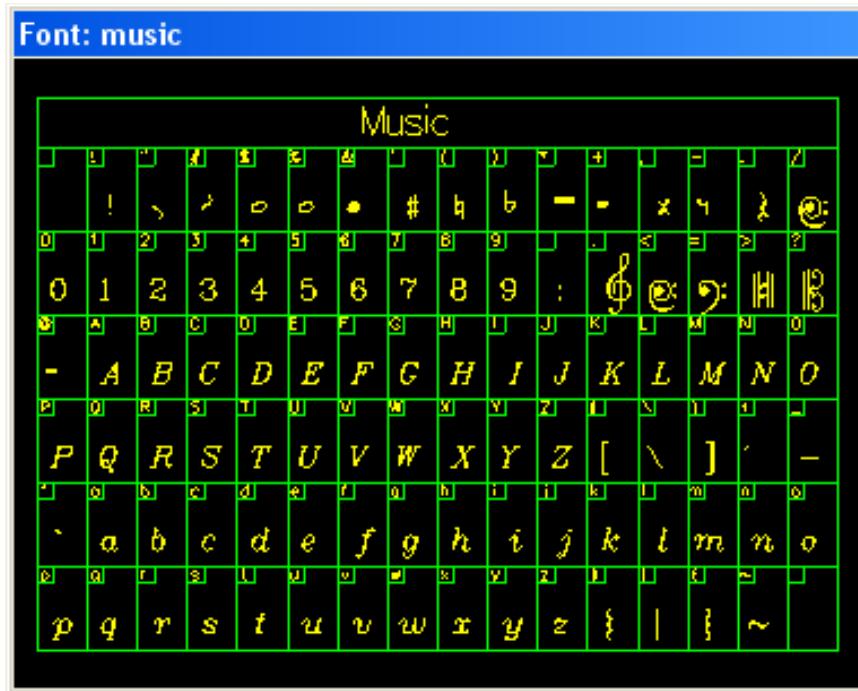
- Mathupp



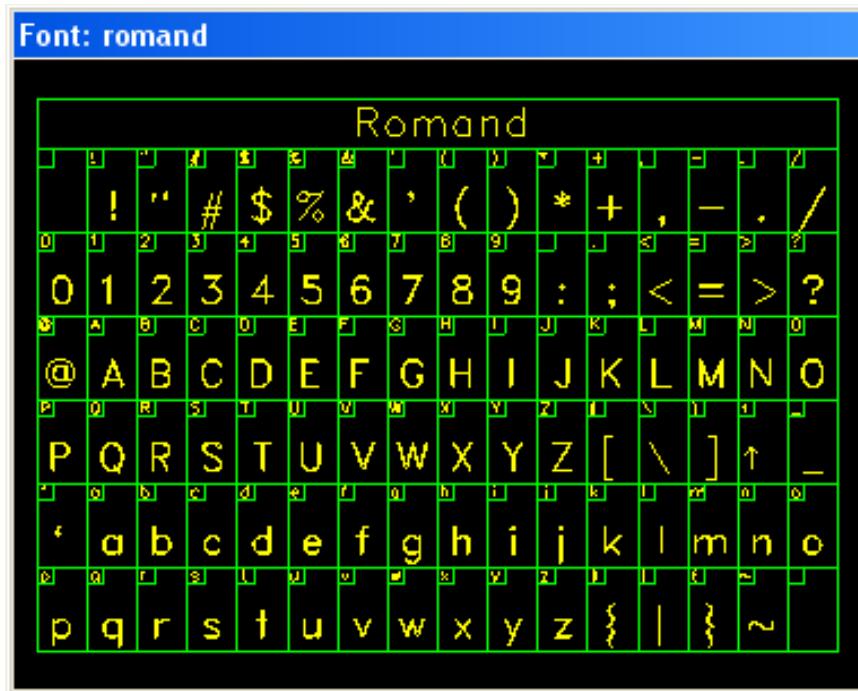
- Meteorology



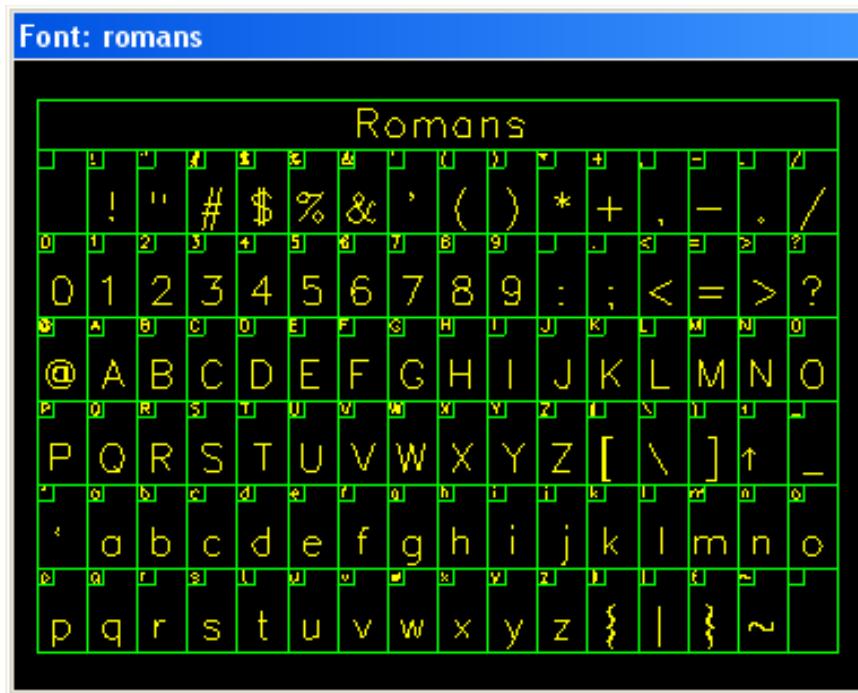
- Music



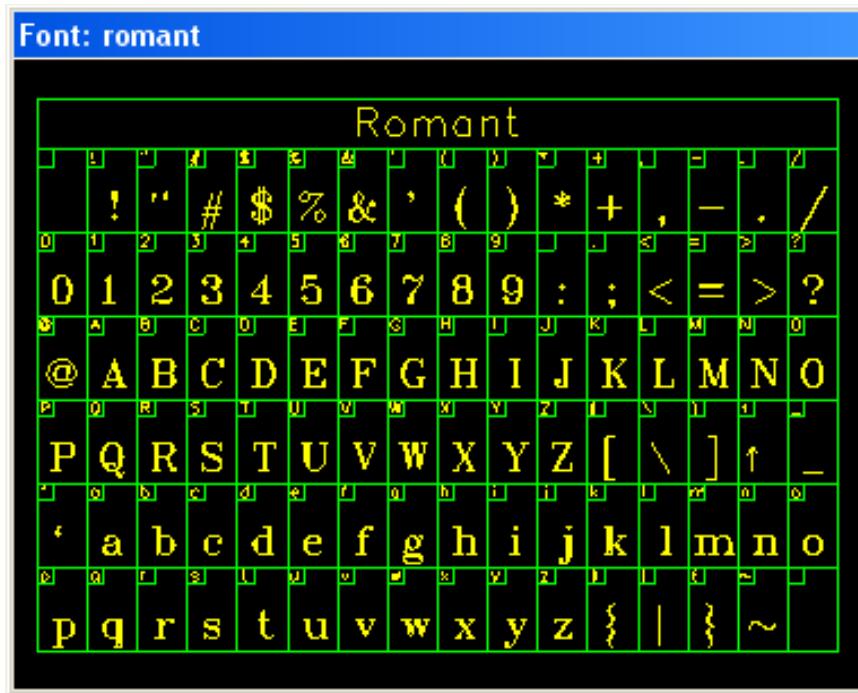
- Romand



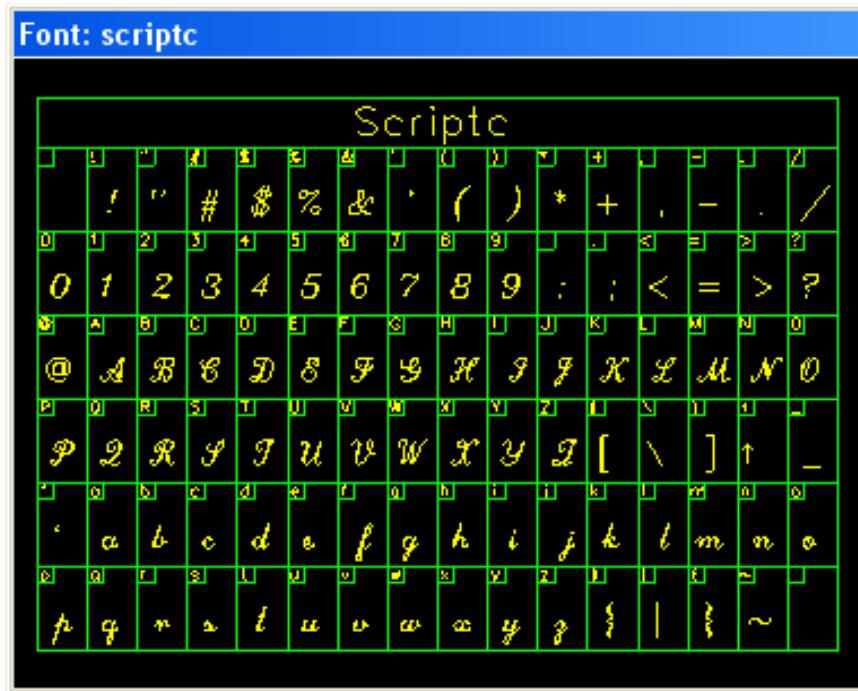
- Rromans



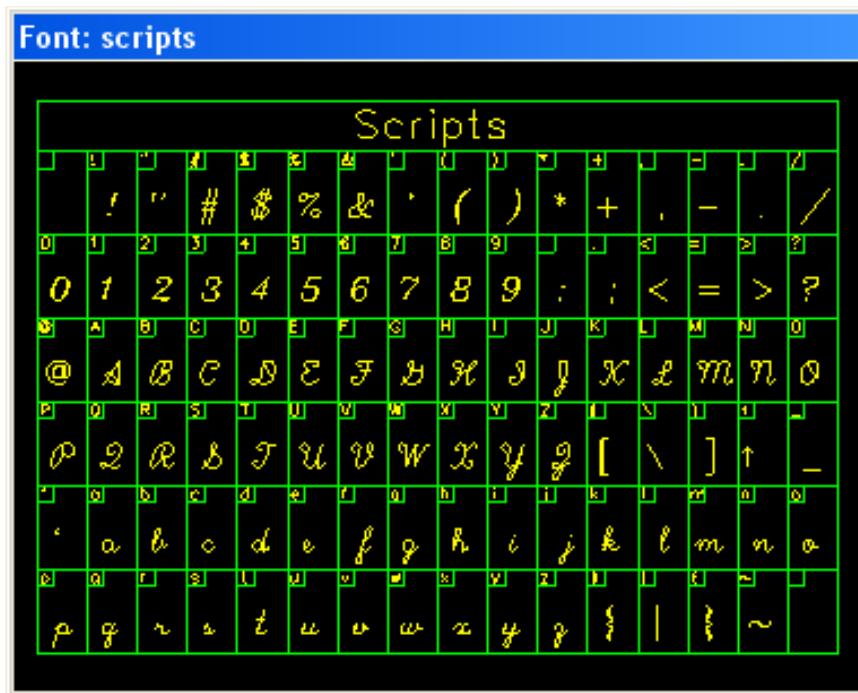
- Romant



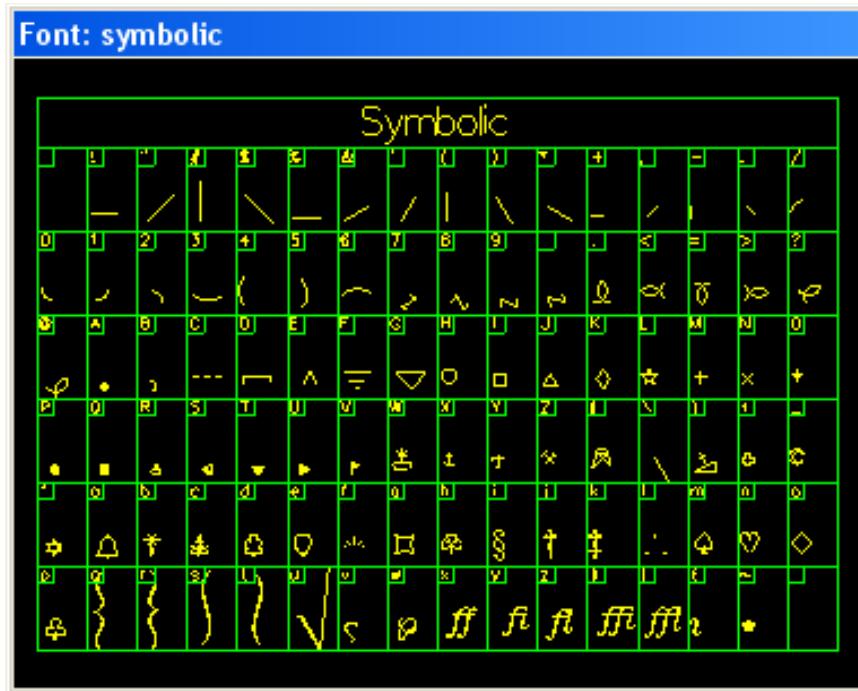
- Scriptc



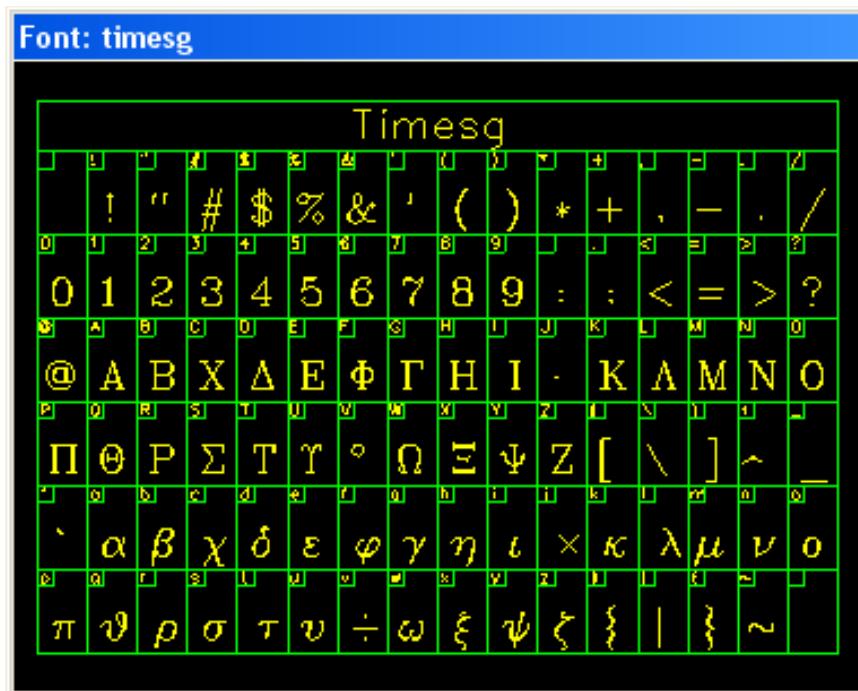
- Scripts



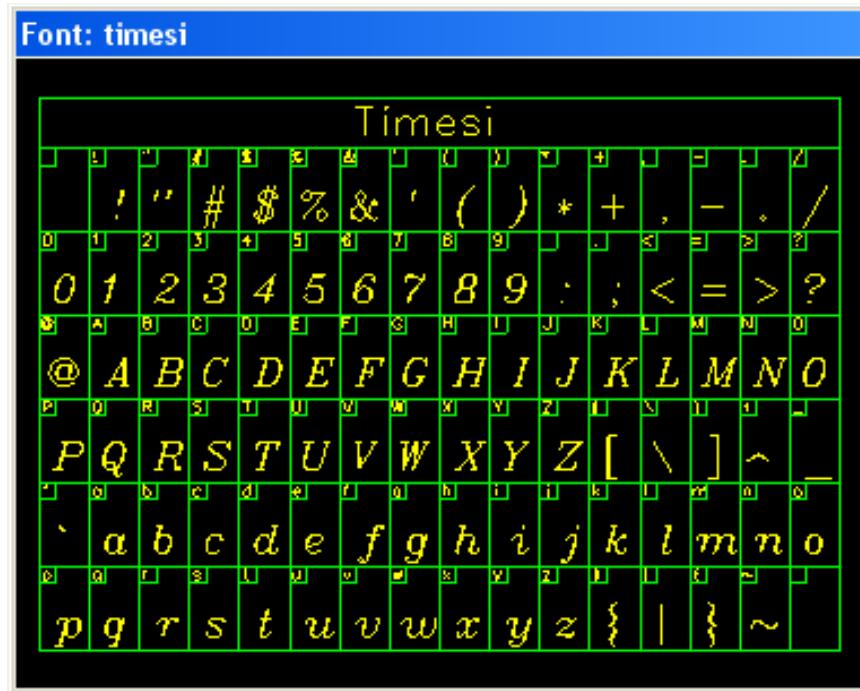
- Symbolic



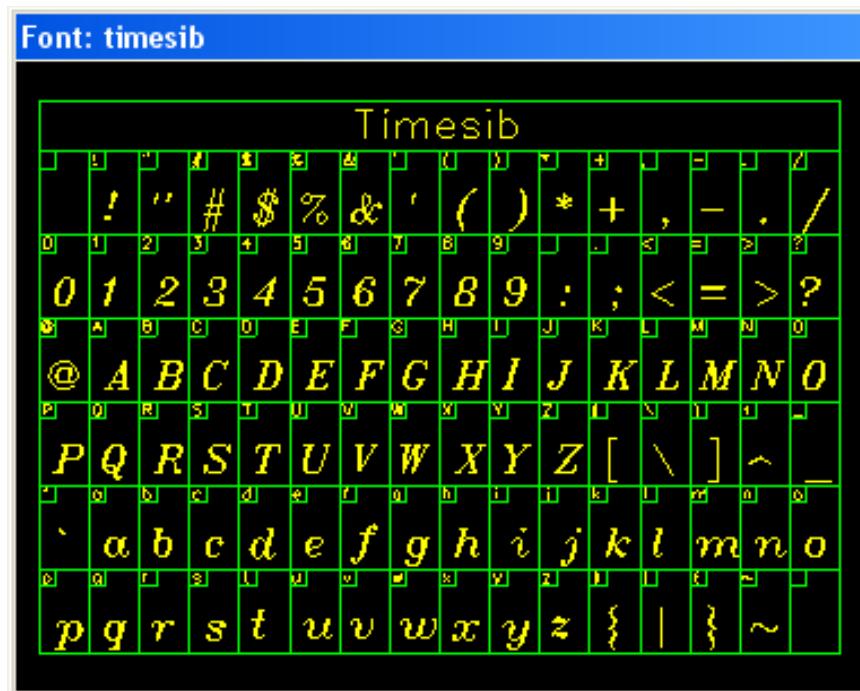
- Timesq



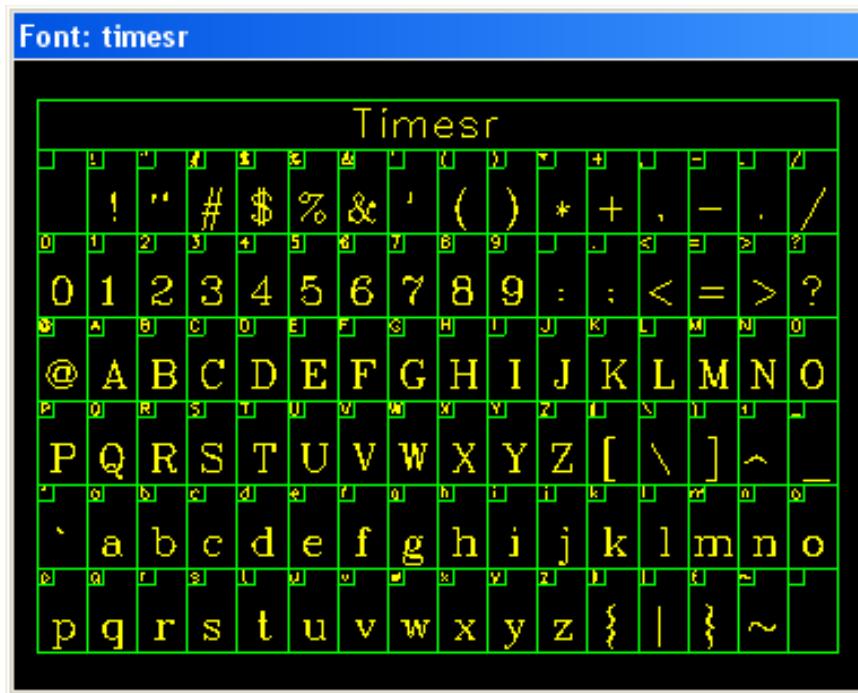
- Timesi



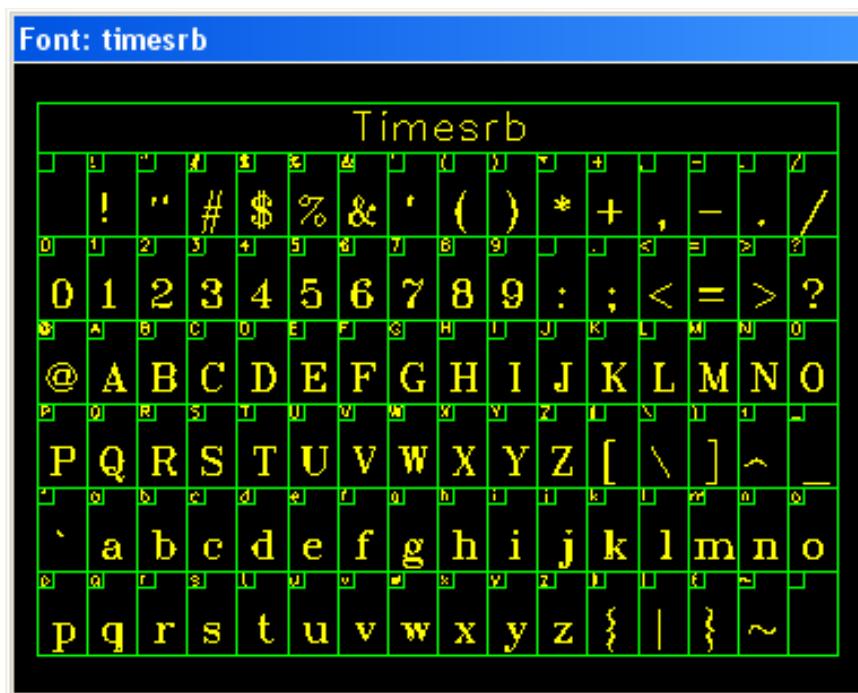
- Timesib



- Timesr



- Timesrb



3.17.2 Obtain Annotation Parameters

Certain parameters from an annotation can be obtained by the OBTAIN command. The command has the following syntax.

```
OBTAIN/anote,nlines,type,x1,y1,z1,x2,y2,z2,x3,y3,z3
```

Where:

anote	The label of the annotation to obtain the parameters
nlines	Variable to store the number of lines in the annotation
type	Variable to store the annotation. 1 Denotes a standard annotation 2 Denotes annotation along an Arc 3 Denotes annotation projected onto a surface
x1, y1, z1	Variables to store the origin of the annotation
x2, y2, z2	Variables to store the location for the character immediately following the annotation text
x3, y3, z3	Variables to store the defined location for the character that will start the next line following the annotation text

4 GEOMETRY CONTROL STATEMENTS

4.1 ANALYZ/ sf1 [[, THRU], sfn] [...] [, sfm [[, THRU], sfmn]]

This statement can be used to analyze a series or a range of surfaces and assign analytical properties to them. Surfaces created by **NCL** V9.2 or **NCL/IGES** V9.2 or higher will automatically be assigned analytical properties. Thus, the ANALYZE command is used to assign analytical properties to surfaces created in earlier versions of **NCL** or **NCL/IGES**.

If the system determines that an input surface is of a particular primitive type then the appropriate attributes will be assigned to that surface. During an interactive session, a "Status Window" will be automatically opened and the results of the analysis will be shown for each input surface.

Setting the [version flag](#) to a version earlier than 9.2 will cause **NCL** not to use or assign analytical properties to primitive surfaces.

4.2 CANON

This feature prevents a geometry identifier from arbitrarily being assigned a new value. The default is CANON/ OFF.

CANON/ ON allows identifiers to be redefined.

CANON/ OFF prevents an identifier from being redefined. If an attempt is made to assign a previously assigned identifier, an error message is displayed.

*SET/CANON and *RESET/CANON will also turn CANON on and off respectively.

Note: Normally CANON/ON allows identifiers to be redefined with exactly the same entity type, i.e. once an identifier was assigned with a geometry type, such as LINE, this identifier can only be redefined to be another LINE entity. It cannot be reassigned to another entity type, such as CIRCLE, without first deleting this entity with the REMOVE command. This is not the case for multi-entities geometry creation commands, such as:

CURVE/OUT, composite-curve, component-list

CURVE/INTOF,ALL,sf-list,AT,zlev ,.....
 plane

4 GEOMETRY CONTROL STATEMENTS

With CANON/ON, the newly created entities from these commands will directly replace the original entities disregard the original entity types if the same identifiers are used for the newly created entities from these multi-entities geometry creation commands.

4.3 Geometry Filter Control

Geometry Filter Control allows groups of geometry to be reference by their attributes such as layer, color, etc. and virtually can be used with all commands except the GET command.

4.3.1 CLIPF Function

CLIPF function filters entities based on the side of plane(s) that the geometry entities reside on. It has the following syntax:

```
CLIPF(plane,dir[,cond][,...,plane,dir[,cond]])
```

Where:

- | | |
|-------|---|
| plane | Label of plane, a maximum of 6 plane are allowed. |
| dir | Specifies which side of the plane the entities must lay on to pass the filter. Available choices are: XLARGE, XSMALL, YLARGE, YSMALL, ZLARGE, ZSMALL, SAME. “*LARGE” means entities lay on the positive side of the plane. “*SMALL” means entities lay on the negative side of the plane. “SAME” means the entire entities must be on the defined plane. |
| cond | Optional parameter specifies if any entity that only partially lies the correct side of the plane(s) should be selected. Available choices are: OFF, ON, CROSS. “OFF” is the default and means only entities entirely lie on the correct side of the plane(s) will be selected. “CROSS” means any entity that only partially lies on the correct side of the plane(s) will be selected. “ON” means only entities that are entirely on the specified plane will be selected. |

Example:

```
REMOVE/CLIPF(PL1,XLARGE,PL2,XSMALL,CROSS)
```

4 GEOMETRY CONTROL STATEMENTS

Above example would remove all entities lie entirely between the XLARGE side of the plane PL1 and the XSMALL side of the plane PL2. All entities lie partially on the XSMALL side of the plane PL2 will be removed also.

4.3.2 COLF Function

COLF function allows groups of geometry to be referenced by their color attribute. These functions will be expanded to a list of geometric entities with the matching attribute. It has the following syntax:

```
COLF(color1[,color2[...]])
```

“color#” can be an integer number representing the color (1, 2, 3, etc.) or a minor word representing the color (WHITE, RED, etc.). For example, the following command would remove all the geometry entities with the ORANGE color.

```
REMOVE/COLF(ORANGE)
```

4.3.3 FILTER Function

FILTER function allows multiple geometry types and multiple filtering types to be specified when referencing geometry in a command line. It has the following syntax:

```
FILTER(attribute_1, attribute_2, ...)
```

This function filters geometry based on the input attributes. The input attributes can be geometry types (POINT, LINE, SURF, etc.), colors (COLF(RED), etc.), layers (LAYF(3), etc.), labels with wildcards (PTX*, SQ*, etc.), and the CLIPF function.

Example:

The following command would invisible all points that have PTX as the start of their label, are on Layer 3, are blue in color, and lie to the left of PL1.

```
INVIS /FILTER(POINT,PTX*,LAYF(3),COLF(BLUE),CLIPF(PL1,XS))  
ERASE
```

The following command would remove all points and circles that are either blue or green in color.

4 GEOMETRY CONTROL STATEMENTS

REMOVE/FILTER(POINT,CIRCLE,COLF(BLUE),COLF(GREEN))

One or more of these attributes can be specified in the command and in any order.

4.3.4 LAYF Function

LAYFF function allows groups of geometry to be referenced by their layer attribute. These functions will be expanded to a list of geometric entities with the matching attribute. It has the following syntax:

`LAYF(layer1[, THRU, layern] [, layerm, [...]])`

“layer#” is an integer number representing the layer number (0, 1, 2, 3, etc.). For example, the following command would change the line weight of all the geometry entities in layer 3 to “Heavy”

`DRAFT/MODIFY=LAYF(3),LINWGT=HEAVY`

4.3.5 MARKF Function

MARKF function allows groups of points or point-patterns to be referenced by their marker type attribute. These functions will be expanded to a list of points with the matching attribute. It has the following syntax:

`MARKF(marker-type1[, marker-type2, ...marker-typen])`

Multiple marker types can be specified in a single MARKF statement. Valid marker types are DOT(1), PLUS(2), STAR(3), CIRCLE(4), CROSS(5), TRIAN(6), DIMOND(7), SQUARE(8), DBLCIR(9), LRGDOT(10) and CUBE(11).

The following statement would change the color of all the points displayed with square marker type to green.

`DRAFT/MODIFY=MARKF(SQUARE),COLOR=GREEN`

4.4 CLONE

Clone will copy a piece of geometry and move it to a new location by using a [MATRIX](#).

4 GEOMETRY CONTROL STATEMENTS

Example:

```
MX1=MX/TRANSL, 0, 1, 0  
LN2=CLONE/LN1, MX1
```

The above example would generate a new line identical to LN1 and would move it the amount specified by MX1. The new geometry can also be scaled, rotated or mirrored depending on the style of MATRIX used. See the [MATRIX](#) section for more information.

It should be noted that while the commands can be entered as shown, these statements are best produced by using the on screen menu sequence:

Edit > Copy

4.5 Creating New Entities From Existing Entities

NCL has the capability to create new geometric entities for all geometry types from existing ones using the following syntax:

```
new-name=geo-type/old-name
```

Example:

```
SF2=SURF/SF1
```

Note:

1. Geo-type does not allowed the entity type NSURF, such as “NSURF/SF1” is not allowed.
2. If the original geometry is of the type native **NCL** curve and the new geo-type is specified as SPLINE, the new geometry created will be a Rational B-Spline curve.
3. If the original geometry is of the type Rational B-Spline curve and the new geo-type is specified as CURVE, the new geometry created will be a native **NCL** curve.

4.6 d1 = DATA / element [delimiter element [. . .]]

The DATA statement element may be a scalar, vocabulary word, geometric entity or another DATA statement variable. The “delimiter” may be any valid **NCL** delimiter such as a comma, slash, equal sign, etc.

4 GEOMETRY CONTROL STATEMENTS

A DATA statement variable may be used as a complete **NCL** statement or may be inserted anywhere into an **NCL** statement, except for fixed field statements such as PARTNO, **PPRINT**, and **INSERT**.

Elements of a DATA statement may be extracted using the **OBTAIN** statement. If a vocabulary word element is extracted, the scalar variable it is extracted into becomes a synonym for that vocabulary word.

The **CAN** function may also be used to extract scalar elements of a DATA statement.

DATA is treated as part of the unibase that it can be stored in a unibase or retrieved from a unibase.

Individual scalar element of a DATA statement can also be extracted by using the following function:

name [n]

where:

name - label of the DATA statement

n - a number representing which element in the DATA statement to be extracted, it must be enclosed by a pair of square brackets.

Example:

```
d1=DATA/3,5,6,9,CCLW  
a1=d1[3] % a1 will have a value of 6
```

DATA statement examples:

```
d1=DATA/iv/vpx  
d2=DATA/GOFWD/LN1,TO,LN2  
d3=DATA/"GRIP LENGTH",10  
d4=DATA/100  
d5=DATA/ipm
```

The following example illustrates how DATA statements may be used in a tool change macro:

```
TOOLS(1)=DATA/1,7.5,1,.12,5,2500,CLW  
TOOLS(2)=DATA/2,8.5,2,.06,3,1500,CCLW
```

4 GEOMETRY CONTROL STATEMENTS

```
$$  
TCH=MACRO/TNUM1  
OBTAIN/TOOLS(TNUM1),TN1,L1,DIA,COR,HGT,RPM1,DIR  
LOADTL/TN1,LENGTH,L1  
SPINDL/RPM1,DIR  
CUTTER/DIA,COR,HGT  
TERMAC  
$$  
CALL/TCH,TNUM1=2
```

4.7 DATA Statement That Reads From A Comma Or Tab Delimited File

NCL has the ability to create DATA statements by reading a comma or tab delimited file to define its parameters. The syntax is:

```
label = DATA/file [,num-def[,start[,end]]] $  
[ ,ERROR ] [,VOCABF,ON ]  
vocab-word OFF  
SPACE  
SCALAR[,value]
```

where:

- label - Name of the first DATA statement and is mandatory. The DATA statements generated will be named as follows based on the label.

```
label = DATA/"file"
```

If the label is non-subscripted ending with a letter, then the generated DATA statement will be labelled using subscripts, with the first DATA statement being named label(1) and the subsequent DATA statements would be label(2), label(3), etc.

```
label10 = DATA/"file"
```

If the label is non-subscripted ending with a number, then the DATA statements will be labelled using the label prefix and an incremental numeric suffix, with the first DATA statement being named label10 and the

4 GEOMETRY CONTROL STATEMENTS

subsequent DATA statements would be label11, label12, etc.

```
label(5) = DATA/"file"
```

If the label is subscripted, then the DATA statements will also be subscripted, beginning with the specified subscript. The first DATA statement will be named label(5) with the subsequent DATA statements would be label(6), label(7), etc.

- | | |
|---------|---|
| file | - Name of the file to read from and can be a quote string or a text variable name. |
| num-def | - Specifies a scalar variable that will receive the number of data definitions defined (number of lines read in from the file). |
| start | - Specifies the beginning line number to start processing from. All lines in the file prior to the starting line will be ignored. |
| end | - Specifies the ending line in the file that will be processed. |

end - start + 1 data definitions will be defined. If the end of file is reached prior to processing the ending line, then a lesser amount of lines will be processed. A value of 0 specified for *end* will process to the end of the file.

The entire file will be processed if *start* and *end* are not specified.

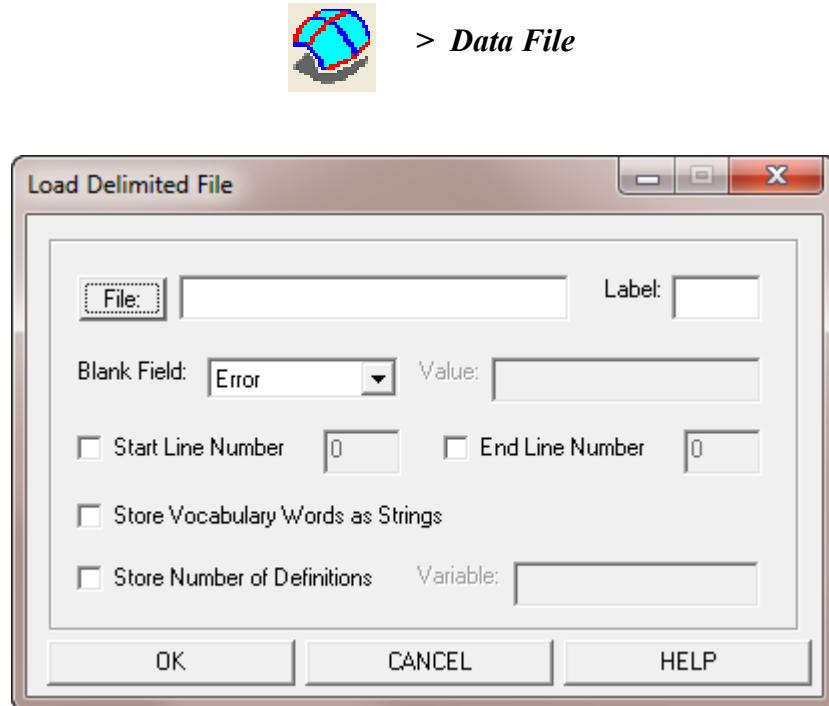
- | | |
|------------|---|
| ERROR | - Specifies that an error will be output when a blank field is processed from the file.

A blank field is defined as two consecutive commas without any value between them. |
| vocab-word | - Specifies a vocabulary word (SPACE, BLACK, etc.) that will be used to fill in the blank fields, i.e. all blank fields processed from the file will contain this vocabulary word in the data definition. |
| SPACE | - Specifies that a blank text string will be stored in the blank fields. |

4 GEOMETRY CONTROL STATEMENTS

- SCALAR - Specifies that the number specified by value will be stored in the blank fields.
- VOCABF - Specifies how vocabulary words will be read from the file.
- ON - Specifies that all vocabulary words will be stored as vocabulary words, the same as they are stored when processing a standard DATA statement.
- OFF - Specifies that the vocabulary words will be stored as text strings.

It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence to open the DATA file form as shown below.



4 GEOMETRY CONTROL STATEMENTS

This command is used to decompose symbol instance into its components. Each component of the symbol instance will be defined as a separate geometry entity and the symbol instance will be deleted.

Where:

label

- Name of the first entity and is mandatory. The entities extracted from the instance will be named as follows based on the label.

label = DECOMP/instance

If the label is non-subscripted ending with a letter, then the extracted entities will be labelled using subscripts, with the first entity being named label(1) and the subsequent entities receiving the following subscripts (label(2), label(3), etc.).

label110 = DECOMP/instance

If the label is non-subscripted ending with a number, then the extracted entities will be labelled using the label prefix and an incremental numeric suffix, with the first entity being named label10 and the subsequent entities receiving the following numeric suffices (label11, label12, etc.).

label(5) = DECOMP/instance

If the label is subscripted, then the extracted entities will also be subscripted, beginning with the specified subscript. The first entity will be named `label(5)` with the subsequent entities receiving the following subscripts (`label(6)`, `label(7)`, etc.).

ALL

- Specifies that all symbol instances will be decomposed.

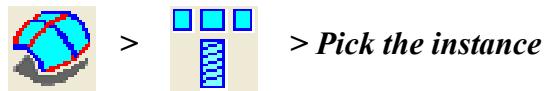
instance1 [...]

- Name list of the instances to be decomposed separated by a comma

4 GEOMETRY CONTROL STATEMENTS

- | | |
|-------|--|
| nent | - An optional scalar that will receive the number of entities extracted from the instance. |
| RESET | - Specifies that the current total number of extracted entities will be reset before the scalar value is set so that the scalar only contains the number of extracted entities from the current command. This is the default setting. |
| PLUS | - Specifies that the number of entities decomposed in this command will be added to the value already stored in the scalar. This syntax is useful when the multiple symbols are selected from the interface to decompose and they do not fit entirely in a single command. |

It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence



4.9

DEFNAM/ geo_type1, name1 [, INDEX] [, geo_type2, name2, ...]

This command allows you to specify the desired naming preference when using the auto name feature or when using commands that generate multiple entities, such as the multiple fillet command.

Where:

- | | |
|-----------|---|
| geo_type1 | - The word specifies the geometry type that the default name applies to (POINT , LINE , CIRCLE , etc.). |
| name1 | - The new default name applies to the specified geometry type. A maximum of 20 alpha characters is allowed for name1. |

If the word INDEX follows the name1 specifier then the auto name generation will create subscripted names for new entities of this type instead of a prefix-number combination. A maximum of 20 alphanumeric characters with at least one alpha character is allowed for name1.

4 GEOMETRY CONTROL STATEMENTS

Example:

```
DEFNAM/POINT, SP1, INDEX, LINE, EDGELINE
```

This statement would generate the following names on subsequent point and line definitions:

```
SP1(1)=POINT/...
SP1(2)=POINT/...
EDGELINE1=LINE/...
EDGELINE2=LINE/...
```

4.10 GENPTS

The GENPTS statement notifies **NCL** to start or end saving the points and vectors generated by **NCL** motion. The valid syntax constructs for the GENPTS statements are:

4.10.1	GENPTS/ [POINT,] [TE ,]	\$
	ARC	
	DS	
	PS	
	 num-points-scalar, reserved-point-array	\$
	 [, reser-vector-1-array [, reser-vector-2-array]]	\$
	 [, NOW]	
	NEXT	

This statement will cause the number of X, Y, Z-coordinate points generated during subsequent **NCL** motion statements to be assigned to the scalar specified by num-points-scalar and the points themselves to be stored in the elements of the reserved-point-array.

The minor word POINT is optional.

The optional parameters TE, ARC, DS and PS specify what points are generated and what should be stored in the optional reser-vector-2-array.

4 GEOMETRY CONTROL STATEMENTS

“TE” specifies the tool end points of the motion will be stored in the reserved-point-array. This is the default condition. If the optional reser-vector-2-array is given, the tool axis at each tool end points of the motion will be stored in its elements.

“ARC” behaves the same as “TE” except if the driving surface is a circle or a surface of the type “CYLINDRICAL” with the tool axis parallel to the axis of the circle/“CYLINDRICAL surface” and normal to the part surface which is of the type “PLANE” or “PLANAR SURFACE”, only the tool end point of the motion will be stored, the intermediate tool end points generated along the circular motion will not be stored. This is similar to specify the [SET/ MODE,CIRCUL](#) command in previous version. This parameter reverts to “TE” if [SET/ MODE,LINEAR](#) is in effect.

“DS” specifies that the drive surface contact points of the motion will be stored in the reserved-point-array and the drive surface normal vectors will be stored in the reserv-vector-2-array if given.

“PS” specifies that the part surface contact points of the motion will be stored in the reserved-point-array and the part surface normal vectors will be stored in the reserv-vector-2-array if given.

If the optional reser-vector-1 array is given, the forward direction vectors at each tool end points of the motion will be stored in its elements.

“NEXT” will cause *GENPTS* to wait until the next *GENPTS* command (including *GENPTS/NOMORE*) to generate the geometry, instead of generating the geometry directly after each successful motion statement. This allows for any clfile modification routines ([ARCSLP/FILLET](#), [REVERS](#), etc.) to be applied to the generated geometry. “NOW is the default.

When the *GENPTS* statement is executed, **NCL** will set the num-points scalar to zero. If the point or vector array names have not been [RESERV](#)ed, they will automatically be RESERVed for the maximum number of elements (1,000,000).

If a subscript is given for the point or vector array IDs, storing of the points or vectors will begin at that subscript. If no subscript is given, 1 is assumed.

The following items should be noted about the *GENPTS* statement:

- CANON/ ON is required to store points or vectors into any variables that have been previously defined.

4 GEOMETRY CONTROL STATEMENTS

- If tool axis modify is in effect, it will apply to the points and vectors generated.
- If **REFSYS** is in effect, it will NOT apply.
- **TRACUT** does NOT apply.
- GENPTS geometry will still be created during a **DNTCUT** sequence.
- Setting the version flag to a version earlier than 9.3 with the default “TE” condition will cause **NCL** reverts back to the old genpts logic of the previous version with the following caveats.
 - a. Points and vectors are NOT stored during **POCKET**, **FMILL**, **RMILL**, **SCRUB** or immediate motion statements (**GOTO**, **GODLTA**).
 - b. Points and vectors generated during the internal calculation of the **GO** commands will be stored also, not just the final point. This is the same for all the continuous motion commands that all the internal calculated positional information will be output, not just the final one.
 - c. **MULTAX/ ON** is required to store the tool axis vector.

It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence



4.10.2 GENPTS/ PNTVEC [, type] [,TE] , n, pv-array [, vec-array]\$
ARC
DS
PS

[, NOW]
NEXT

This command syntax can generate two types of point-vectors:

type = 1 - pv-array will contain point-vectors where the point-vector direction are the forward directions associated with the generated points. This is the default. The optional variable vec-array will contain the optional vectors specified by the optional parameters TE, ARC, DS and PS.

4 GEOMETRY CONTROL STATEMENTS

type = 2 - pv-array will contain point-vectors where the vector directions are the vectors specified by the TE, ARC, DS and PS optional parameters associated with the generated points. The optional variable vec-array will contain the forward direction vectors.

The optional parameters TE, ARC, DS and PS specify what points are generated and what should be stored in the pv-array and the optional vec-array.

“TE” specifies the tool end points of the motion will be stored as the generated points. This is the default condition. The tool axis at each tool end points of the motion will be stored in the corresponding vector array.

“ARC” behaves the same as “TE” except if the driving surface is a circle or a surface of the type “CYLINDRICAL” with the tool axis parallel to the axis of the circle/“CYLINDRICAL surface” and normal to the part surface which is of the type “PLANE” or “PLANAR SURFACE”, only the end point of the motion will be stored, the intermediate points generated along the circular motion will not be stored. This is similar to specify the [SET/ MODE,CIRCUL](#) command in previous version. This parameter reverts to “TE” if [SET/ MODE,LINEAR](#) is in effect.

“DS” specifies that the drive surface contact points of the motion will be stored as the generated point and the drive surface normal vectors will be stored in the corresponding vector array.

“PS” specifies that the part surface contact points of the motion will be stored as the generated points and the part surface normal vectors will be stored in the corresponding vector array.

“NEXT” will cause *GENPTS* to wait until the next *GENPTS* command (including *GENPTS/NOMORE0* to generate the geometry, instead of generating the geometry directly after each successful motion statement. This allows for any clfile modification routines ([ARCSLP/FILLET](#), [REVERS](#), etc.) to be applied to the generated geometry. “NOW is the default.

It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence



4 GEOMETRY CONTROL STATEMENTS

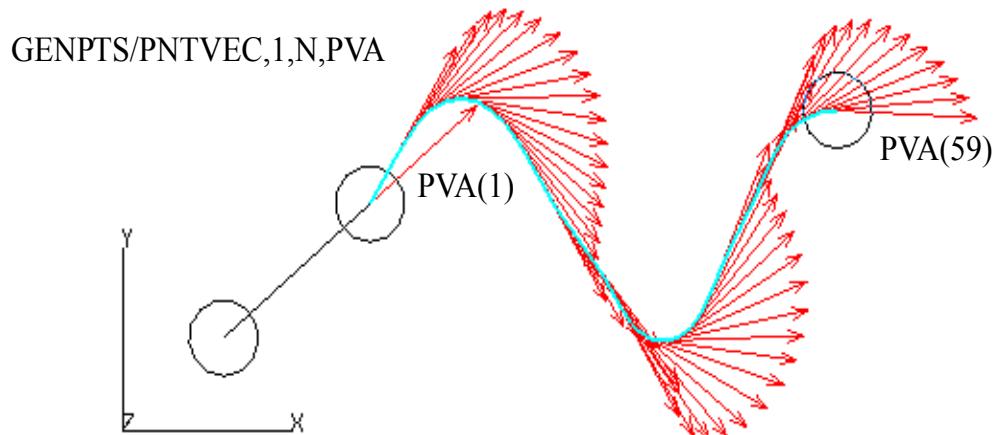
4.10.3 GENPTS/ NOMORE

This statement will cause **NCL** to stop storing the points and vectors generated during motion.

Example GENPTS Statements:

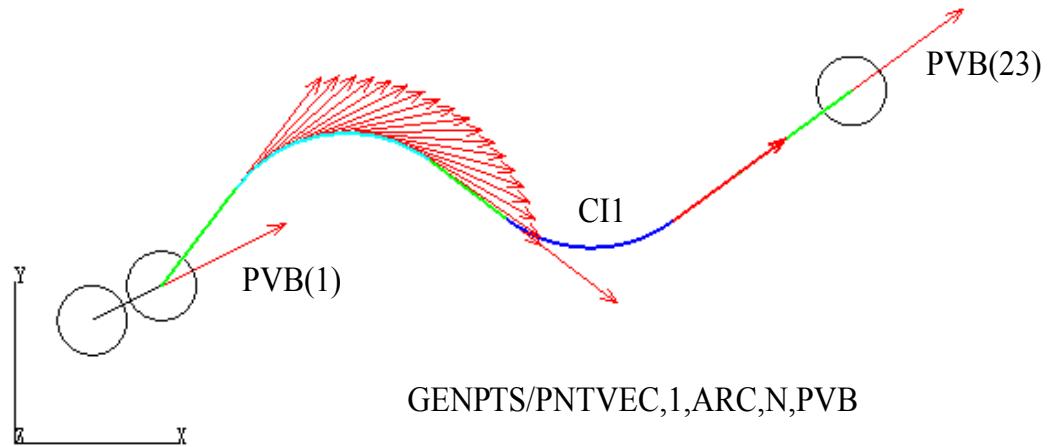
```
GENPTS/N, P  
GENPTS/N, P(6), VFWD  
GENPTS/N, P, VFWD, VTLAX(3)  
GENPTS/N, P,, VTLAX  
GENPTS/NOMORE  
GENPTS/PNTVEC, 1, N, PVA  
GENPTS/PNTVEC, 2, N, PVA, V
```

The following illustrations show the result of a tool path along a curve with GENPTS/ PNTVEC in effect:

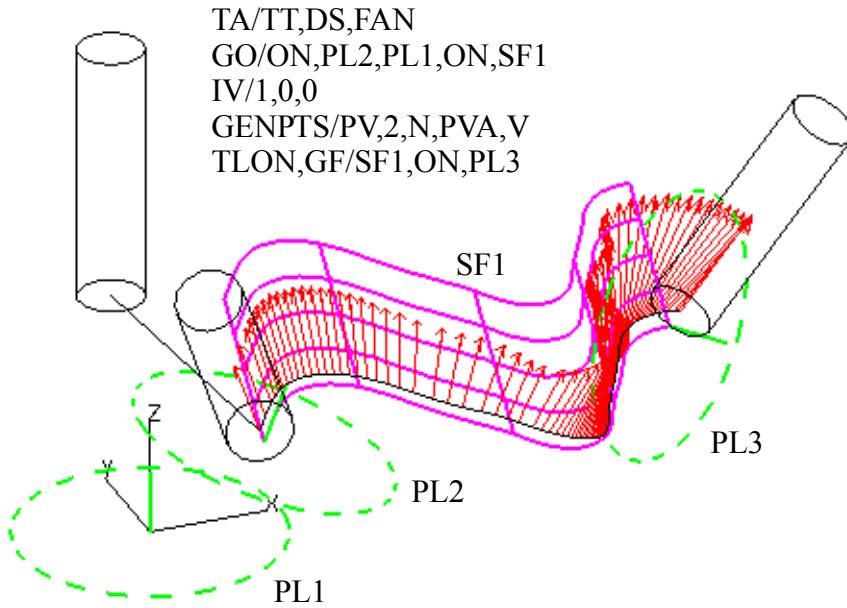


In the foregoing example, the point-vectors PVA have been generated using the type=1 pv-array.

4 GEOMETRY CONTROL STATEMENTS



The ARC option has been specified in above example. Note that there is no point-vectors generated for motion along the circle entity.



In the above 5-axis example, the point-vectors PVA have been generated using the type=2 pv-array. Note that in this example, only the resulting tool axis point vectors are shown for clarity.

4 GEOMETRY CONTROL STATEMENTS

It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence



or



4.11 **CHKPTS/ tolerance, maxdp, maxang [, MACRO, m1 [, PS]] DS**

CHKPTS/ NOMORE
CHKPTS/ REMOVE

Used in a manner similar to the [GENPTS](#) command. When a continuous path motion statement ([GOFWD](#), [GOLFT](#), etc.) is executed while a **CHKPTS** statement is in effect, the CL points and tool axis vectors generated are analyzed and only those required to satisfy the “tolerance”, “maxdp” and “maxang” values specified in the **CHKPTS** statement are output to the CL file.

The “tolerance” parameter specifies the maximum chordal tolerance. That is the maximum distance from the control surfaces ([DS](#) and [PS](#)) to the mid point of a line that passes through any two consecutive output points.

The “maxdp” parameter specifies the maximum distance between any two consecutive output points. This can be used to ensure that point are output at least every “maxdp” inches/millimeters even when driving relatively flat surfaces.

The “maxang” parameter specifies the maximum angle change (in degrees) between any two consecutive tool axis vectors.

If the optional MACRO clause is specified, the generated points are not output to the CL file but are stored in temporary variables.

When a **CHKPTS/ NOMORE** statement is encountered, the macro specified by “m1” is automatically called and the first 5 macro variables are assigned the following values:

4 GEOMETRY CONTROL STATEMENTS

- | | | |
|-------|---|---|
| Var#1 | - | The number of points generated. |
| Var#2 | - | The name of a reserved variable containing the points. |
| Var#3 | - | The name of a reserved variable containing the forward vectors. |
| Var#4 | - | The name of a reserved variable containing the tool axis vectors. |
| Var#5 | - | The name of a reserved variable containing the vectors normal to the part surface, if PS is specified, or the drive surface, DS is specified. PS is assumed if neither was specified in the MACRO clause. |

The macro specified by “m1” must have been defined using a minimum of 5 variables.

The CHKPTS/ REMOVE statement will cancel the CHKPTS statement and remove any temporary points and vectors that have been generated. This is useful when you want to stop the generation of points because of an error condition or if you change your mind and do not want the MACRO call to take place (which would occur if you were to use CHKPTS/ NOMORE). This would be similar to using *TERMAC or *RESET/ CALL when you want to stop the processing of a macro or loop

4.12

MODSYS/ matrix-id MODSYS/ NOMORE

Use to temporarily move the base coordinate system to a more convenient position and orientation in space. matrix-id is the name of a previously defined matrix that defines the desired coordinate system. Upon execution of this statement all subsequent operations (geometry definitions, tool motion, TRACUT, REFSYS, etc.) will be relative to the coordinate system defined by matrix-id. This will remain in effect until a MODSYS/ NOMORE or another MODSYS/ matrix-id statement is processed.

This differs from REFSYS and TRACUT in that REFSYS applies only to geometry definitions and TRACUT applies only to the output CL points. MODSYS applies to everything. A REFSYS may be in effect with a MODSYS but when MODSYS is changed, REFSYS is reset. Changing UNITS resets both MODSYS and REFSYS.

4 GEOMETRY CONTROL STATEMENTS

4.13 MOVE

Move allows a piece of geometry to be relocated by using a MATRIX.

Example:

```
MX1=MATRIX/TRANSL, 0, 1, 0  
MOVE/LN1, MX1
```

The above example would move LN1 by the coordinates given in MX1. The geometry can also be rotated, scaled or mirrored depending on the **MATRIX** used. See the MATRIX section for more information. MOVE actually redefines the specified geometry whereas **CLONE** leaves the original geometry alone and makes a new copy modified by the specified matrix.

It should be noted that while the commands can be entered as shown, these statements are best produced by using the following on screen menu sequence:

Edit > Move

4.14 OBTAIN

The OBTAIN statement allows the canonical data/attributes of geometry items, the individual element of the **DATA** statement and current values of “Modal Setting” to be stored into individual scalar variables.

The number of variables can be extracted depends upon the canonical form of the geometry, the number of elements in the DATA statement or the type of the ‘Modal Setting’. For example, the canonical form of a point (X,Y,Z) would allow a maximum of 3 scalars. If a particular element of the canonical form is not desired, that ordinal position may be nulled by placing two commas in a row.

Example OBTAIN Statements:

```
OBTAINT/PT1,X,Y,Z  
OBTAINT/PV1,X,Y,Z,I,J,K  
OBTAINT/CI1,X,Y,Z,,,R  
OBTAINT/CU,CDIA,CRAD,,CSANG  
OBTAINT/THICK,CPS,,CCS  
OB      /MAXDP,CMAXDP  
OB      /FEDRAT,FR1RUF  
OBTAINT/VEA,A  $$ VEA is a DATA statement or label of geometry
```

4.14.1 OBTAIN/ ATTRIB, geometry-id

The syntax of the OBTAIN statement for geometry attributes is:

OBTAIN/ATTRIB,geometry-id,color,layer,line-type,line-weight

Where:

geometry-id	the geometry entity to obtain the attributes of
color	returns the color of the geometry entity
layer	returns the layer number of the geometry entity
line-type	returns the line style of the geometry entity
line-weight	returns the line weight of the geometry entity

For the returned color, line-type and line-weight value, see the [DRAFT/MODIFY](#) statements.

4.14.2 OBTAIN/ geometry-id

The syntax of the OBTAIN statement for geometry is:

OBTAIN/geometry-id,variable,...,variable

For the canonical form of each of these geometry types, see the [geometry section](#).

4.14.3 OBTAIN/ data-id

The syntax of the OBTAIN statement for DATA is:

OBTAIN/data-id,variable,...,variable

This form of the OBTAIN statement is to extract the individual element of the DATA statement. See section of DATA statement for details.

If the element to be extracted is a vocabulary word, the number corresponding to this word will be stored in the specified variable. If the element to be extracted is not a scalar or a vocabulary word, a value of zero will be stored in the corresponding variable.

4 GEOMETRY CONTROL STATEMENTS

4.14.4 OBTAIN/ "Modal Setting"

The OBTAIN statement may also be used to obtain the current values of the UNITS, THICK, TOLER, MAXDP, NUMPTS, MAXANG, CONTCT, FEDRAT, CUTTER, Part Surface, Drive Surface, Check Surface and tool settings.

The syntax of the OBTAIN statement for “Modal Setting” is:

OBTAIN/word,variable(s)

Where word is a valid vocabulary word that the following variables are assigned to:

CONTCT return 1 if ON and -1 if OFF
CS return the Check Surface entity in a text string.
CUTTER Up to 6 values available: in the order they are input in the CUTTER statement (except the APT 7 parameter CUTTER).

Note: If a *SET/VER command is issued with a version of less than 9.6, the values returned are in the order they are stored internally rather than in the order they are input in the CUTTER statement.

APT 7 parameter cutter input format would be converted to the **NCL** CUTTER format before the values can be obtained.

DS return the Drive Surface entity in a text string.

FEDRAT primary feed rate available

PS return the Part Surface entity in a text string.

MAXANG 1 variable available

MAXDP 1 variable available

NUMPTS 1 variable available

THICK 7 values available:
 part surface,
 drive surface,
 5 check surfaces

4 GEOMETRY CONTROL STATEMENTS

TOLER	4 values available: chordal, positional, AUTOST positional, AUTOST angular
UNITS	return 1 for Inches and 2 for MM
PSEUDO	Up to 6 values available. The values returned represent the parameters of the CUTTER/DISPLAY,dia,... command, if one is defined, or if a profile or symbol is used to define a mill cutter, then NCL will estimate the largest diameter and height of the cutter as represented by the geometry defining the cutter shape. If the cutter type is a blade or a lathe bit, or a display cutter had not been defined, then the actual values of the CUTTER/dia,.. command will be returned.
SHANK	Up to 5 values available. The values returned represent the parameters of the CUTTER/DISPLAY,SHANK command. The values returned as follows: Mill: Diameter, Height, Angle, Z-offset Lathe: X-width, Y-length, Z-height, X-offset, Y-offset The diameter and height will be estimated when a profile or symbol is used to define a mill tool. The width, length, and height will be estimated when a profile or symbol is used to define a lathe tool.
HOLDER	Up to 5 values available. The values returned represent the parameters of the CUTTER/DISPLAY,HOLDER command. The values returned as follows: Mill: Diameter, Height, Angle, Z-offset Lathe: X-width, Y-length, Z-height, X-offset, Y-offset The diameter and height will be estimated when a profile or symbol is used to define a mill tool. The width, length, and height will be estimated when a profile or symbol is used to define a lathe tool.

4 GEOMETRY CONTROL STATEMENTS

4.14.5 OBTAIN/ "Tool End Point And Tool Axis Vector"

The syntax of this form of OBTAIN statement is:

OBTAIN/variable1,variable2,variable3,variable4,variable5,variable6

This form of the OBTAIN statement is to extract the current tool end point coordinates and the tool axis vector components.

The XYZ coordinates are stored in the first three variables and the tool axis vector components are stored in the last three variables. If a particular element is not desired, that ordinal position may be nulled by placing two commas in a row.

Example:

OBTAIN/X,Y,Z	\$\$ Stored the current tool end points in the scalar variables X, Y, Z
OBTAIN/, ,I,J,K	\$\$ Store the current tool axis vector components in the scalar variables I, J, K.

4.15 label = PLACE/ [symlib,] sym, AT, pt [, SCALE, s] \$ [, ROTATE, r]

This command is used to place a symbol instance into the current unibase.

Where:

- | | |
|----------|---|
| label | - Name of the instance to be created. |
| symlib | - Optional name of the symbol library from which to retrieve the symbol. If not specified, currently loaded symbols will be searched for a match. |
| sym | - Name of symbol to be placed. |
| AT,pt | - Specifies the point at which the symbol will be placed. The attach point of the symbol will be placed at the point specified by "pt". |
| SCALE,s | - Specifies that the symbol will be scaled by the value "s". |
| ROTATE,r | - Specifies that the symbol will be rotated (about the current Z-axis, with the center of rotation at "pt") by "r" degrees. |

4.16 PRINT

This form of the PRINT statement will cause the canonical data associated with each geometric name to be printed in the **NCL** listing if the processor is running in batch mode. It also is used to format the printed output (PRINT/ 0, PRINT/ OFF and PRINT/ ON). This form of the PRINT statement is ignored when encountered in interactive mode.

Example PRINT Statements:

```
PRINT/0  
PRINT/ALL  
PRINT/POINT  
PRINT/PV  
PRINT/LN  
PRINT/SCALAR
```

See the PRINT statement under the MOTION STATEMENTS section for the other formats of the **PRINT** statement.

4.16.1 PRINT/ ALL

This statement causes **NCL** to print the canonical data for all geometry that has been defined to this point in the program.

4.16.2 PRINT/ geometry

This statement causes **NCL** to print the canonical data for all of one particular Geometry Type. Valid entries for the geometry variable are:

**ANOTE, CIRCLE, CURVE, LINE, PATERN, PLANE, POINT, PNTVEC,
SOLID, SURF, VECTOR and MATRIX.**

The canonical data for CURVES and non-primitive type SURFS will not be printed since they are rather complex and not of much use for the part program. Instead, the name and geometry type only will be printed for those two types.

All of the names will be printed in an alphabetically sorted order. Subscripted names will also be printed in a numerically ascending order.

If there is a REFSYS in effect at the time the PRINT statement is issued, an indication to that effect is shown for each geometry name and the canonical data will reflect the **REFSYS** coordinate system.

4 GEOMETRY CONTROL STATEMENTS

4.16.3 PRINT/ SCALAR

This will print all the scalar variables.

4.16.4 PRINT/ 0

This statement causes **NCL** to skip to the top of the next page before printing any more lines.

4.16.5 PRINT/ OFF, IN

This statement causes **NCL** to stop printing any output to the second half of the batch output file.

4.16.6 PRINT/ ON, IN

This statement causes **NCL** to start normal printing in the second half of the batch output file. This is the default condition and can be turned off with a PRINT/ OFF, IN statement.

4.17 REDEF

The REDEF statement is used to redefine an entity such as trimming a wire-frame entity, redefine an entity as CLOSE or OPEN, redefine a surface as a trimmed surface, etc. Once a line or circle has been redefined, the canonical form of that line or circle is permanently changed.

4.17.1 REDEF/ circle, line-1, line-2, modifier

This is a special fillet case where a circle is redefined to the tangency points of the two lines specified by line-1 and line-2. The modifier is used to specify which of the two possible arcs is the portion of the circle to be generated. The modifier may be **XLARGE**, **XSMALL**, **YLARGE** or **YSMALL**.

4.17.2 REDEF/ curve, CLOSE OPEN

This statement will cause **NCL** to mark a curve as “closed” or “open.” A curve marked closed will be considered to have its end points coincident and its slope

4 GEOMETRY CONTROL STATEMENTS

vectors at the end points parallel. **NCL** will usually automatically “close” a curve when it detects this condition. This does not modify the curves shape, only the way **NCL** evaluates the surface for motion generation purposes. The purpose of closing a curve is to have **NCL** ignore the extensions of the curve when driving a cutter across the coincident point.

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu icon sequence.

Edit > Close-Open > Pick the curve

4.17.3 REDEF/ curve1 [, near-pt1], EDGE, curve2 [, near-pt 2]
modifier modifier

This statement will cause **NCL** to duplicate the **NCL/CADD** trim and extend functions.

Where:

- A curve class entity such as a line, circle or conic that is to be trimmed or extended.

near-pt1 - An optional near point or direction modifier in relation to curve1 that determines the part of the curve to trim or extend.

When curve1 and curve2 intersect, then the side of the curve adjacent to the near point or modifier will be trimmed. When extending curve1, the near point or modifier specifies which side of the curve is to be extended.

Near-pt1 can be omitted when extending a line. The direction modifier can be **XSMALL**, **XLARGE**, **YSMALL**, **YLARGE**, **ZSMALL** or **ZLARGE**.

- Curve class type entity (such as a line, circle, or conic) or point (if curve1 is not a conic curve) identifier of geometry used to trim or extend curve1.
 - Optional near point on curve1 used to define the intersecting point when more than one intersection of curve1 and curve2 exist. A direction modifier can be used instead of near-pt2 to define the relative position of the intersection point.

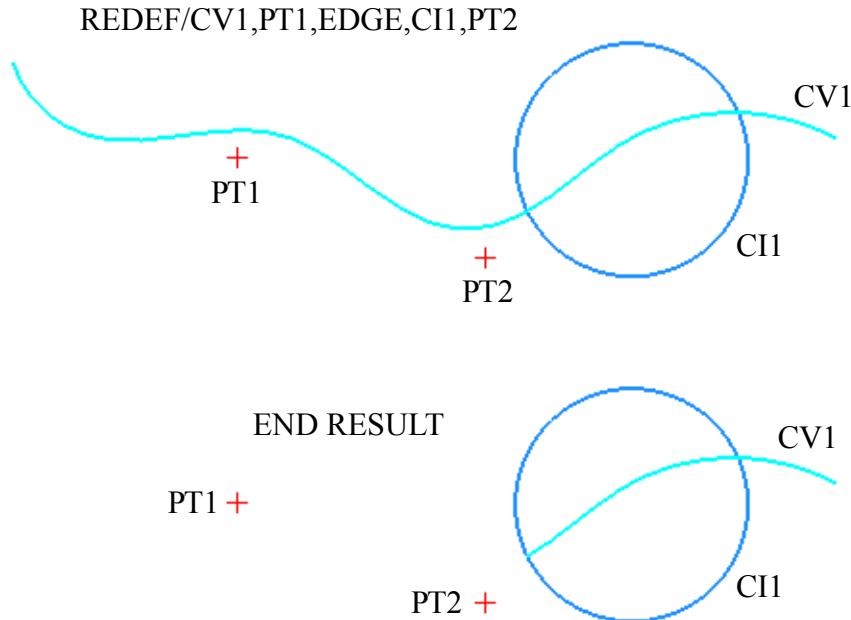
4 GEOMETRY CONTROL STATEMENTS

Available modifiers are XSMALL, XLARGE, YSMALL, YLARGE, ZSMALL or ZLARGE.

Note: Followings apply if the entity to be trimmed/extended is a composite curve.

- When a composite curve is extended, the extension will be along the tangent vector at the end of the curve, similar to the way B-spline curves are extended. A new line segment will be added to the composite curve to represent this extension.
- When a composite curve is extended multiple times in the same direction additional lines are not created; instead, the original extending line is modified to be the new extension. This prevents the creation of unnecessary geometry.
- A trimmed composite curve will behave in the same manner as a trimmed B-spline curve in that when the end of the curve is reached the tangency extension will be used and not the underlying (trimmed) components of the composite curve. If the composite curve is untrimmed after being trimmed, then the original shape of the composite curves will be restored.

The illustrations below show how this statement may be used.



It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu sequence:

Edit > Trim_Extend

4 GEOMETRY CONTROL STATEMENTS

4.17.4 REDEF/ curve1 [, cv-name] [, near-pt1], EDGE, curve2 \$ modifier

[, near-pt2], EDGE, curve3 [, near-pt3]
modifier modifier

This statement will duplicate the **NCL/CADD** midtrim functions.

Where:

curve1 - A curve class entity such as a line, circle or conic that is to be mid trimmed.

cv-name - An optional identifier used to name the second part of the curve when it is split into two separate entities. If cv-name is not specified, then the auto naming feature of **NCL** will create the name for the second part of curve1.

near-pt1 - Optional near point on curve used to define the modifier part of curve to keep.

If the specified near-pt1 lies between the intersections of curve2 and curve3 with curve then the central part of curve is kept, otherwise the curve is split into two segments laying outside the intersections of the trimming curves. Direction modifiers can be used instead of near points. Available modifiers are **XSMALL**, **XLARGE**, **YSMALL**, **YLARGE**, **ZSMALL** or **ZLARGE** and **CENTER**. All of the -LARGE and -SMALL modifiers will split curve and **CENTER** will keep the central portion of curve. The default is **CENTER**.

curve2 - The identifier of the first entity used to mid trim curve. It can be a point (if curve1 is not a conic curve), line, circle or conic curve if curve1 is a line.

near-pt2 - Optional near point on curve2 used to define the modifier's closest intersection point of curve1 and curve2.

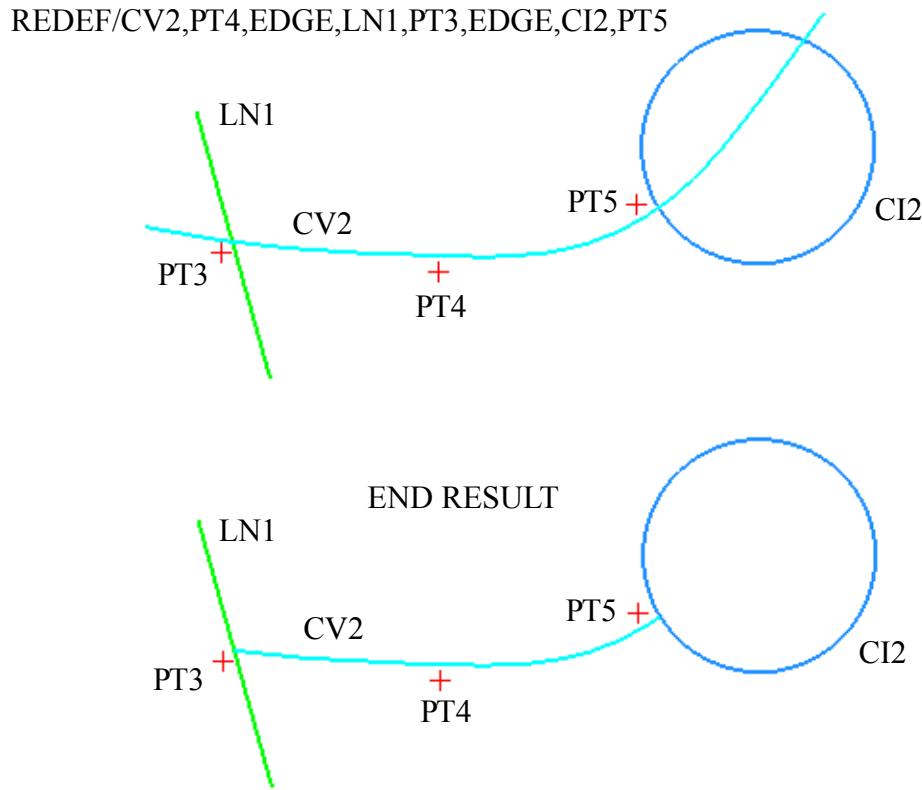
curve3 - Identifier of the second entity used to mid trim curve. See specification for curve2.

4 GEOMETRY CONTROL STATEMENTS

near-pt3 - Optional near point on curve3 used to define the modifier's closest intersection point of curve and curve3.

Note: Composite curves that are closed will not have their ends joined after a midtrim where the ends are kept. Since the geometry for a composite curve is defined in a specific order, joining the ends would modify this order. As a result, the output from a midtrim will result in two composite curves when the portions of the curve that are kept are the ends.

The illustrations below show how this statement may be used.



It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu sequence:

Edit > Mid_End Trim

4.17.5 REDEF/ surface

This statement is used to change the associated properties of a surface such as to mark a surface as a closed surface or an open surface, to change the default automatic PARLEM direction of a surface, to mark a surface as BASE or FACE, to modify a trimmed surface and to redefine a non-trimmed surface as a trimmed surface.

4.17.5.1 REDEF/ surface, CLOSE, 0

1

This statement will cause **NCL** to mark a surface as “closed.” A surface will be marked as closed in the "U" direction if the edge from the start of the first defining entity to the start of the last defining entity closely matches the edge from the end of the first defining entity to the end of the last defining entity. A surface will be marked as closed in the "V" direction if the edge along the first defining entity closely matches the edge along the last defining entity.

When "0" is specified as the last parameter, the surface will be closed in the "U" direction. When "1" is specified as the last parameter, the surface will be closed in the "V" direction.

During toolpath generation if the tool reaches the closed seam of a “closed” surface, the tool will continue to follow the surface rather than the extension of the surface.

Note: This feature does not apply to [QUILT](#), [TRIMMED](#) or [NET](#) surfaces.

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu icon sequence.

Edit > Close-Open > Pick the surface

4.17.5.2 REDEF/ surface, OPEN, 0

1

This statement will cause **NCL** to mark a surface as "open."

When "0" is specified as the last parameter, the surface will be open in the "U" direction. When "1" is specified as the last parameter, the surface will be open in the "V" direction.

4 GEOMETRY CONTROL STATEMENTS

During toolpath generation if the tool reaches the closed seam of an “opened” surface, the tool will follow the surface extension rather than the portion of the surface on the opposite side of the seam.

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu icon sequence.

Edit > Close-Open > Pick the surface

4.17.5.3 REDEF/ surface, [PARAMS, [SWAP,]] \$

```
[ , REVERS, 0           [ , 1           [ , NORMAL ]]]] $  
      1                 NORMAL  
      NORMAL  
  
[ [ , ] PARLEM, 0 ] [ [ , ] BASE ]  
      1             FACE  
      -1
```

The purposes of this statement are:

1. To modify the UV, normal direction of a surface.
 - “PARAMS” is required for syntax.
 - “SWAP” means swap the UV direction of the surface, i.e. U becomes V and V becomes U with the normal direction reversed.
 - “REVERS” is used to reverse the U, V or normal direction of a surface. “0” means to reverse the U direction. “1” means to reverse the V direction. “NORMAL” means to reverse the normal direction and has the same meaning as “SWAP”.
2. To change the automatic setting of the PARELM direction of a surface.
 - “PARLEM” is required for syntax.
 - The PARELM direction is the direction the tool axis aligns itself with when using one of the PARELM tool axis modes. In automatic mode the direction closest to the current tool axis will be chosen. If this is not the direction desired, the REDEF command can be used to modify the direction.
 - A "0" in the last parameter will set the tool axis to the "V" rulings. A "1" will set the tool axis to the "U" rulings and a "-1" will set the PARELM mode back to automatic.

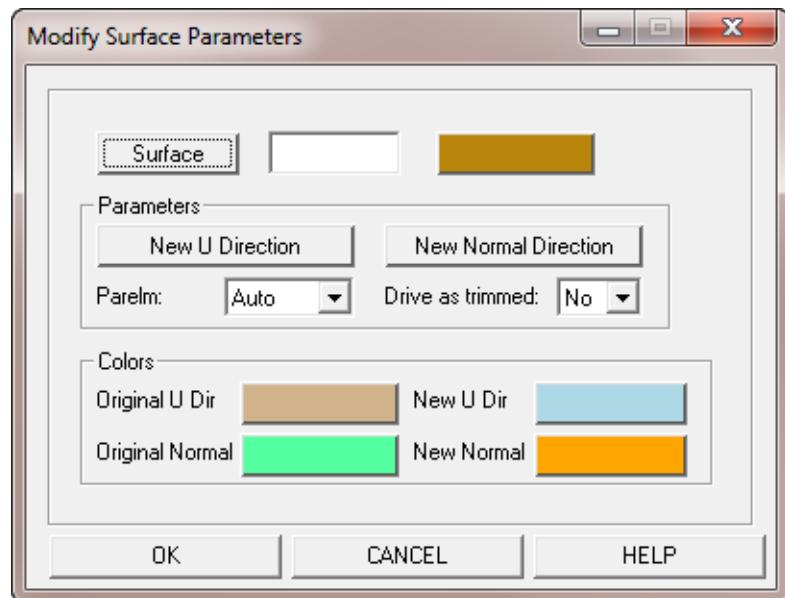
4 GEOMETRY CONTROL STATEMENTS

3. To specify whether to drive the extensions of a trimmed surface or the underlying surface. “BASE” drives the underlying surface and “FACE” respects the outer boundary of the trimmed surface.

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu icon sequence with the Modify Surface Parameter form.

Edit > Surfaces > Parameters > Pick the surface

The following shows the graphical interactive interface of this form and a description of how to use this interface.



This form allows the user to manipulate surface parameters:

- swap the U and V parameters,
- reverse the U or V direction,
- reverse the direction of a surface normal,
- set the PARELM type,
- set the drive type for a trimmed surface.

Surface

Select a surface.

4 GEOMETRY CONTROL STATEMENTS

(Text Field)

Contains the current surface's label.

Color

Choose a color to highlight the selected surface.

New U Direction

Pick an assist-vector along the desired U-direction from the eight possibilities. The other assist-vector adjacent to the picked U-direction will be the new V-direction. The choice determines whether the U and V parameters are to be swapped, and whether U or V is to be reversed. The user can pick the current U-direction or press the Alt-Act button to leave the current parameters in place.

New Normal

Pick an assist-vector along the desired normal direction. The user can pick the current normal direction or press the Alt-Act button to leave the current choice in place.

Parelm

Choose between Auto, Along V, or Along U.

Drive as Trimmed

Choose between Yes (FACE in REDEF command syntax) and No (BASE in REDEF command syntax). Ignored if the surface is untrimmed.

Colors

The sub form allows the user to choose how to highlight assist-vectors:

the original and new U-direction, and the original and new normal direction.

OK

Click this button to accept the changes, close the form and output the corresponding REDEF/surf command.

CANCEL

Click this button to cancel and close the form.

HELP

Click this button to open the inline help of this form.

4.17.5.4 [sfn =] REDEF/ surface

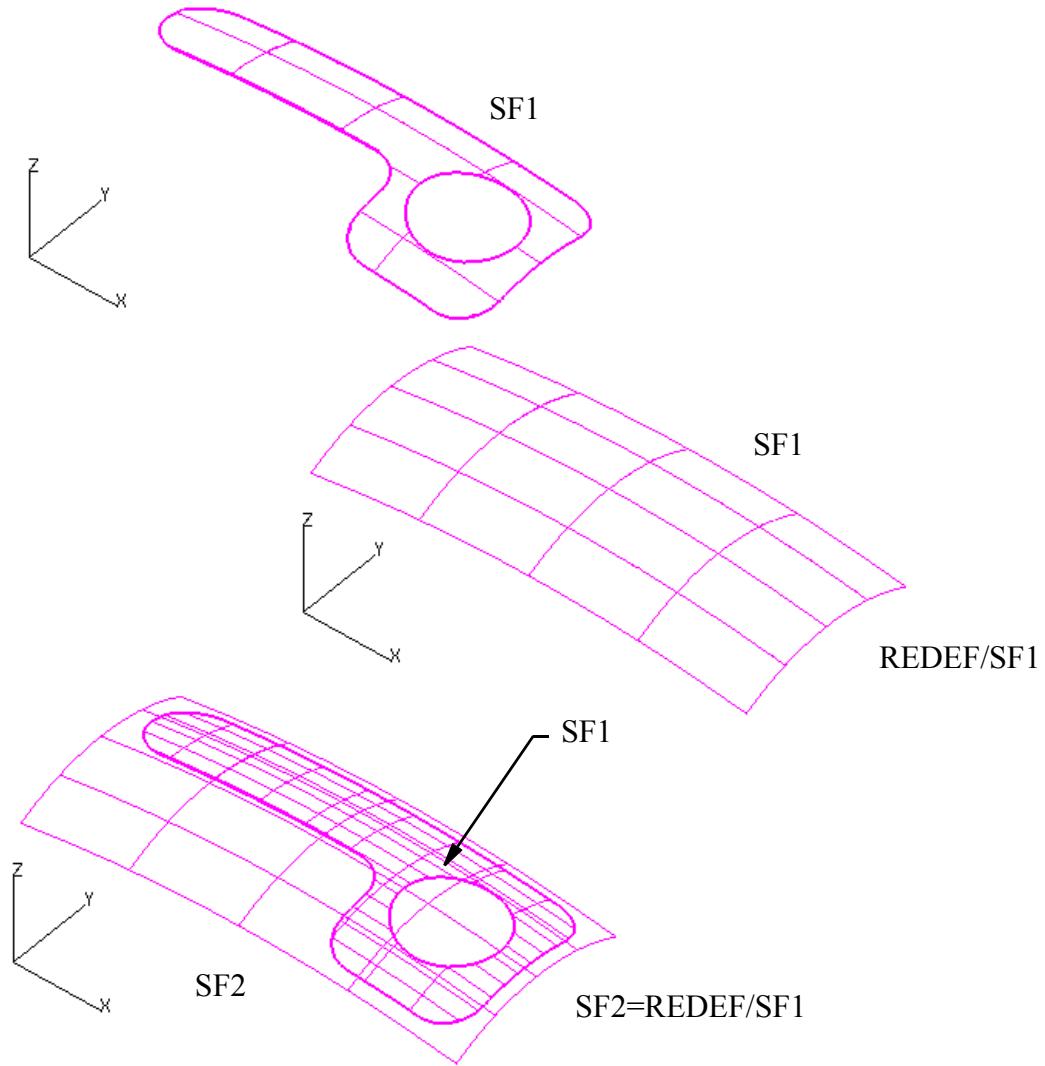
This command is used to extract or modify a surface into its basic format.

If “surface” is trimmed, this command extracts its base surface, otherwise, if “surface” is not trimmed and has a primitive type of SPHERE, CYLINDER, CONE, or a Revolved Surface, this command creates a full 360-degrees primitive type surface. If “surface” is untrimmed and is not a surface primitive then it is copied if a new label is given, otherwise the command is ignored.

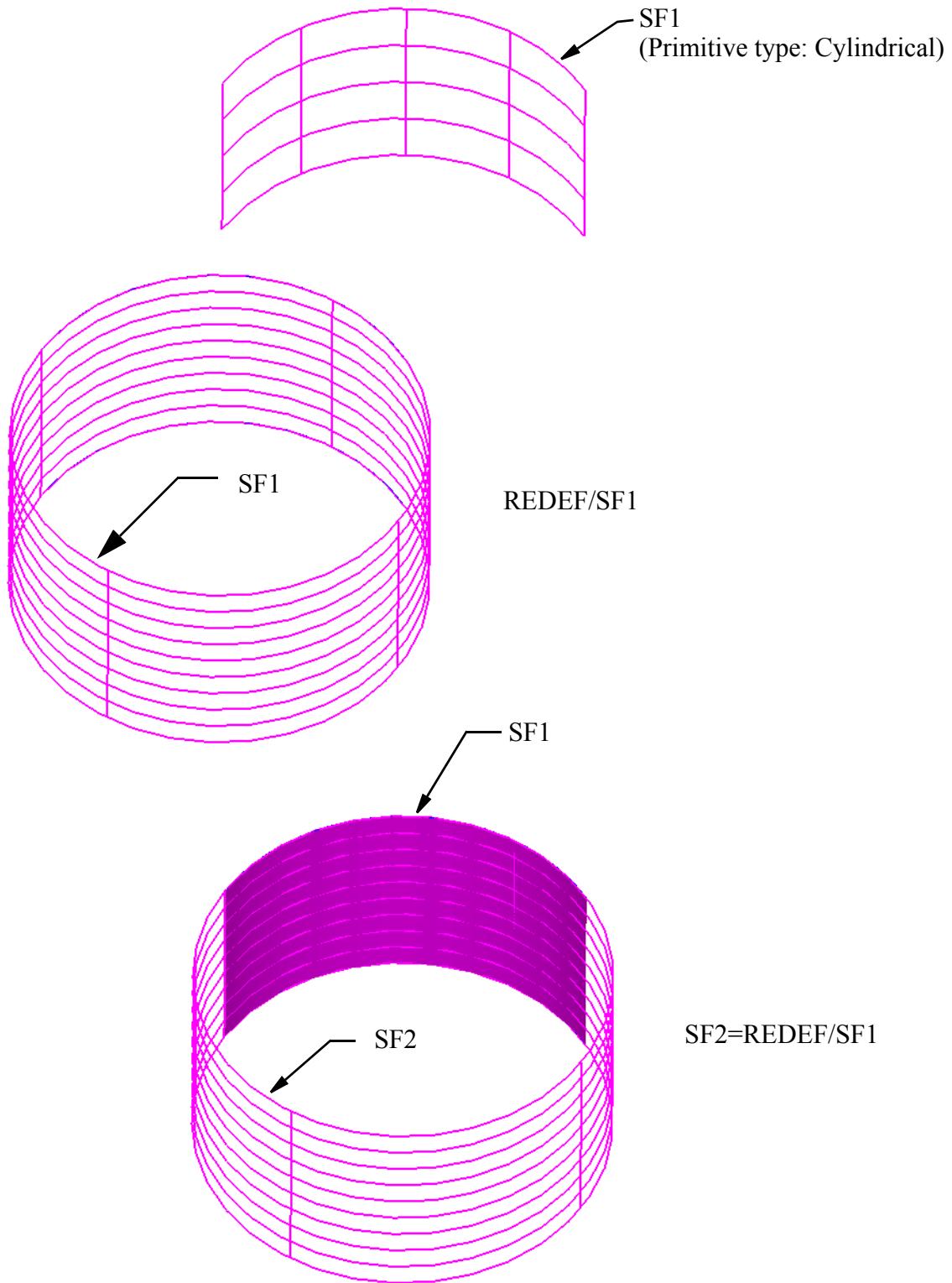
If “sfn” is not given, the original surface will be overwritten, otherwise a new surface with the label “sfn” will be created.

The illustrations on next page show how this statement may be used.

4 GEOMETRY CONTROL STATEMENTS



4 GEOMETRY CONTROL STATEMENTS



4 GEOMETRY CONTROL STATEMENTS

**4.17.5.5 [sfn =] REDEF/ sf1 [, OUT, cv1] [, modifier] [, IN, cv2, ...]
sf2
pl1**

This command is used to create/modify a trimmed surface or create/modify a non-trimmed surface into a trimmed surface.

If “sf1” is trimmed, (a copy of) its base surface is used as the new trimmed surface’s base, otherwise (a copy of) “sf1” is used as the base surface.

If the outer boundary is a curve, it is projected onto the base surface to create a bounding surface curve.

If the outer boundary is a surface or a plane, it is intersected with the base surface to create a bounding surface curve.

In the case when the surfaces do not intersect and the base surface is a primitive type, the algorithm attempts to extend the surface so that it intersects with the bounding entity. If the base is planar, or has surface of revolution primitive type (such as SPHERE, CYLINDER, CONE or Surface of Revolution), then it is extended on the side closest to the bounding surface.

If the bounding surface curve is closed, it is used as the outer trimming boundary.

If the bounding surface curve is not closed, it is intersected with the base surface boundary box to make the outer trimming boundary. To choose between several possible trimmed surface solutions, an optional “modifier” can be used to designate which part of surface to keep:

- A surface point, given by a pair of UV-parameters.
- A 3-D near point that is projected onto the base surface.
- A vector modifier - a point in the middle of the bounding surface curve is moved in the vector direction, and the resulting point is used as a near point modifier.
- A directional modifier (XLARGE, XSMALL, YLARGE, YSMALL, ZLARGE, ZSMALL) is interpreted as the unit vector in the specified direction.
- The vocabulary word LARGE or SMALL is used to select the largest or smallest of all possible components.

4 GEOMETRY CONTROL STATEMENTS

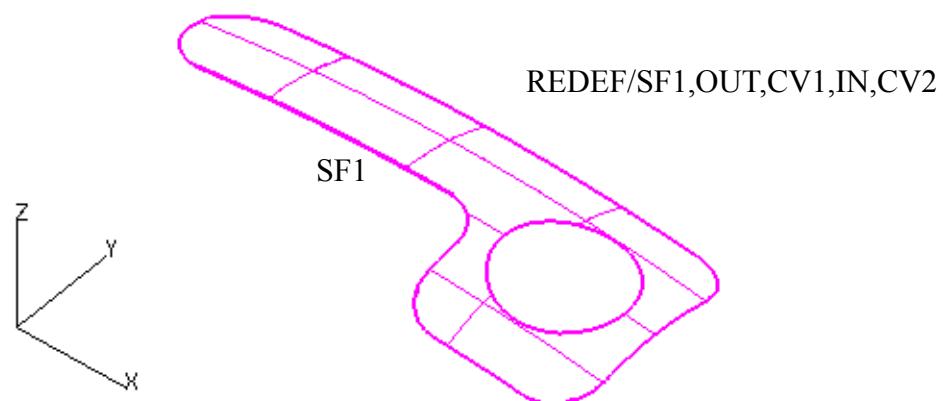
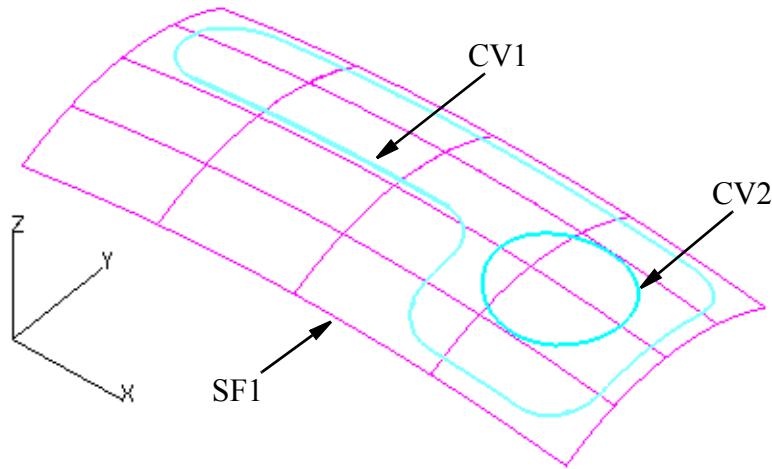
If an inner boundary is specified, it must be a closed curve or circle. The following rules apply to inner boundary curves:

- The curve is projected onto the base surface to create a closed surface curve.
- If an inner boundary curve intersects the outer trimming boundary or other inner boundaries, the correct polygon intersection is used as the resulting trimming.
- Multiple inner boundary curves can be specified on a single command.

If “sf1” is trimmed, its existing trimming is intersected with the new one and the correct polygon intersection is used as the resulting trimming boundary.

If “sfn” is not given, the original surface will be overwritten, otherwise a new surface with the label “sfn” will be created.

The illustration below shows how this statement may be used.



4 GEOMETRY CONTROL STATEMENTS

It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence



4.17.5.6 [sfn =] REDEF/ [pl1,] OUT, cv1 [, IN, cv2, . . .] sf1

This command is used to create a planar trimmed surface.

In the case when no plane is specified the outer boundary has to be a closed planar curve or a full circle - then the curve/circle plane is used to create the base surface and the curve/circle projections makes the outer trimming boundary.

If a plane “pl1” is specified and the outer boundary is a curve, “cv1”, (does not have to be on “pl1”) or a circle, it is projected onto the plane and the projection points are used to create the base surface and the outer trimming boundary.

If a plane “pl1” is specified and the outer boundary is a surface “sf1”, the intersected curve (must be a closed curve) is used to create the base surface and the outer trimming boundary.

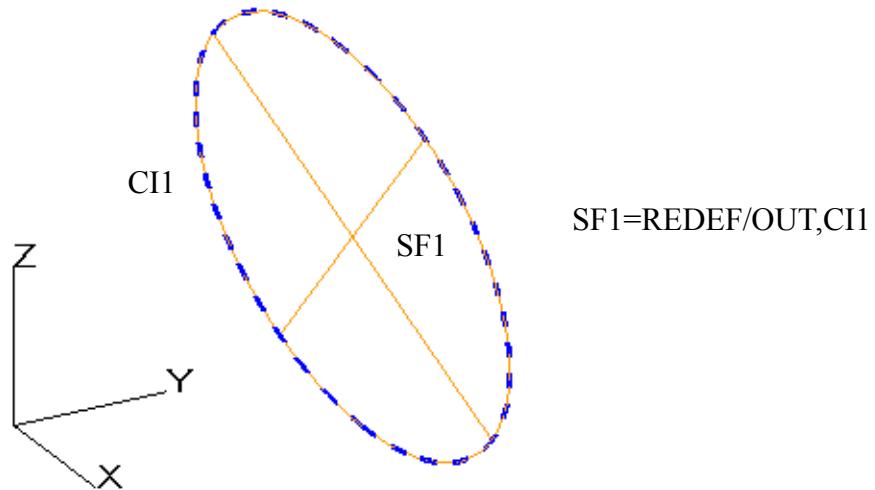
If an inner boundary is specified, it must be a closed curve or circle. The following rules apply to inner boundary curves:

- The curve is projected onto the base surface to create a closed surface curve.
- If an inner boundary curve intersects the outer trimming boundary or other inner boundaries, the correct polygon intersection is used as the resulting trimming.

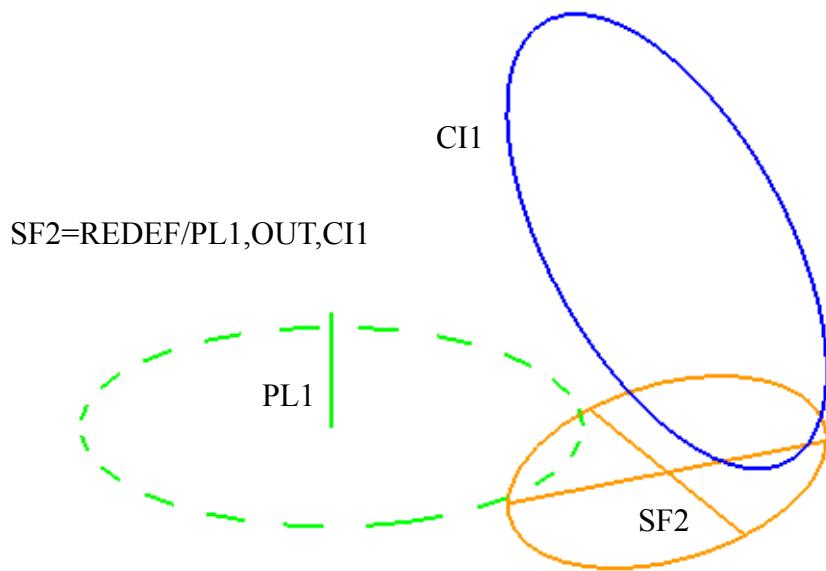
Multiple inner boundary curves can be specified on a single command.

The illustrations on next two pages show how this statement may be used.

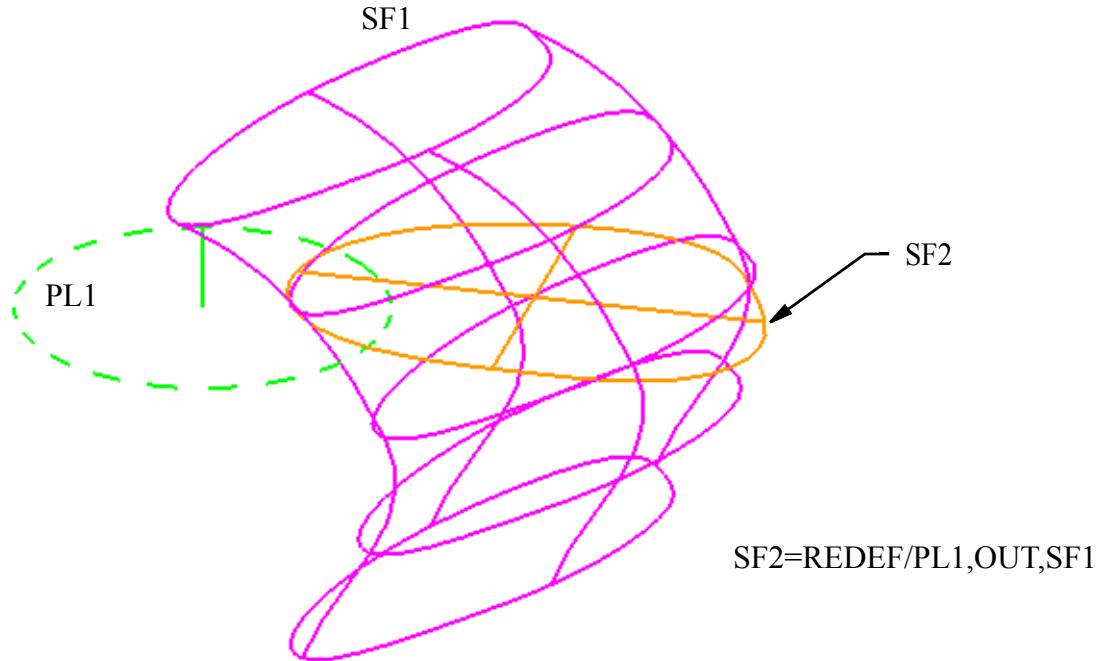
4 GEOMETRY CONTROL STATEMENTS



SF1=REDEF/OUT,CI1



SF2=REDEF/PL1,OUT,CI1



It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence



**4.17.5.7 [sfn =] REDEF/ sf1, REMOVE, i1 [[, THRU], i2, ...]
[0,] ALL**

This command is used to modify a trimmed surface.

If “sf1” is trimmed, this command removes its boundary curve(s) indicated by their numbers. The outer boundary has the number 0 and the inner boundaries are numbered 1,2,3, etc. If the boundary number is out of range for the surface, it is ignored. The “THRU” clause can be used to specify the range of inner boundaries to be removed.

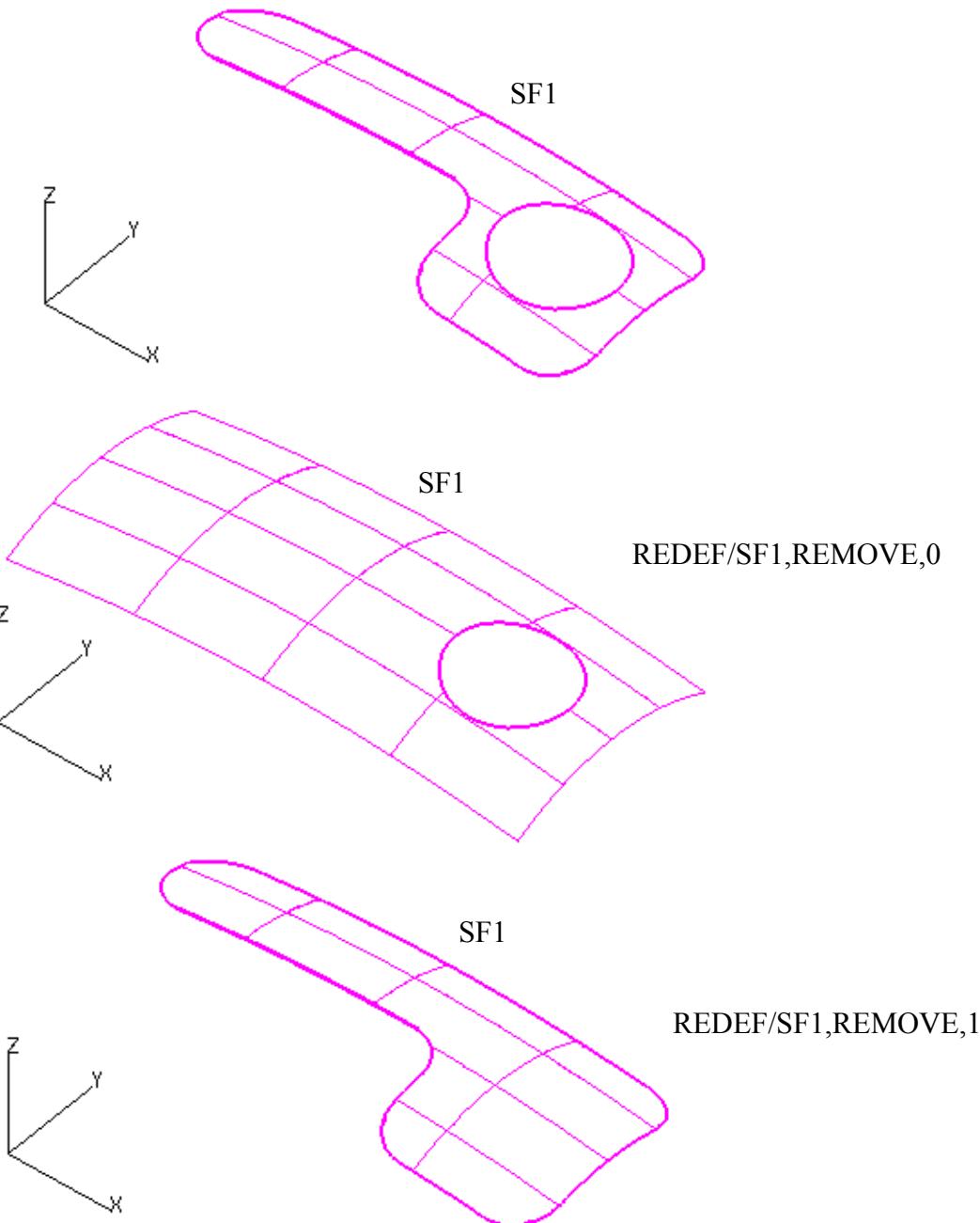
“ALL” specifies to remove all the inner boundaries of the trimmed surface.

4 GEOMETRY CONTROL STATEMENTS

If “sf1” is not given, the original surface will be overwritten, otherwise a new surface with the label “sf1” will be created.

If “sf1” is not trimmed it is copied if a new label is provided, otherwise the command is ignored.

The illustrations below show how this statement may be used.



4 GEOMETRY CONTROL STATEMENTS

It should be noted that while the command can be entered as shown, this statement is best produced by using the following on screen menu icon sequence



4.17.6 REDEF/ geo

This statement allows you to untrim previously trimmed wireframe entities.

Where:

geo - Represents the name of the wireframe entity to untrim. The entity will be restored to its default state. For example, a line will be untrimmed to be one inch (25 mm) long, a circular arc will become a 360 degree circle, and a curve will be returned to its original state.

Examples:

```
REDEF/CI2  
REDEF/CV2  
REDEF/LN1
```

4.18 REFSYS

The REFSYS (REFerence SYStem) statement allows the establishment of an alternate coordinate system to be used for the purpose of creating geometry. The coordinate system is defined by a specified matrix. When a REFSYS statement is in effect **NCL** calculates all subsequent geometry definitions relative to the newly established coordinate system. Tool motion output is not affected by a REFSYS and it is generally recommended that tool motion not be processed with REFSYS in effect. The **MODSYS** command should be used for establishing alternate coordinate systems that affect both geometry and tool motion.

Example:

```
REFSYS/MX1  
REFSYS/NOMORE
```

4.18.1 REFSYS/ matrix

This statement establishes the coordinate system defined by the specified matrix to be the currently active coordinate system for the purpose of defining geometry.

4.18.2 REFSYS/ NOMORE

This statement notifies **NCL** that REFSYS is no longer in effect.

4.19 RENAME

Allows you to rename entities from within the part program file using the following syntax:

```
RENAME/org_name-1,new_name-1,...,      $  
          org_name-n,new_name-n
```

Where:

`org_name` - The original name of the entity to be renamed.

`new_name` - The new name assigned to the entity.

4.20 REMOVE

The REMOVE statement is used to delete a type of geometry, a specific geometric entity, a scalar value, a text string entity, a group of geometric entities or all defined geometric, text strings entities and scalar entities (ALL). When an entity is removed, all information about that entity is deleted from **NCL**'s internal tables. It is as though the entity had not been defined in the first place. The removal of an entity has no effect on statements that were processed before the REMOVE statement is processed.

```
REMOVE/geometry-id  
REMOVE/geometry-type  
REMOVE/scalar-id  
REMOVE/"text-string-id"  
REMOVE/ALL
```

Where entity-id can be a single entity (except a single text entity, a single text entity must be enclosed in a pair of double quotes), name or a portion of a name

4 GEOMETRY CONTROL STATEMENTS

(wild card), followed by an asterisk (*). This will remove all entities with names starting with the characters preceding the "*" The geometry-id may also be specified as a "THRU clause." For example, "CI4, THRU, CI57" would cause all geometry with names CI4, CI5, CI6, . . . to CI57 to be removed.

Examples:

REMOVE/LN	\$\$ This removes all the line \$\$ entities.
REMOVE/X	\$\$ This removes the entity \$\$ X (See notes below for \$\$ what will be removed for \$\$ this id label).
REMOVE/PT1, THRU, PT17, PV, Y, B*	\$\$ This removes all the point \$\$ entities PT1 through PT17 \$\$ all point-vector entities, \$\$ Y (See notes below for \$\$ what will be removed for \$\$ this id label.) and all \$\$ the entities (scalar, \$\$ geometry or text-string) \$\$ with id starts with "B".
REMOVE/ALL	\$\$ This removes all the \$\$ geometries, scalars and \$\$ text-strings.
REMOVE/CI, LN3, A1	\$\$ This removes all the \$\$ circle type entities, the \$\$ line entity LN3 and A1 \$\$ (See notes below for what \$\$ will be removed for this \$\$ id label).
REMOVE/ "ABC"	\$\$ This removes the text \$\$ string entity with the id \$\$ ABC.

Notes:

1. If the text string variable (TX1) only contains a single label of other entities (such as scalar, geometry, text string), "REMOVE/TX1" will not remove the text string variable, instead the entity specified in this string variable will be removed.

4 GEOMETRY CONTROL STATEMENTS

Example:

- i. The following sequence of commands will not remove the text string TX1, instead the geometry PT1 will be removed.

```
PT1      =POINT/1,2  
TX1      ="PT1"  
REMOVE/TX1
```

- ii. The following sequence of commands will not remove the text string TX2, instead the scalar A1 will be removed.

```
A1      =POINT/1,2  
TX2      ="A1"  
REMOVE/TX2
```

- iii. The following sequence of commands will not remove the text string TX4, instead the text string TX3 will be removed and PT1 will not be removed.

```
TX3      ="PT1"  
TX4      ="TX3"  
REMOVE/TX4
```

2. If the contains of a text string variable (TX5) is a series of entity labels separates by a comma or just a regular text string, “REMOVE/TX5” will not remove the entities or the string specified in this text string variable, instead **NCL** will output an error message of “Valid geometry entity required”.

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu icon sequence if the entities to be removed are geometry and displayed on the graphic area.

Edit > Remove > Pick the entities

4.21 RESERV

The RESERV statement allows an identifier to be subscripted. If a subscripted variable is used without having been RESERVED, it will automatically be RESERVED for the maximum value (1,000,000).

4 GEOMETRY CONTROL STATEMENTS

This statement causes **NCL** to mark the identifier with a subscripted numeric value. The smallest subscript allowed is 1. The largest subscript allowed is specified by the scalar. An identifier that has been RESERVed, must always be referenced with a subscript.

The use of the RESERV statement is primarily for APT compatibility and normally is not used by **NCL** users.

Example:

```
RESERV/A,10
RESERV/PTX,100,PTY,100,PTZ,100
NPTS=NUM(PTY)
DO/100,I=1,NPTS
CYCLE/DRILL,FEDTO,.5,RAPTO,.1,15
GOTO/PTY(1)
100)CONTIN
CYCLE/OFF
```

4.21.1 Inclusive Subscripted Variables

Command Syntax:

```
var([a1,]THRU,a2[, [INCR ],a3])
      ALL     DECR
```

Where:

- var - A subscripted variable.
- a1 - The starting subscript value.
- a2 - The ending subscript value.
- a3 - The increment value.

If a1 is omitted, then 1 is assumed. If ALL is used in place of a2, then the highest subscript defined is used. If the INCR or DECR phrase is omitted, INCR 1 is assumed.

Examples:

```
CV1=CURVE/PTCV(2,THRU,20)
CV2=SPLINE/PVCV(THRU,ALL,INCR,2)
SF1=SURF/CVX(NUM(CVX),THRU,1,DECR,1)
POCKET/PL1,PL2,CVPK(ALL)
```

4.22 REVERS

4.22.1 REVERS/ data-statement

This format of REVERS statement allows the order of the entities in a **DATA** statement to be "reversed". The syntax of this format of REVERS statement is:

```
REVERS/data-statement-label
```

Example:

```
ABC=DATA/1,2,3,4,5,6  
REVERS/ABC
```

This effectively changes the ABC DATA statement to:

```
ABC=DATA/6,5,4,3,2,1
```

4.22.2 REVERS/ geometry

This format of REVERS statement allows a geometric entity to be "reversed." This means the direction of geometry as described by its canonical data is changed to point in the opposite direction. The syntax of this format of REVERS statement is:

```
REVERS/geometry-id[,direction-modifier]
```

Another identifier may be added to the front of a REVERS statement. For example:

```
LN2=REVERS/LN1
```

In this case, the identifier LN2, will be defined as the REVERSED geometry and the original geometry, LN1, will be left unchanged.

Valid geometry types to REVERS are **VECTORS**, **POINT-VECTORS**, **LINEs**, **CIRCLEs** and **CURVEs**. REVERS ignores whether **CANON/ ON** is in effect if it is assigning the REVERSED canonical data to "geometry-id."

If a second identifier is given at the beginning of the statement, **NCL** will check for a **CANON/ ON** state if it determines the second identifier had been previously defined.

4 GEOMETRY CONTROL STATEMENTS

When a VECTOR or POINT-VECTOR is REVERSED, the delta values become the negative of their original values.

When a LINE is REVERSED, the end point becomes the start point and the delta values become the negative of their original values.

When a CIRCLE is REVERSED, the delta values of the vector defining the axis of the circle become the negative of their original value. Only non-360 circles may be REVERSED.

When a CURVE is REVERSED, the first point on the curve becomes the last point and the last point becomes the first point and all other points are re-ordered in reverse order of what they were. The slope and curvature values are recalculated to generate a curve that matches the original curve.

The "direction-modifier" is optionally used to indicate in which direction the REVERSED geometry should be going. If the geometry is already going in the general direction the "direction-modifier" is pointing in, no reversing of the canonical data is done.

By using a "direction-modifier," the user can ensure a piece of geometry is not REVERSED twice if the same REVERS statement is reprocessed for some reason.

When testing if the geometry is going in the direction specified by "direction-modifier," the direction of a CIRCLE is determined by a vector that is tangent to the circumference of the existing portion of the arc, midway between the two end points.

The direction of a CURVE is defined as the direction of a vector tangent to the middle of the curve. The middle of a CURVE is at the point 50% along its length.

The vocabulary words **POSX**, **NEGX**, **POSY**, **NEGY**, **POSZ** and **NEGZ** may be used as valid "direction-modifier" words.

Example:

```
REVERS/LN23  
PVY=REVERS / PVX  
REVERS/CI4, POSX  
REVERS/CV9, NEGY  
CV9R=REVERS/CV9, NEGY (creates CV9R and leaves CV9 alone)
```

4 GEOMETRY CONTROL STATEMENTS

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu icon sequence.

Edit > Reverse > Pick the entities

4.23 **sym1 = SYMBOL/ [AT, point,] geometry-list**

The SYMBOL statement is used to define a symbol with the name “sym1” inside the part program. The defined symbol “sym1” will be composed of all the geometry entities listed in the geometry-list. The origin point of the symbol can be specified by the optional parameter “AT,point”.

Symbols created with this command are considered simple symbols comprised of geometry only without multiple snap nodes, text, or text nodes.

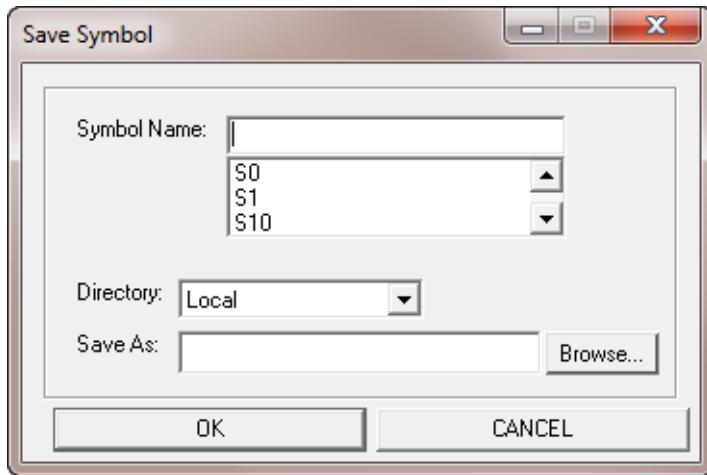
These symbols are stored in the active unibase only and are not automatically stored to an external file, such as when using the interface to create a symbol.

Note:

1. Only the entity types POINT, LINE, CIRCLE, CURVE, SURFACE and SOLID are allowed in the geometry-list.
2. Subscripted symbol names are allowed.
3. Symbol names can be up to 63 characters long if the symbols are used for tool display, e.g. cutter.
4. Symbol names can be up to 63 characters long if auto naming is not used for instance creation.
5. If auto naming is used to create instance from symbol, the maximum number of characters for symbol is anywhere from 55 to 61. If the total number of instances to be created is less than 10, the symbol name can have a maximum of 61 characters. If the total number of instances is less than 100, the symbol name can have a maximum of 60 characters.
6. The symbol name defined with this command is not case sensitive.
7. The symbol created with this command can be stored to the system symbol folder with the following interface sequence to open the save symbol form as shown on next page.

Tools > Cadd > Symbol > Save

4 GEOMETRY CONTROL STATEMENTS



Highlight the symbol that needs to be saved. The list can be moved up and down by moving the slide bar. Toggle the Directory to either Local or System. Click OK to accept the changes and close the form if no file name is specified. The selected symbol will be saved in the corresponding folder with the specified name if file name is specified, otherwise, the symbol will be saved to file "sym.sy" where "sym" is the name of the selected symbol.

4.1 ZSURF

The ZSURF statement is used to assign a Z coordinate to points that are defined without Z values. For points, or point-vectors, that are defined with an explicit Z value, ZSURF will be ignored. The valid syntax constructs for the ZSURF statement are:

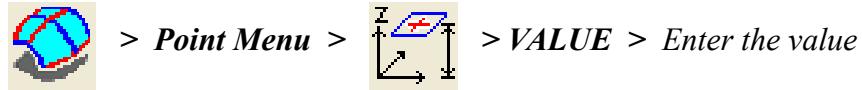
4.1.1 ZSURF/ scalar

The scalar specifies the Z coordinate that will be given to all points, or point-vectors subsequently defined without a Z value.

Examples:

ZSURF / 2

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu icon sequence as shown on next page..



4.1.2 ZSURF/ plane

All subsequently defined points or point-vectors without a Z value will fall on the specified plane. The plane need not be perpendicular to the Z-axis but may not be parallel to it.

Examples:

ZSURF / PL1

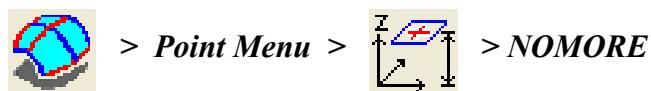
It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu icon sequence.



4.1.3 ZSURF/ NOMORE

This construct specifies that ZSURF is no longer in effect.

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu icon sequence.



5 ACCESSING UNIBASE FILES

5 UNIBASES

A **NCL** unibase (universal data base) file is used to store and retrieve data created by **NCL**. Data stored in a unibase includes: geometric entities and their associated attributes (layer, color, line style, line weight, and logical pen), scalars, drafting entities, views, drawings, and symbols.

5.1 UBFN

The UBFN (Unibase File Name) statement is used to establish access to a unibase file from within a **NCL** part program file.

5.1.1 UBFN/Unibase file name

This statement opens the specified unibase file. Once a unibase file is opened the **GET** and **PUT** commands can be used to get entities out of or put entities into the unibase file.

If the specified file does not exist a new unibase file is created.

If another unibase file is already opened **NCL** will generate an error.

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu sequence.

File > Secondary Unibase > Open Unibase

5.1.2 UBFN/ CLOSE

This statement causes the previously opened unibase file to be closed.

Example UBFN Statements:

```
UBFN/PART401.u  
UBFN/CLOSE
```

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu sequence.

File > Secondary Unibase > Close Unibase

5.2 ASCII Unibase Files

Unibase files are normally stored as binary files. However, it is possible to store a unibase file as a text (or ASCII) file. The advantage of an ASCII unibase file is that it can be transferred to different platforms (such as UNIX to Windows) and loaded directly into **NCL**. To create an ASCII Unibase file, simply use the “.ud” or “.UD” (Windows only) extension when specifying the file name. **NCL** will automatically create an ASCII Unibase.

Example:

```
UBFN/file.ud
```

5.3 GET

The GET statement retrieves specified entities from the unibase file referenced in the last **UBFN** statement. The entities retrieved are added to the currently active model. The GET statement provides a convenient way to GET selective entities from a unibase. This is desirable when the entire contents of a unibase file is not necessary for the task (fewer entities means more efficient interactive processing). GET commands can also be used to transform geometry by a specified matrix. For example, rotating geometry or mirroring geometry to create opposite hand parts. Items that may be retrieved from a unibase using the GET command are scalars, points, lines, circles, planes, vectors, point-vectors, curves, surfaces, matrices, patterns, DATA statements, symbols and symbol instances.

Example GET Statements:

```
GET/PT4  
GET/DBPT34,AS,PT15  
GET/LN  
GET/A23*  
GET/SF2,THRU,SF13  
GET/SUBPT(33),THRU,SUBPT(95)  
GET/ALL  
GET/PT2,PT3,PT5,CV2,THRU,CV5,SF18,THRU,SF12  
GET/PT,LN,SF,CIRCLE  
GET/A*,B*,C*,PT  
GET/DATA  
GET/ABC $$ ABC is a DATA statement
```

5 ACCESSING UNIBASE FILES

5.3.1 GET/ alphanumeric-character-string*

This statement causes **NCL** to get all items (*except symbols and instances*) from the unibase file whose names start with the characters given in the alphanumeric character string. For example, the statement GET/G589* would get all item that have a name starting with the characters G589 regardless of what type of entity it is.

5.3.2 GET/ entity-id, AS, new-id

The entity-id is the name of the item in the unibase file. "AS" indicates that the item is to be renamed. The new-id is the new name the item is to be known as to the part program.

5.3.3 GET/ entity-id [, THRU, entity-id]

The entity-id is the name of an item in the unibase file.

The THRU option causes **NCL** to get a Range Of Items from the Unibase file. The range for geometries may be specified using names that use the default auto name generation prefixes which are:

AN, CI, CV, LN, PN, PL, PT, SF, SO, VE and MX

Other types may be a range of subscripts of a subscripted name.

An example of a range using the auto name prefixes would be:

GET/VE3 , THRU , VE8

This would get the following items:

VE3 , VE4 , VE5 , VE6 , VE7 and VE8

An example of a range using a subscripted name would be:

GET/DAT1 (5 , THRU , 9)

This would get the following items:

DAT1 (5) , DAT1 (6) , DAT1 (7) , DAT1 (8) and DAT1 (9)

Ranges may be either ascending or descending. For example, the statements:

```
GET/SF2, THRU, SF8  
GET/SF8, THRU, SF2
```

would produce identical results.

It should be noted that while this command can be entered as shown, it can also be produced by using either one of the following on screen menu sequences.

File > Secondary Unibase > Get Geo

or

File > Secondary Unibase > Get Geo Thru

5.3.4 GET/ type

This statement gets all items from the unibase file that are of the specified type. Valid entries for type are:

ANOTE, POINT, VECTOR, POINT-VECTOR, LINE, PLANE, CIRCLE, CURVE, SOLID, SURF, PATERN, MATRIX, DATA, DRAFT, SYMBOL and PLACE or the synonym for each type.

Any combination of the above constructs may be strung together in a single GET statement.

```
GET/CI, CV, LN, DAT1(5, THRU, 9), SF2, THRU, SF8
```

Note:

- DRAFT retrieves all the drafting entities.
- All geometry associated with a drafting entity will be retrieved with the drafting entity if it does not already exist in the unibase.
- When retrieving geometry and drafting entities from a secondary unibase it is highly recommended that the geometry is retrieved first and then the drafting entities.
- SYMBOL retrieves all the master symbols without instances.
- PLACE retrieves all the instances and the corresponding master symbols.

5 ACCESSING UNIBASE FILES

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu sequence.

File > Secondary Unibase > Get Geo By Type

5.3.5 GET/[RENAME,] LAYER, n1 [[, THRU], nn] [. . .] \$
[,m [, THRU], mn] [, OFFSET, k]

This statement is used to retrieve geometries (*except symbols and instances*) from a list of specific layers or a range of layers. Also, the retrieved geometries can be renamed by specifying RENAME and the retrieved layer can be offset by specifying a “k” value.

Where:

RENAME	Can be specified to rename entities as they are retrieved.
LAYER	Specifies that the subsequent parameters will be a list of layers on which entities to be retrieved exist.
n1,nn,m,mn	Specifies the layer numbers to retrieve.
THRU	An optional parameters used to specify a range of layer between "n1" and "nn" or "m" and "mn"
OFFSET,k	Specifies the offset number for each layer retrieved.

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu sequence.

File > Secondary Unibase > Get Geo By Layer

5.3.6 GET/ ALL [, OFFSET, n]

This statement causes **NCL** to get every item (*except symbols and instances*) in the unibase file with option to offset all entity layer numbers by a scalar value “n”.

As a result of the different method of processing the GET statement, if an error occurs on an item somewhere in the GET statement, the items that appear in the

5 ACCESSING UNIBASE FILES

statement before the item in error will have already been copied from the Unibase file.

For example, if the item LN2 caused an error in the GET statement after opening the unibase file:

```
UBFN/unibase.ud  
GET/PT8,CI3,LN2,SF1
```

PT8 and CI3 would have been copied before it reported that it could not find LN2 in the Unibase file or LN2 was already defined in the program. In the cases where a naming method was used that specified multiple names, such as the THRU construct, and an item was not copied from the Unibase file because no item by that name existed, no error message will be generated.

The execution of a GET statement may be interrupted at any time by pressing the escape key (ESC for platforms running the GL driver or Cntrl-C on all platforms running the X or Motif driver). If this is done, a warning message is displayed. The user is cautioned that any items that have been retrieved before the statement was interrupted, have been copied and will not be removed as a result of aborting the command.

If **REFSYS** is “ON”, any item that is retrieved via the GET statement is transformed by the REFSYS matrix.

Example:

```
MX1=MX/XYROT,90  
REFSYS/MX1  
UBFN/part.u  
GET/ALL  
UBFN/CLOSE
```

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu sequence.

File > Secondary Unibase > Get All

5.3.7 GET/ RENAME, ...

This syntax is used to rename entities (*except symbols and instances*) as they are retrieved from the unibase.

5 ACCESSING UNIBASE FILES

Entities are renamed according to the current autonaming convention. The autonaming convention can be changed by using the [DEFNAM/....](#) command prior to the GET operation. For example:

DEFNAM/SF, NSF, INDEX

Would retrieve all surfaces and rename them to NSF(1), NSF(2), etc.

Symbols will be assigned the label MS with an integer suffix. For example, if a symbol with the label TSYM is already defined the new label for the import symbol will be renamed to MS1.

Instances will be assigned with the label SY with an integer suffix. For example, if an instance with the label TSY1 is already defined the new label for the import instance will be renamed to SY1.

It should be noted that while this command can be entered as shown, it can also be produced by using any one of the following on screen menu sequences.

File > Secondary Unibase > Get Geo Rename

or

File > Secondary Unibase > Get Geo Thru Rename

or

File > Secondary Unibase > Get Geo By Layer

or

File > Secondary Unibase > Get Geo By type

5.4

LOADU

The LOADU statement is used to load the entire contents of a unibase file into the current work session. The LOADU command replaces the current working environment with the contents of the specified unibase. Any entity defined prior to the LOADU command is removed. The LOADU command is generally used to load an existing part model (created via **NCL/IGES** or a **NCL** part program) for the purposes of machining or additional geometry definition or both. The LOADU command differs from the [UBFN - GET](#) combination in that the former replaces the currently active model and the later adds to the currently active model.

5 ACCESSING UNIBASE FILES

```
LOADU/unibase-id, [MM      ]  
                  INCLES  
                  SAME  
                  UNITS
```

Where:

- | | |
|------------|--|
| unibase-id | Is the name of the unibase file that is to be loaded. The filename may be specified either without any extension, with a “.”, with a “.u”, or a “.ud” (ASCII) extension. If none is specified, a “.u” will be assumed. No other filename extension is valid. |
| MM | The optional MM setting specifies that when the file is loaded NCL will be set to MM regardless of what units were stored in the original unibase-id. |
| INCLES | The INCLES setting specifies that when the file is loaded NCL will be set to INCLES regardless of what units were stored in the original unibase-id. |
| SAME | Specifies that NCL will be set to the UNITS specified in the original unibase-id. |
| UNITS | Specifies that NCL will use the current setting of UNITS, either from the UNITS command or the U_UNITS environmental variable. |

Example LOADU Statements

```
LOADU/PARTABC.  
LOADU/PARTXYZ.u  
LOADU/PARTX.ud  
LOADU/NEWPRT,MM  
LOADU/NEWPRT,INCLES  
LOADU/OLDPRT,SAME  
LOADU/NOWPRT,UNITS
```

5.5

PUT

The PUT statement is used to add entities from the currently active model to the unibase file specified in the last processed **UBFN** statement. The same entities supported by the **GET** command are also supported by the **PUT** command.

5 ACCESSING UNIBASE FILES

Example PUT Statements:

```
PUT/PT4  
PUT/DBPT34,AS,PT15  
PUT/LN  
PUT/A23*  
PUT/SF2,THRU,SF13  
PUT/SUBPT(33),THRU,SUBPT(95)  
PUT/ALL  
PUT/PT2,PT3,PT5,CV2,THRU,CV5,SF18,THRU,SF12  
PUT/PT,LN,SF,CIRCLE  
PUT/A*,B*,C*,PT
```

5.5.1 PUT/ alphanumeric-character-string*

This statement notifies **NCL** to store, in the Unibase file, items (*except symbols and instances*) in the program whose names start with the characters given in the alphanumeric-character-string. For example, the statement PUT/G589* would store in the Geometry Unibase file all items in the program that have a name starting with the characters G589 regardless of what type of entity it is.

5.5.2 PUT/ entity-id, AS, new-id

The entity-id is the name of the item in the program that is to be stored in the Unibase file. "AS" indicates that the item is to be renamed to a different name. The new-id is the name the item is to be known as in the Unibase file.

5.5.3 PUT/ entity-id [, THRU, entity-id]

The entity-id is the name of the item in the program that is to be stored in the Unibase file.

The THRU option causes **NCL** to store, in the Unibase file, a range of items from the program. The range of geometries may be specified using names that use the default autoname generation prefixes which are:

PT, VE, PV, LN, PN, PL, CI, CV, SF and MX or it may be a range of subscripts of a subscripted name.

Other types may be a range of subscripts of a subscripted name.

An example of a range using the autoname prefixes would be:

5 ACCESSING UNIBASE FILES

PUT/VE3 , THRU, VE8

This would store the following items:

VE3 , VE4 , VE5 , VE6 , VE7 and VE8

An example of a range using a subscripted name would be:

PUT/SVPT(5 , THRU , 9)

This would store the following items:

SVPT(5) , SVPT(6) , SVPT(7) , SVPT(8) and SVPT(9)

Ranges may be either ascending or descending. For example, the statements:

PUT/SF2 , THRU , SF8

PUT/SF8 , THRU , SF2

would produce identical results.

It should be noted that while this command can be entered as shown, it can also be produced by using either one of the following on screen menu sequences.

File > Secondary Unibase > Put Geo

or

File > Secondary Unibase > Put Geo Thru

5.5.4 PUT/ type

This statement causes **NCL** to store, in the Unibase file, all items in the program that are of the specified type. Valid entries for type are:

ANOTE, POINT, VECTOR, POINT-VECTOR, LINE, PLANE, CIRCLE, CURVE, SOLID, SURF, PATERN, MATRIX, DATA, DRAFT, SYMBOL and PLACE or the synonym for each type.

Any combination of the above constructs may be strung together in a single PUT statement.

PUT/CI , CV , LN , SVPT(5 , THRU , 9) , SF2 , THRU , SF8

5 ACCESSING UNIBASE FILES

Note:

- DRAFT stores all the drafting entities.
- All geometry associated with a drafting entity will be stored with the drafting entity if it does not already exist in the unibase.
- Any drafting entity in the secondary unibase will be deleted if any of its associated entities is replaced by a new geometric entity with the same label.
- When storing geometry and drafting entities from a secondary unibase it is highly recommended that the geometry is stored first and then the drafting entities.
- SYMBOL stores all the master symbols without instances.
- PLACE stores all the instances and the corresponding master symbols.

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu sequence.

File > Secondary Unibase > Put Geo By Type

5.5.5 PUT/ LAYER, n1 [[, THRU], nn] [. . .] [, m [, THRU], mn]

This statement is used to store geometries (*except symbols and instances*) from a list of specific layers or a range of layers.

Where:

LAYER	Specifies that the subsequent parameters will be a list of layer numbers on which entities to be stored exist.
n1,nn,m,mn	Specifies the layers numbers to be stored.
THRU	An optional parameters used to specify a range of layer between "n1" and "nn" or "m" and "mn"

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu sequence.

File > Secondary Unibase > Put Geo By Layer

5.5.6 PUT/ ALL

This statement causes **NCL** to put every item (*except symbols and instances*) in the Unibase file.

The PUT statement is processed differently than most **NCL** statements in that data is copied to the unibase file from the program's current internal data file as each item's name is encountered in the PUT statement. Most **NCL** statements are completely syntax checked before any processing is done. This is not the case with the PUT statement. As a result of the different method of processing the PUT statement, if an error occurs on an item somewhere in the PUT statement, the items that appear in the statement before the item in error will have already been copied to the Unibase file.

For example, if the item LN2 caused an error in the following statement:

```
PUT/PT8, CI3, LN2, SF1
```

PT8 and CI3 would have copied before it reported that it could not find LN2 in the program. In the cases where a naming method was used that specified multiple names, such as the THRU construct, and an item was not stored in the Unibase file because no item by that name existed, no error message will be generated.

The execution of a PUT statement may be interrupted at any time by pressing the escape key (ESC for platforms running the GL driver or Cntrl-C on all platforms running the X or Motif driver). If this is done, a warning message is displayed. The programmer is cautioned that any items that have been stored in the unibase file before the statement was interrupted, have been copied and will not be removed as a result of aborting the command.

If **REFSYS** is “ON”, any item that is inserted into a unibase file via the PUT statement is transformed by the REFSYS matrix.

Example:

```
MX1=MX/XYROT, 90
REFSYS/MX1
UBFN/part.u
PUT/ALL
UBFN/CLOSE
```

5 ACCESSING UNIBASE FILES

It should be noted that while this command can be entered as shown, it can also be produced by using the following on screen menu sequence.

File > Secondary Unibase > Put All

5.6 SAVEU

The SAVEU statement is used to save the current working model to a unibase file. The valid syntax construct for the SAVEU statement is

SAVEU/Unibase-id

Unibase-id is the name of the unibase file that is to be saved. The filename may be specified either without any extension, with a “.”, with a “.u”, or a “.ud” (ASCII) extension. If none is specified, a “.u” will be assumed. No other extension is valid.

Example SAVEU Statements

```
SAVEU/ PARTABC.  
SAVEU/ PARTXYZ.u  
SAVEU/ PARTX.ud
```

6 MOTION CONTROL STATEMENTS

This section deals with the CUTTER, TLAXIS, Point to Point, Continuous Path Motion generation, Automatic Motion Routine and related statements.

6.0 Introduction

This chapter divides into 5 sub-sections. They are:

1. **CUTTER and TLAXIS statements.**
 - a. Shows how to use the CUTTER statements to define a cutter for motion calculation; the CUTTER/DISPLY statements to define the image of the cutter/shank/holder for motion display and **NCL/IPV** verification purpose.
 - b. Shows how to control the tool axis with the TLAXIS statement.

2. Point to Point Motion Statements.

This sections shows all the point-to-point motion statements such as: GOTO, GO, etc. All these statements are usually used to position the cutter before any kind of continuos motion.

3. Continuous Path Motion Statements.

This sections shows all the continuous path motion statements such as: GOFWD, GORGT, TLLFT, TLON, etc.

4. Automatic Motion Routine Statements.

This sections shows all the Automatic Motion Routine such as POCKET, Waterline Roughing, Flowline Milling, etc.

5. Miscellaneous Motion Statements.

This section shows all the miscellaneous motion statements such as MULTAX, THICK, TRACUT, etc.

6 CUTTER AND TOOL AXIS STATEMENTS

Cutter and Tool Axis Statements

This chapter deals with the Cutter definitions for (motion calculation)/display purposes and the tool axis control statements.

6.1 CUTTER

The CUTTER statement is used to define the current cutter dimensions. A cutter must be defined BEFORE the motion statements are processed. If no cutter is defined, a default cutter with zero parameters will be assumed.

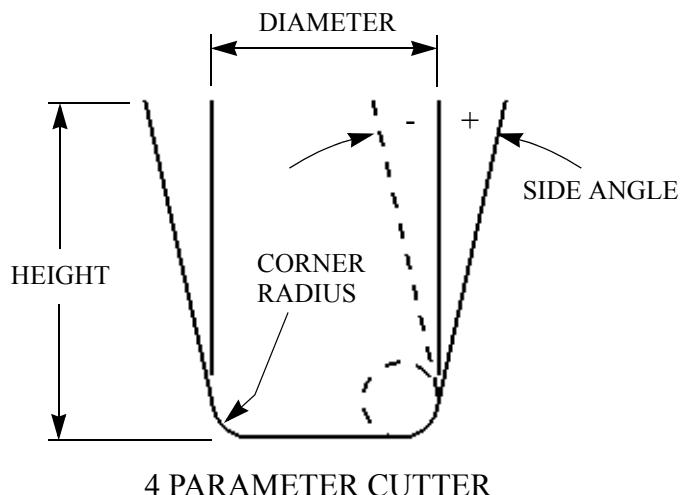
The CUTTER statement is also used to define the shank and holder image used for both motion display in **NCL** and **NCL/IPV**.

NCL supports three major type of cutters: Mill Style, Lathe Style and Blade Style.

6.1.1 Mill Style Cutter

This section describes how to define a Milling Style cutter for motion calculation, the corresponding images of cutter, shank and holder for motion display and verification in **NCL/IPV**.

6.1.1.1 CUTTER/ diameter [, corner-radius [, height [, side-angle]]]



This is the **NCL** default cutter definition and it is different from the APT 7 parameters CUTTER definition.

6 CUTTER AND TOOL AXIS STATEMENTS

The diameter parameter must be specified. This value does not give the diameter at the widest point of the cutter but rather it is the diameter of the cutter as if the bottom corner radius and the height were the only other parameters given with the height being at least as large as the corner radius.

That is, if the cutter were only described as a diameter, corner radius and height, the cutter would be cylindrical and the diameter dimension would be the diameter of the cylinder.

The corner-radius parameter is optional and if omitted, 0 is assumed.

The height parameter is also optional. If omitted, the value of the corner radius is used. The cutter height may be less than the corner radius to define a disk cutter. In this case, the draft angle specified by the fourth parameter must be 0.

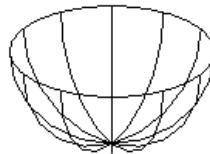
The side-angle parameter specifies the draft angle on the side of the cutter. This allows the definition of a "cone" or "bell" cutter. A positive value specifies a cone cutter, that is, one with the side angling away from the top of the cutter. A negative value specifies a bell cutter, that is, one with the side angling towards the top of the cutter. If omitted, 0 is assumed.

If a cone, bell cutter or a cutter with height less than corner radius is specified, the only valid drive surface types are SURF and PLANE unless the tool relationship with the drive surface is **TLON**.

The check surface relationship must be ON with all geometry types other than PLANE and SURF.

Example CUTTER Statements:

```
CUTTER/1  
CUTTER/2, 0  
CUTTER/1, .25, 5  
CUTTER/3, .75, 2.25, -20  
CUTTER/3, .75, 2.25, 20
```



CUTTER/2,1

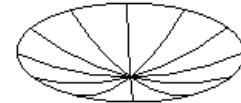


CUTTER/3,0.25

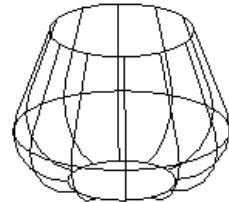
6 CUTTER AND TOOL AXIS STATEMENTS



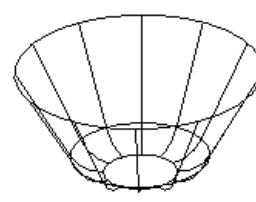
CUTTER/3



CUTTER/9.5,4.75,0.5



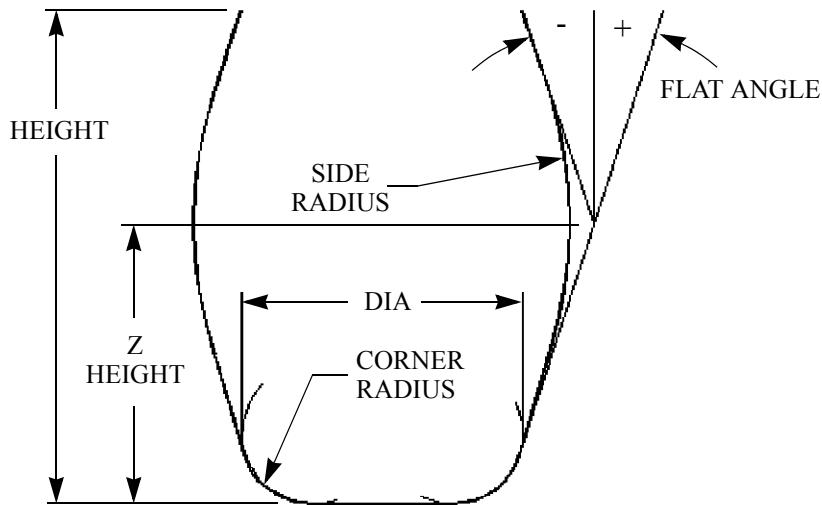
CUTTER/3,0.75,2.25,-20



CUTTER/3,0.75,2.25,30

6.1.1.2 CUTTER/ diameter, corner-radius, height, side-radius, Z-height \$

[, flat-angle]



6 PARAMETER CUTTER

A "barrel" cutter defines the diameter, bottom corner radius, height, side radius and optional flat at the top of the side radius. The first five (5) parameters are required in the definition of a "barrel" cutter.

The diameter, corner-radius and height are all the same for a barrel cutter as for a regular cutter as defined above.

The side-radius parameter specifies the radius of the side of the barrel cutter. This parameter is the one that defines the "barrel bulge" of the cutter.

6 CUTTER AND TOOL AXIS STATEMENTS

The Z-height parameter specifies the center point of the side radius.

This height is measured from the bottom of the cutter to the center point of the arc describing the side radius. Z-height may be a negative value.

The optional flat-angle parameter specifies the angle of the flat on the side of the barrel cutter. If this parameter is omitted, there is a flat of zero length on the side.

The flat extends from a tangency point at the top of the side radius to the specified height of the cutter. A positive angle defines a cutter with the flat angling away from the top of the cutter. A negative angle leans toward the center of the cutter.

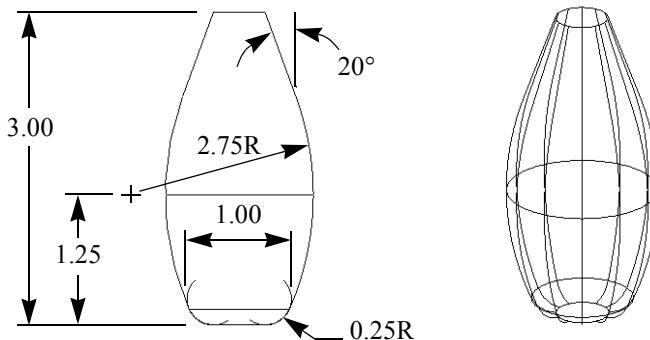
A barrel cutter may be used like any other **NCL** cutter with all **TLAXIS** modes. Note, however, that when using **TLAXIS/... FAN**, a non-zero side flat must be included in the cutter definition to align the final position of the fanning motion. Also, when using any of the TANTO, DS modes, the height of the tangent point must fall on the side flat.cutter. If omitted, 0 is assumed.

The only valid drive surface types for this type of cutters are SURF and PLANE unless the tool relationship with the drive surface is **TLON**.

The check surface relationship must be ON with all geometry types other than PLANE and SURF.

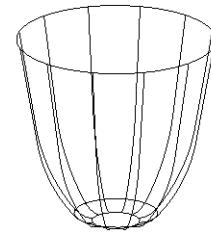
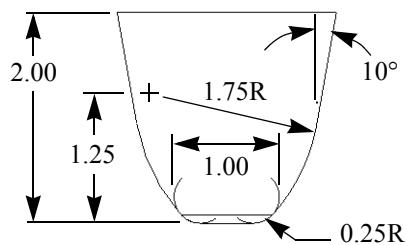
Example CUTTER Statements:

```
CUTTER/3,1,4,6,3,-10  
CUTTER/3,1,4,6,3,10  
CUTTER/3,1,4,6,3
```

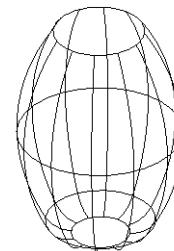
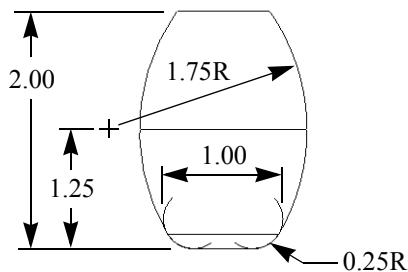


CUTTER/1,0.25,3,2.75,1.25,-20

6 CUTTER AND TOOL AXIS STATEMENTS



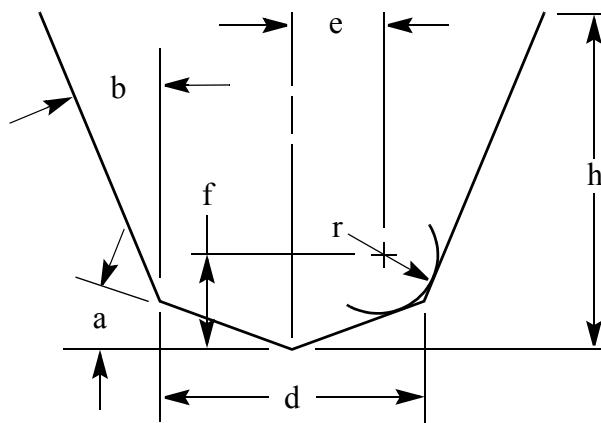
CUTTER/1,0.25,2,1.75,1.25,10



CUTTER/1,0.25,2,1.75,1

6.1.1.3 CUTTER/ d, r, e, f, a, b, h

This is the standard APT 7 parameter cutter definition.



6 CUTTER AND TOOL AXIS STATEMENTS

Where:

- d Cutter Diameter
- r Corner Radius
- e Horizontal distance from the center of corner radius to the center of cutter
- f Vertical distance from the center of corner radius to the bottom of cutter. ***Only a value equal to “r” will be supported.***
- a Angle in degrees that the bottom of the cutter makes with the horizontal bottom line. ***Only a value of “0” will be supported.***
- b Angle in degrees that the side of cutter makes with the vertical line
- h Height of the cutter

Note: Only NCL Style cutter will be supported.

- 6.1.1.4 CUTTER/ PSEUDO, dia, corner-radius, height, side-angle**
CUTTER/ PSEUDO, dia, corner-radius, height, side-radius, \$
Z-height [, flat-angle]

The *CUTTER/PSEUDO* command allows the user to define a pseudo cutter for driving geometry without affecting the current *CUTTER/DISPLAY* parameters. This allows the user to redefine the cutter used for motion calculations without having to restate the visual cutter parameters.

If the current displayed cutter is from the standard *CUTTER* command, then the pseudo cutter will actually be displayed, since the parameters will override the standard cutter parameters. Any defined tool shank and holder will then be displayed with the pseudo cutter.

- 6.1.1.5 CUTTER/ DISPLAY, dia, corner-radius, height, side-angle**

Defines a cutter shape to use for motion display rather than using the actual cutter definition. The *CUTTER/parameters* command will be used for motion calculations while the *CUTTER/DISPLAY* parameters will be used to define the cutter shape for display purposes only. All parameters are identical to the actual *CUTTER* statements

6 CUTTER AND TOOL AXIS STATEMENTS

6.1.1.6 CUTTER/ DISPLAY, dia, corner-radius, hgt, side-radius,\$

Z-height [, flat-angle]

Defines a cutter shape to use for motion display rather than using the actual cutter definition. The *CUTTER/parameters* command will be used for motion calculations while the *CUTTER/DISPLAY*, parameters will be used to define the cutter shape for display purposes only. All parameters are identical to the actual *CUTTER* statement.

6.1.1.7 CUTTER/ DISPLAY, surface

**cv [, pv]
pt-list
solid
[symlib,] symbol**

“surface” specifies a surface-of-revolution to be used for the cutter display.

“cv” specifies a two dimensional curve that represents the outline of the cutter. This curve is typically defined in the XY-Plane, with the cutter diameter being defined along the X-axis and the cutter height along the Y-axis. If the curve is defined in any other orientation, then a *point-vector* “pv” can be specified that defines the actual orientation of the curve.

“pt-list” specifies a cutter profile that is stored in the tool profile description file.

“solid” specifies a solid to be used for the cutter display. Only revolved type solid is allowed.

“symlib” defines the symbol library where the symbol can be found. The current directory will be the first to be searched for this library, and if it is not found there, then the system library directory will be searched. The library name must be specified by using *lower case* letters. The default library name is symlib.

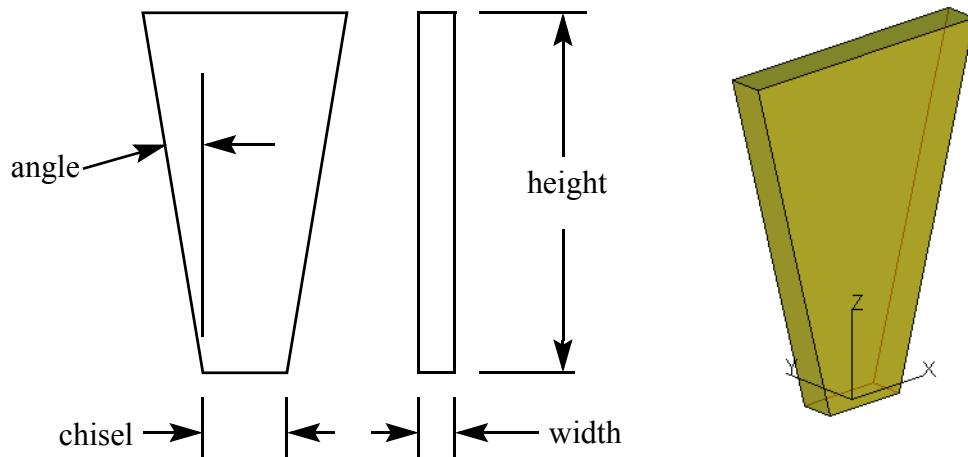
“symbol” defines the name of the **NCL/CADD** symbol to use as the displayed cutter representation. **NCL/IPV** requires there is one and only one *Revolved NSURF* type entity, or one and only one revolved type solid in this symbol. **NCL** does not has this requirement.

6.1.2 Blade Style Cutter

Used in association with Ultrasonic Blade cutting.

Note: The output tool end point for this type of cutter is always at the center point of the bottom of the cutter.

6.1.2.1 CUTTER/ BLADE, width, chisel, height, angle



6.1.2.2 CUTTER/ DISPLAY, curve pt-list solid [symlib ,] symbol

Defines a predetermined shape to be used with blade style cutters for motion display when a simple shape other than the calculated cutter shaped needs to be displayed.

“curve” specifies a two dimensional curve that represents the outline of the cutter. This curve must be defined in the XY-Plane.

“pt-list” specifies a cutter profile that is stored in the tool profile description file.

“solid” specifies a solid to be used for the cutter display.

6 CUTTER AND TOOL AXIS STATEMENTS

“symbol” specifies a symbol to be used to display the cutter along with an optional symbol library “symlib”.

The “curve” or the “pt-list” profile will be extruded by the defined width of the actual cutter and will be displayed in the correct orientation as defined by the tool axis and forward vectors.

6.1.3 Lathe Style Cutter

Defines a lathe style cutter to use for both cutter definition and motion display.

Note: The output tool end point for this type of cutter is always at the center point of the lower left corner of the displayed insert cutter.

6.1.3.1 CUTTER/ LATHE, radius, diameter, height \$

[, angle [, mount-angle]]

This command defines a lathe style insert cutter for both cutter definition and motion display.

Where:

radius = the radius at the tip of the lathe insert. This is the only parameter used when driving a lathe style cutter. The remaining parameters are used for display only. For example, the cutters defined with the following commands are considered the same by **NCL** when driving geometry.

CUTTER/.12,0,.1
CUTTER/LATHE,.06,0,.1

diameter = the diameter of the inscribed circle within the lathe insert.

height = defines the height of the tool. This tool will be displayed starting at Z=0, with the lower part of the tool displayed at Z = -height.

angle = defines the included angle of the nose tip. A value of 90 defines a square insert. A value of 60 defines a triangle insert. A value not equal to 90 or 60 defines a diamond insert. It is not used with circular inserts.

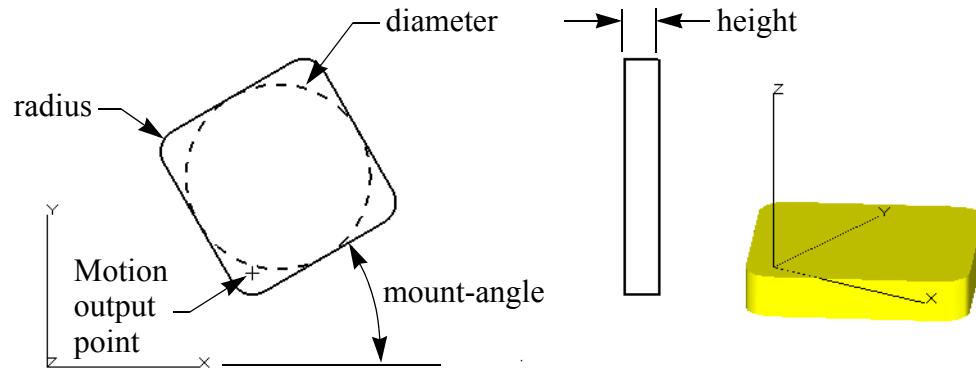
6 CUTTER AND TOOL AXIS STATEMENTS

mount-angle = defines the angle to rotate the tool between the X-axis and tool tip. A positive number rotates the tool in counter-clockwise direction. A negative number rotates the tool in clockwise direction.

The following types of lathe style insert cutters can be defined using this command.

6.1.3.1.1 Square Insert

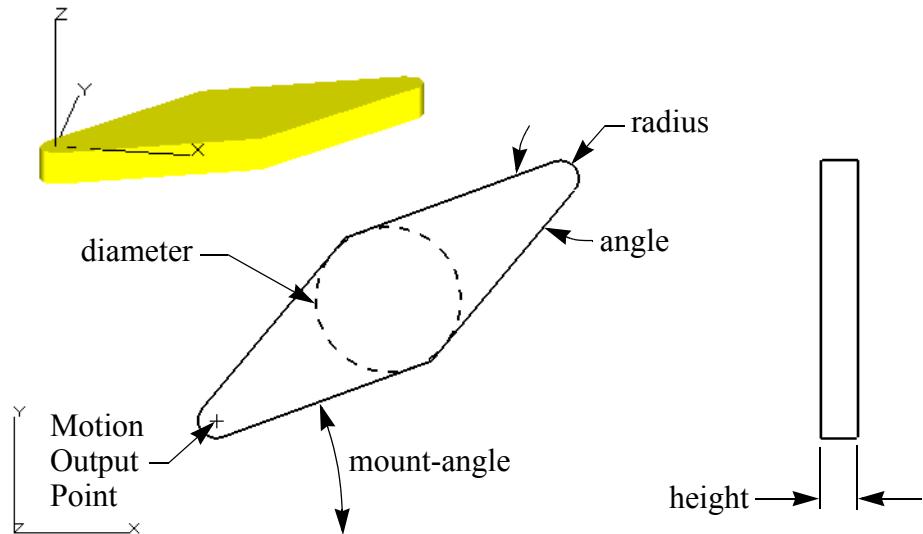
CUTTER/LATHE, radius, diameter, height, 90, mount_angle



6.1.3.1.2 Diamond Insert

CUTTER/LATHE, radius, diameter, height, angle, mount-angle

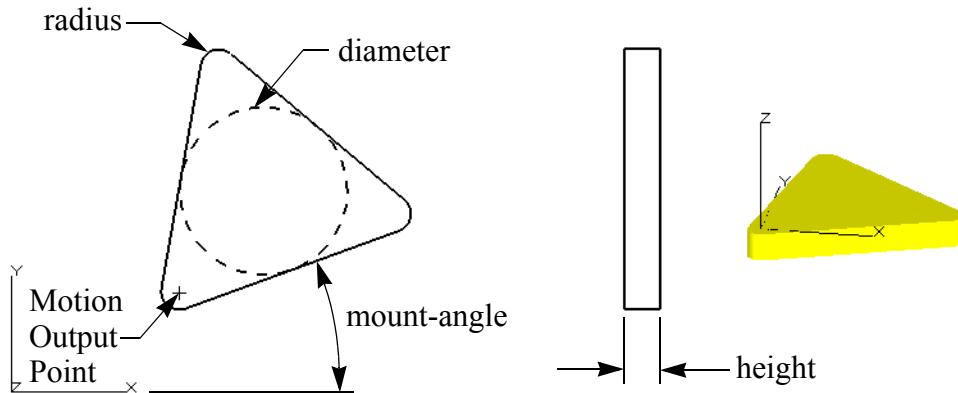
Note: "angle" cannot equal to 0, 60 or 90.



6 CUTTER AND TOOL AXIS STATEMENTS

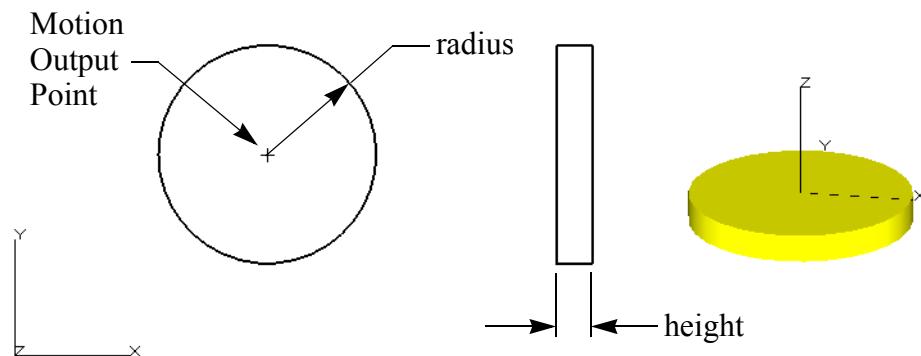
6.1.3.1.3 Triangle Insert

CUTTER/LATHE, radius, diameter, height, 60, mount_angle



6.1.3.1.4 Round Insert

CUTTER/LATHE, radius, 0, height



6.1.3.2 CUTTER/ LATHE, radius, width, height, 0, length

This command defines a lathe style grooving tool for both cutter definition and motion display.

Where:

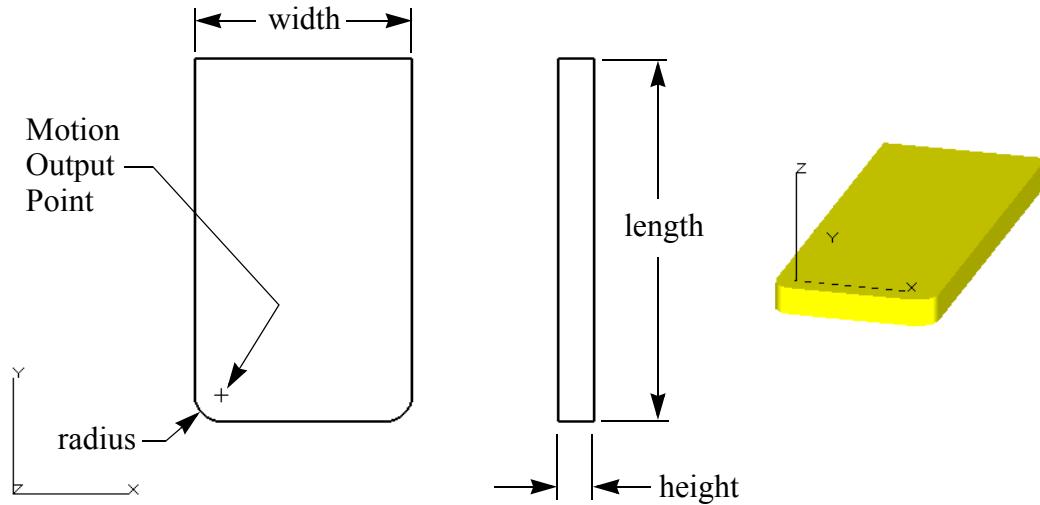
radius = the radius at the tip of the lathe insert. This is the only parameter used when driving a lathe style cutter. The

6 CUTTER AND TOOL AXIS STATEMENTS

remaining parameters are used for display only. For example, the cutters defined with the following commands are considered the same by **NCL** when driving geometry.

```
CUTTER/.12,0,.1  
CUTTER/LATHE,.06,0,.1
```

- width = the width along the X-axis for a grooving tool. Refer to the diagrams below.
- height = defines the height of the tool. This tool will be displayed starting at Z=0, with the lower part of the tool displayed at Z = -height.
- 0 = A required numerical parameter, it can be any value.
- length = the length of a grooving tool along the Y-axis.



6.1.4 CUTTER/ DISPLAY, SHANK, ...

Defines a cutter shank, which is used for display purposes in **NCL** and for display and clash detection purposes in **NCL/IPV**. The cutter shank is an extension of the defined cutter for Mill/Blade style cutters and a rectangular extrusion for Lathe style cutters.

6 CUTTER AND TOOL AXIS STATEMENTS

6.1.4.1 CUTTER/ DISPLAY, SHANK, diameter, height [, side-angle] \$

**[,ofs] [, CUTTER]
HOLDER**

This command defines a shank for the Mill/Blade Style cutter.

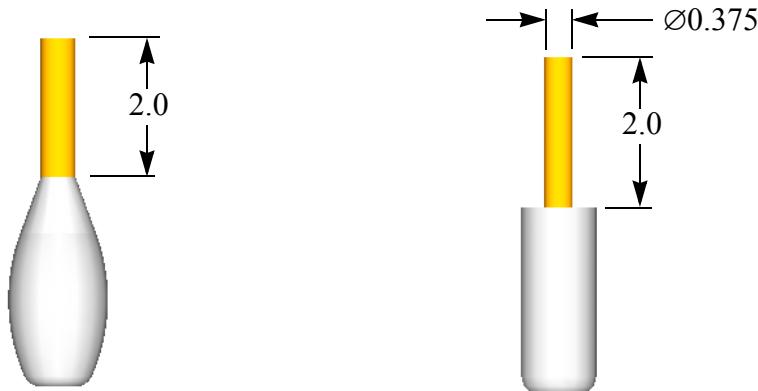
“diameter” defines the diameter of the shank. A value of zero will use the diameter of the cutter at the top of the tool.

“height” defines the height of the shank. The shank will be attached at the top of the defined cutter.

“side-angle” specifies the draft angle on the side of the shank. This allows the definition of a “cone” or “bell” shank. A positive value specifies a cone shank, that is, one with the side angling away from the top of the shank. A negative value specifies a bell shank, that is, one with the side angling towards the top of the shank. If omitted, 0 is assumed.

“ofs” specifies an offset value to apply along the height of shank when attaching it to the cutter. By default, the shank will be attached to the top of the cutter. The optional angle parameter must also be specified as a placeholder in the command if “ofs” is specified.

“CUTTER” specifies that the shank should be treated as a cutting part of the actual cutter for clash detection in **NCL/IPV**. “HOLDER” specifies that the shank is a non-cutting portion of the cutter, so therefore it will be treated as part of the holder in **NCL/IPV**. This parameter is not used during **NCL** motion display. The default shank clash detection in **NCL/IPV** is defined in the **NCL/IPV** Tool Modal form and can be changed in the Edit Tool List form.



CU/1,.25,3,2.75,1.25,-20
CU/DISPLAY,SHANK,0,2

CU/1,.25,2.5
CU/DISPLAY,SHANK,0.375,2

6 CUTTER AND TOOL AXIS STATEMENTS

6.1.4.2 CUTTER/ DISPLAY, SHANK, surface [, ofs] \$

**cv [, pv]
pt-list
solid
[symlib,] symbol**

**[, CUTTER]
HOLDER**

This command defines a shank for the Mill/Blade Style cutter.

“surface” specifies a surface-of-revolution to be used for the shank display.

“cv” specifies a two dimensional curve that represents the outline of the shank. This curve is typically defined in the XY-Plane, with the shank diameter being defined along the X-axis and the shank height along the Y-axis. If the curve is defined in any other orientation, then a *point-vector* “pv” can be specified that defines the actual orientation of the curve.

“pt-list” specifies a shank profile that is stored in the tool profile description file.

“solid” specifies a solid to be used for the shank display. Only revolved type solid is allowed.

“symlib” defines the symbol library where the symbol can be found. The current directory will be the first to be searched for this library, and if it is not found there, then the system library directory will be searched. The library name must be specified by using *lower case* letters. The default library name is symlib.

“symbol” defines the name of the **NCL/CADD** symbol to use as the displayed shank representation. **NCL/IPV** requires there is one and only one *Revolved NSURF* type entity, or one and only one revolved type solid entity in this symbol. **NCL** does not has this requirement.

“ofs” specifies an offset value to apply along the height of shank when attaching it to the cutter. By default, the shank will be attached to the top of the cutter. The optional angle parameter must also be specified as a placeholder in the command if “ofs” is specified.

“CUTTER” specifies that the shank should be treated as a cutting part of the actual cutter for clash detection in **NCL/IPV**. “HOLDER” specifies that the shank is a

6 CUTTER AND TOOL AXIS STATEMENTS

non-cutting portion of the cutter, so therefore it will be treated as part of the holder in **NCL/IPV**. This parameter is not used during **NCL** motion display. The default shank clash detection in **NCL/IPV** is defined in the **NCL/IPV** Tool Modal form and can be changed in the Edit Tool List form.

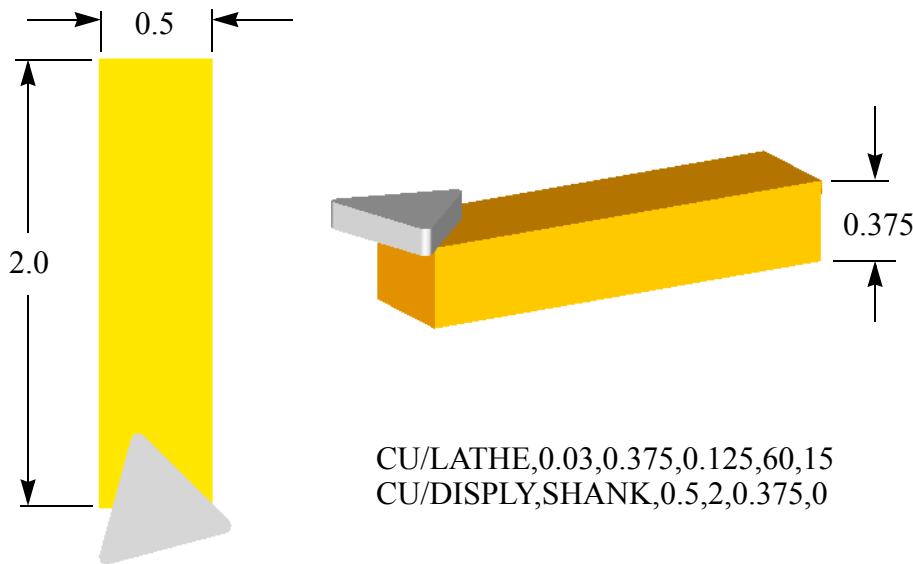
6.1.4.3 CUTTER/ DISPLAY, SHANK, width, length, depth, y-offset \$

**[, CUTTER]
HOLDER**

This command defines a rectangular shape shank for Lathe style cutter.

“width” defines the width of the rectangular shank along the X-axis. The “length” is defined along the Y-axis and the “depth” along the Z-axis. By default the cutter shank will be attached to the tool at the inscribed circle of the cutter in the Y-axis, at the bottom of the cutter in the Z-axis and attached at the center of the cutter along the X-axis. The “y-offset” parameter allows to position the shank up or down the Y-axis in relationship to this default attach point.

“CUTTER” specifies that the shank should be treated as a cutting part of the actual cutter for clash detection in **NCL/IPV**. “HOLDER” specifies that the shank is a non-cutting portion of the cutter, so therefore it will be treated as part of the holder in **NCL/IPV**. This parameter is not used during **NCL** motion display. The default shank clash detection in **NCL/IPV** is defined in the **NCL/IPV** Tool Modal form and can be changed in the Edit Tool List form.



6 CUTTER AND TOOL AXIS STATEMENTS

6.1.4.4 CUTTER/ DISPLAY, SHANK, curve [, x, y] \$

**pt-list
solid
[symlib,] symbol**

**[, OFFSET, zatt, zdep] [, CUTTER]
HOLDER**

This command define a lathe style shank.

“cv” specifies a two dimensional curve that represents the outline of the shank. The curve must be defined in the XY-plane.

“pt-list” specifies a lathe shank profile that is stored in the tool profile description file.

“solid” specifies a solid to be used for the shank display.

“symlib” defines the symbol library where the symbol can be found. The current directory will be the first to be searched for this library, and if it is not found there, then the system library directory will be searched. The library name must be specified by using *lower case* letters. The default library name is symlib.

“symbol” specifies the name of the **NCL/CADD** symbol to use as the lathe shank. If symbol is a composite, then this curve will be extruded by the “zatt” and “zdep” parameters to create three dimensional shape.

“x, y” defines the offset of the symbol shank in the X and Y directions. By default the shank will be attached at the inscribed circle of the cutter in the Y-axis, at the bottom of the cutter in the Z-axis and at the center of the cutter along the X-axis.

“OFFSET,zatt,zdep” specifies the starting Z-axis position and the depth of the shank part. The default starting position is at the bottom of the cutter. The default depth is the defined cutter height from the CUTTER/LATHE statement.

“CUTTER” specifies that the shank should be treated as a cutting part of the actual cutter for clash detection in **NCL/IPV**. “HOLDER” specifies that the shank is a non-cutting portion of the cutter, so therefore it will be treated as part of the holder in **NCL/IPV**. This parameter is not used during **NCL** motion display. The default shank clash detection in **NCL/IPV** is defined in the **NCL/IPV** Tool Modal form and can be changed in the Edit Tool List form.

Note:

6 CUTTER AND TOOL AXIS STATEMENTS

1. Revolved geometry (solids, surfaces, symbols, etc.) is allowed for lathe shank display. However, **NCL/IPV** only reports a clash when the revolved shank collides with stock or fixture, but the stock or fixture will not be cut.
2. A lathe shank can be defined in the Y-axis or X-axis direction. When using a solid or surface of revolution the axis of the revolved geometry determines the direction of the shank. When an extruded shank is defined, then if the geometry is longer in the X-axis than it is in the Y-axis, an X-axis direction shank will be assumed.

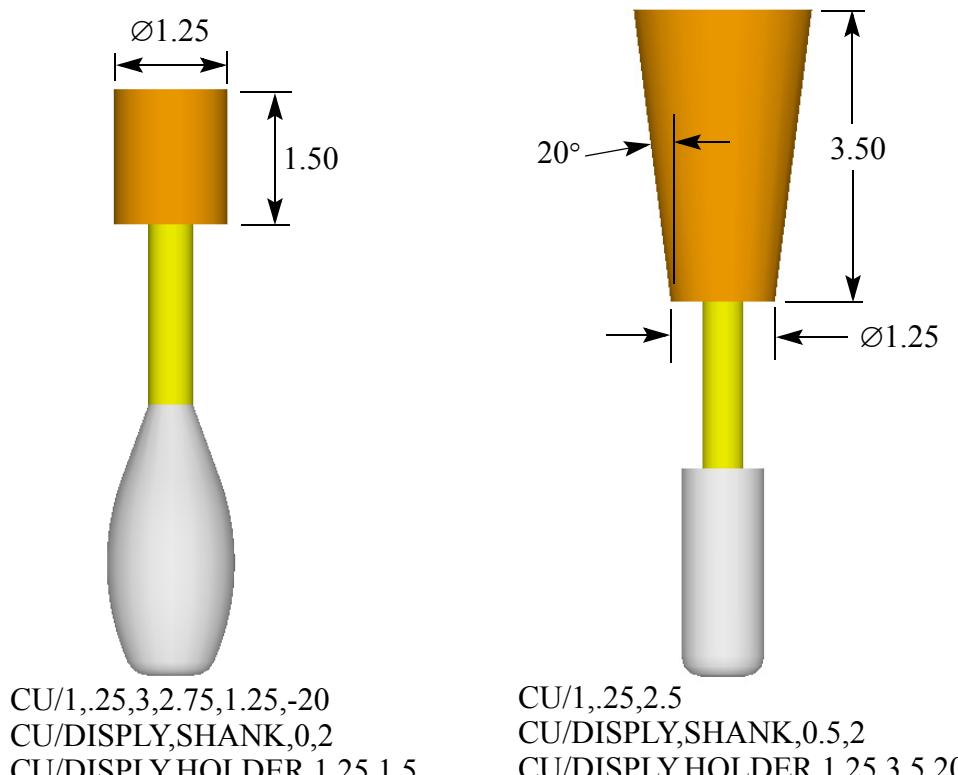
6.1.5 CUTTER/ DISPLAY, HOLDER, ...

Defines a cutter holder, which is used for display purposes in **NCL** and for display and clash detection purposes in **NCL/IPV**. The cutter holder is an extension of the defined cutter for Mill/Blade style cutters and a rectangular extrusion for Lathe style cutters.

6.1.5.1 CUTTER/ DISPLAY, HOLDER, diameter, height [, side-angle] \$

[,ofs]

This command defines a holder for the Mill/Blade Style cutter.



6 CUTTER AND TOOL AXIS STATEMENTS

“diameter” defines the diameter of the holder. A value of zero will use the diameter of the cutter at the top of the tool.

“height” defines the height of the holder. The holder will be attached at the top of the defined shank or top of the cutter if no shank is specified.

“side-angle” specifies the draft angle on the side of the holder. This allows the definition of a “cone” or “bell” holder. A positive value specifies a cone holder, that is, one with the side angling away from the top of the holder. A negative value specifies a bell holder, that is, one with the side angling towards the top of the holder. If omitted, 0 is assumed.

“ofs” specifies an offset value to apply along the height of holder when attaching it to the shank or cutter. By default, the shank will be attached to the top of the shank or the top of the cutter if no specified. The optional angle parameter must also be specified as a placeholder in the command if “ofs” is specified.

6.1.5.2 CUTTER/ DISPLAY, HOLDER, surface [, ofs]
cv [, pv]
pt-list
solid
[symlib,] symbol

This command defines a holder for the Mill/Blade Style cutter.

“surface” specifies a surface-of-revolution to be used for the holder display.

“cv” specifies a two dimensional curve that represents the outline of the holder. This curve is typically defined in the XY-Plane, with the holder diameter being defined along the X-axis and the holder height along the Y-axis. If the curve is defined in any other orientation, then a *point-vector* “pv” can be specified that defines the actual orientation of the curve.

“pt-list” specifies a holder profile that is stored in the tool profile description file.

“solid” specifies a solid to be used for the shank display. Only revolved type solid is allowed.

“symlib” defines the symbol library where the symbol can be found. The current directory will be the first to be searched for this library, and if it is not found there, then the system library directory will be searched. The library name must be specified by using *lower case* letters. The default library name is symlib.

6 CUTTER AND TOOL AXIS STATEMENTS

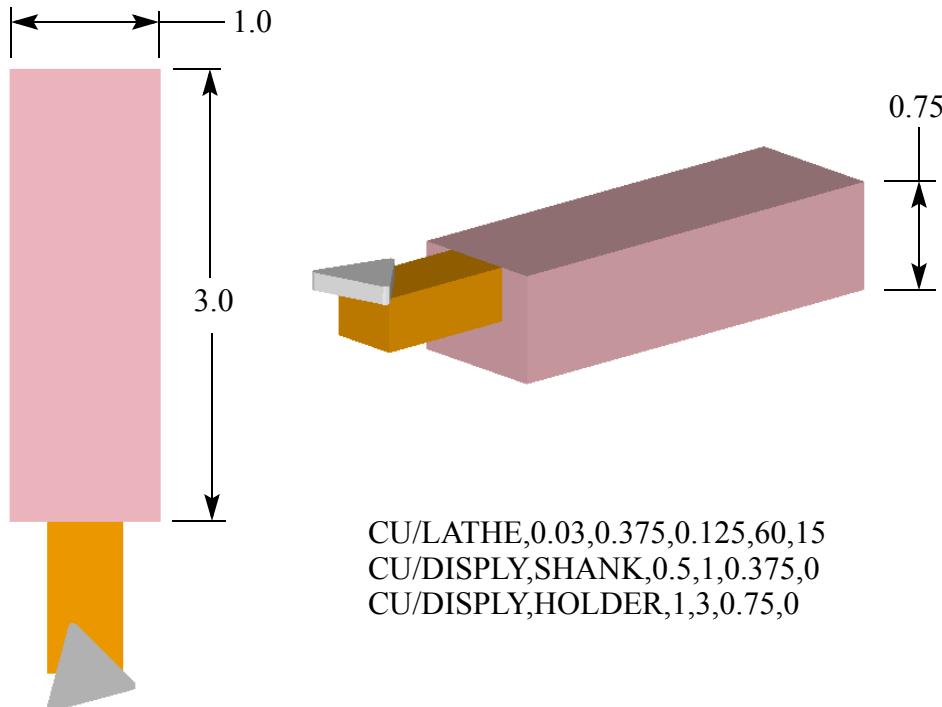
“symbol” defines the name of the **NCL/CADD** symbol to use as the displayed holder representation. **NCL/IPV** requires there is one and only one *Revolved NSURF* type entity, or a maximum of 24 solid type entities in this symbol. **NCL** does not have this requirement.

“ofs” specifies an offset value to apply along the height of holder when attaching it to the top of the shank or the top of the cutter if no shank is specified. By default, the holder will be attached to the top of the shank or the top of the cutter if no shank is specified. The optional angle parameter must also be specified as a placeholder in the command if “ofs” is specified.

6.1.5.3 CUTTER/ DISPLAY, HOLDER, width, length, depth, y-offset

This command defines a rectangular holder shank for Lathe style cutter.

“width” defines the width of the rectangular holder along the X-axis. The “length” is defined along the Y-axis and the “depth” along the Z-axis. The cutter holder will be attached to the top of the cutter/shank in both the Y-axis and Z-axis. The holder will be attached at the center of the shank along the X-axis, or the center of the cutter if no shank is specified along the X-axis. The “y-offset” parameter allows to position the holder up or down the Y-axis in relationship to this default attach point.



6 CUTTER AND TOOL AXIS STATEMENTS

6.1.5.4 CUTTER/ DISPLAY, HOLDER, curve [, x, y] \$

pt-list
solid
[symlib,] symbol

[, OFFSET, zatt, zdep]

This command defines a lathe style holder.

“cv” specifies a two dimensional curve that represents the outline of the holder. The curve must be defined in the XY-plane.

“pt-list” specifies a holder profile that is stored in the tool profile description file.

“solid” specifies a solid to be used for the shank display. Revolved solid type entity is not allowed.

“symlib” defines the symbol library where the symbol can be found. The current directory will be the first to be searched for this library, and if it is not found there, then the system library directory will be searched. The library name must be specified by using *lower case* letters. The default library name is symlib.

“symbol” specifies the name of the **NCL/CADD** symbol to use as the lathe holder. If symbol is a composite, then this curve will be extruded by the “zatt” and “zdep” parameters to create three dimensional shape. A maximum of 24 solid type entities can be specified inside the symbol definition.

“x, y” defines the offset of the symbol holder in the X and Y directions. By default the holder will be attached at the top of the cutter/shank in both the Y-axis and Z-axis. The holder will be attached at the center of the previous tool part along the X-axis, i.e. the shank if shank is specified, or the cutter if no shank is specified.

“OFFSET,zatt,zdep” specifies the starting Z-axis position and the depth of the holder part. The default starting position is at the bottom of the cutter. The default depth is the defined cutter height from the CUTTER/LATHE statement.

Note:

1. Revolved geometry (solids, surfaces, symbols, etc.) is allowed for lathe shank display. However, **NCL/IPV** only reports a clash when the revolved shank collides with stock or fixture, but the stock or fixture will not be cut.

6 CUTTER AND TOOL AXIS STATEMENTS

2. A lathe shank can be defined in the Y-axis or X-axis direction. When using a solid or surface of revolution the axis of the revolved geometry determines the direction of the shank. When an extruded shank is defined, then if the geometry is longer in the X-axis than it is in the Y-axis, an X-axis direction shank will be assumed.

6.1.6 CUTTER/ TOOL, [lib,] tool [, params]

Used to access the **NCL** Tool Library to load a tool.

Where:

lib = Name of the **NCL** tool library to load the tool from.

tool = The tool number to load.

params = Optional parameter list. These parameters will be used instead of the fields in the data base tool entry which are associated with a parameter. The first parameter in the list will replace the field referencing parameter #1, The second parameter #2, etc. The SAME qualifier can be used in the parameter list to use the field value from the data base instead of using a value from this command.

6.1.7 CUTTER/ READ, [lib,] tool [, params]

Used to access the **NCL** Tool Library to load a tool and output the tool library commands normally issued internally with the CUTTER/TOOL command.

Where:

lib = Name of the **NCL** tool library to load the tool from.

tool = The tool number to load.

params = Optional parameter list. These parameters will be used instead of the fields in the data base tool entry which are associated with a parameter. The first parameter in the list will replace the field referencing parameter #1, The second parameter #2, etc. The SAME qualifier can be used in the parameter list to use the field value from the data base instead of using a value from this command.

6 CUTTER AND TOOL AXIS STATEMENTS

Note:

- The input CUTTER/READ command is comment out as “\$\$ CUTTER/READ” statement.
- The tool library commands normally issued internally with the CUTTER/TOOL command are actually written to the source file and will be saved with the part program.
- A CUTTER/PROFIL command will be issued, when a Cutter Profile library is specified in the tool library, with the CUTTER/TOOL commands to denote which profile library is currently set as the default.

6.1.8 CUTTER/PROFIL,”lib-name”

This command specifies the default tool profile library.

“*lib-name*” is the name of the profile library file. **NCL** will search for this file in the local and then the system directory if the directory path is not given.

6.1.9 CUTTER/ DISPLAY, PART ALL

This command controls how many segments are used in the cutter display.

PART displays the cutter with the fewest number of segments. *ALL* displays the cutter with more segments along the length of the cutter for a more realistic looking cutter display.

6.1.10 CUTTER/ DISPLAY, MOVE, ON OFF

Turns on or off the dynamic display of the cutter. *ON* causes the cutter to move dynamically to each position without leaving a copy of the cutter shape at previous locations. *OFF* stamps the cutter shape at each position at the end of each move and when driving curved geometry at intervals specified by the **DRAFT/CUTTER** command.

6 CUTTER AND TOOL AXIS STATEMENTS

6.1.11 CUTTER/ DISPLAY, SHADE, ON [, CUTTER] OFF SHANK HOLDER

Support for shaded cutters, shanks or holders when running the OpenGL graphics driver.

Where:

- ON = Display shaded cutters. Moving cutters will have jumpy motion.
- OFF = Display wireframe cutters. This is the fastest display mode.
- CUTTER = Specifying this applies to the display of the cutter only. The display of the shank and holder will not be affected. This is the default if nothing is specified.
- SHANK = Specifying this applies to the display of the shank only. The display of the cutter and holder will not be affected.
- HOLDER = Specifying this applies to the display of the holder only. The display of the cutter and shank will not be affected.

6.1.12 Tool Profile Description File

The tool profile description file contains a list of profiles that are used to define the cutter, shank, and holder shapes. The environmental variable NCL_TOOL_DESC in the ncl.init file specifies the name of the tool description file.

NCL will look for this file in the current directory first and then in the NCL_TOOL directory. The file is formatted as follows:

```
[UNITS = Inch or MM]
tool-name
[class]
Point (x, y)
Arc (x, y, dir)
...
```

6 CUTTER AND TOOL AXIS STATEMENTS

The UNITS record states the units that the point data from this record forward is stored in and can be either Inch or MM. The UNITS record can be anywhere within the file and can appear multiple times, affecting only the Point and Arc records that follow it. Therefore, both inch and metric tools can be stored in the same file.

tool-name contains the name of the tool profile being defined. It can contain up to 20 characters. class specifies an optional class designator for the tool profile. The class text is used to filter the display of tool profiles in the Tool Display form. If a class is not specified, then it defaults to None.

The Point record defines an end point along the profile. Two consecutive points implicitly define a line segment. Defining two consecutive points that are the same will display a circular cross section for mill style cutters at this location of the cutter when it is displayed in wireframe mode.

The Arc record defines a circular arc. "xy" defines the center of the arc and "dir" defines the direction of the arc. "dir" can be either CLW or CCLW. A Point record that defines the start of the arc must immediately precede the Arc record and another Point record that defines the ending angle of the arc must immediately follow the Arc record. The second point does not have to lie on the actual arc.

Lathe style cutter definitions should define closed curves and not have duplicate consecutive points. Mill style cutters should only define the right hemisphere of the cutter shape, i.e. it should start at 0,0 and end at 0,n, where n is the height of the cutter.

The parentheses and commas in the Point and Arc records are optional and do not need to be included. For example, POINT 0 1 is an acceptable syntax.

Comment lines begin with the // characters and can be included anywhere in the file.

Multiple tool profiles can be defined in the file.

Following is a sample tool file that defines several mill cutters, lathe cutter, ultrasonic blade, holders and shanks.

Units = MM

MillingFormMM

Cutters

POINT (0, 0)

ARC (0, 9.525, CCLW)

6 CUTTER AND TOOL AXIS STATEMENTS

POINT (9.525, 9.525)
POINT (9.525, 25.4)
ARC (34.925, 25.4, CCLW)
POINT(34.925, 50.8)
ARC (34.925, 69.85, CCLW)
POINT(53.975, 69.85)
ARC (34.925, 69.85, CCLW)
POINT (34.925, 88.9)
POINT (19.05, 88.9)
POINT (19.05, 88.9)
POINT (19.05, 152.4)
POINT (0, 152.4)

Units = Inch

MillingHolderTool1

Holders

POINT (0, 0)
POINT (0.375, 0)
POINT (.375, .375)
POINT (.375, 1)
ARC (1.375,1, CLW)
POINT (1.375, 2)
ARC (1.375, 2.75, CCLW)
POINT (1.375, 3.5)
POINT (.75, 3.5)
POINT (.75, 6)
POINT (0, 6)

MillingShankTool1

Shanks

POINT(0, 0)
POINT (0.25, 0)
POINT (.375, .375)
POINT (.375, 1)
ARC (1.375,1 ,CLW)
POINT (1.375, 2)
ARC (1.375,2.75, CCLW)
POINT (1.375, 3.5)
POINT (.75, 3.5)
POINT (.75, 6)
POINT (0, 6)

6 CUTTER AND TOOL AXIS STATEMENTS

MillingFormTool1

Cutters

```
POINT (0, 0)
ARC (0, .375, CCLW)
POINT (.375, .375)
POINT (.375, 1)
ARC (1.375, 1, CLW)
POINT (1.375, 2)
ARC (1.375, 2.75, CCLW)
POINT (1.375, 3.5)
POINT (.75, 3.5)
POINT (.75, 6)
POINT (0, 6)
```

FaceMill1

Cutters

```
POINT (0, 0)
POINT (2, 0)
ARC (2, .25, CCLW)
POINT (2.25, .25)
ARC (2.25, CCLW)
POINT (2, .5)
POINT (0, .5)
```

LatheBit1

Cutters

```
POINT (0, 0)
ARC (0, .1, CCLW)
POINT (.1, .1)
POINT (.3, .5)
POINT (0, .35)
POINT (-.3, .5)
POINT (-.1, .1)
ARC (0, .1, CCLW)
POINT (0, 0)
```

LatheTurret1

Holders

```
POINT (0, 0)
POINT (.5, 0)
POINT (1.207, .707)
POINT (1.207, 1.414)
POINT (.5, 2.121)
```

6 CUTTER AND TOOL AXIS STATEMENTS

```
POINT (-.5, 2.121)
POINT (-1.207, 1.414)
POINT(-1.207, .707)
POINT (-.5, 0)
POINT (0, 0)
```

BladeCutter1

Cutters

```
POINT (0, 0)
POINT (.375, .5)
ARC (.625, .5, CLW)
POINT (.625, .75)
POINT (.625, .90)
POINT (.05, .90)
POINT (.05, 1.4)
ARC (0, 1.4, CCLW)
POINT (-.05, 1.4)
POINT (-.05, .90)
POINT (-.1, .90)
POINT (0, 0)
```

6.2 TLAXIS

The TLAXIS statement is used to specify the tool (cutter) angle and position in relation to the part surface, drive surface and check surface. Any TLAXIS statement may contain an optional Modify clause. The optional modify clause is used to offset the tool position and angle from the calculated position as the tool is driven along the control surfaces. It is important to note that the calculation takes place after the tool has been properly applied to the control surfaces. For example, if a modify clause was specified that would cause the tool to be offset in the up direction (along the tool axis) the tool would no longer be in contact with the control surfaces. The modify clause is generally used with the [GENPTS](#) command for the purpose of constructing geometry and is normally used within a [DNTCUT-DNTCUT/NM](#) sequence. If the modify clause is to be used for generating actual output data it is recommend that the cutter be positioned to the start of the cut within a DNTCUT-DNTCUT/NM sequence before driving the control surfaces. This is because the modify clause is not applied to [GO](#), [GODLTA](#), [GOTO](#) and [CUT](#) statements which are usually used for initial positioning. The valid syntax for the optional modify clauses are:

```
MODIFY [,right[,forward[,up[,angle-right
[,angle-forward]]]]]
```

6 CUTTER AND TOOL AXIS STATEMENTS

Where:

- | | |
|---------------|--|
| Right | - specifies the amount of offset to the right. |
| Forward | - specifies the amount of offset in the forward direction. |
| Up | - specifies the amount of the offset in the up direction. |
| Angle right | - specifies the angle of tilt to the right. |
| Angle forward | - specifies the angle of tilt in the forward direction. |

Certain TLAXIS statements may contain an optional PERPTO vector clause.

This clause is used in 4-axis machining applications and allows the fifth axis to be locked out. The valid syntax construct for the optional PERPTO vector clause is:

PERPTO, vector

The calculated tool axis vector will always be perpendicular to the vector specified by vector.

TLAXIS statements that result in a fixed tool axis ([6.2.1](#), [6.2.2](#) and [6.2.3](#)) may have the vocabulary word NORMAL added to the end of the statement before the optional modify-clause. This means that the tool will be held in the specified fixed tool axis as the tool end points are calculated. Then, instead of outputting the specified fixed tool axis, a new tool axis is generated at each tool end point that is a vector normal to the part surface at each point. In order for tool axis modes that cause varying tool axis vectors to generate and output the changing tool axis vectors, [MULTAX](#) must first be specified as "on".

Example TLAXIS Statements:

```
TLAXIS/0,0,1
TLAXIS/VE1,MODIFY,.5,0,15
TLAXIS/SAME
TLAXIS/NORMAL,PS,MODIFY,0,1,0,0,2
TLAXIS/NORMAL,PS,PERPTO,VE1
TLAXIS/ATANGL,15,PS
TLAXIS/TANTO,DS,1.25
TLAXIS/TANTO,DS,1,FAN
TLAXIS/TANTO,DS,.5,PERPTO,VE1
TA/TT,DS,1,PARELM
```

6 CUTTER AND TOOL AXIS STATEMENTS

```
TLAXIS/COMBIN, .75, .5
TA/COMBIN, .75, PARELM, .35
TLAXIS/COMBIN, .75, .5, 1.25
TA/COMBIN, .75, PARELM, .35
TA/COMBIN, .75, PARELM, .35
TA/COMBIN, .75, PARELM, .35, .625
TA/TT, DS, 1, FAN, RIGHT, 10, FWD, 5
TA/RIGHT, 5, FWD, -10
```

6.2.1 TLAXIS/ I, J, K [, NORMAL] [, modify-clause]

The I, J and K coordinates describe the vector which becomes the current tool axis. This statement invokes TLAXIS/ SAME.

6.2.2 TLAXIS/ vector [, NORMAL] [, modify-clause] point-vector

The vector or point-vector becomes the current tool axis. This statement invokes TLAXIS/ SAME

6.2.3 TLAXIS/ 1 SAME [, NORMAL] [, modify-clause]

This construct specifies that the tool axis is to remain at its current value until changed by another TLAXIS statement.

6.2.4 TLAXIS/ NORMAL, PS [, surface] \$ NORMPS plane

```
[ , PERPTO, [ LAST, ] vector      ][ , modify-clause ]
                                         point-vector
```

This construct specifies that the tool axis is always to remain normal to the part surface. If the optional PERPTO vector or point-vector is specified, the tool will also remain perpendicular to that vector.

If the optional surface or plane is specified, the tool axis will remain normal to this entity (the tool axis control surface) instead of the actual part surface.

6 CUTTER AND TOOL AXIS STATEMENTS

"LAST" allows the tool axis modifiers "FWD", "RIGHT", "GUIDE" and "GOUGCK" to be used while still maintaining the perpendicularity to the specified vector. If "LAST" is not specified, the output tool axis would not repeat the perpendicular vector if a tool axis modifier is specified with the perpendicular vector.

**6.2.5 TLAXIS/ ATANGL, angle, PS [, surface] [, CLDIST, dist] \$
plane**

[, CONTCT] [, PERPTO,[LAST,]vector] \$
point-vector

[, modify-clause]

This construct specifies that the tool axis is to remain tilted the number of degrees specified by "angle" in the forward direction from a vector normal to the part surface.

If the optional surface or plane is specified, this entity (the tool axis control surface) will be used to control the tool axis instead of the actual part surface.

If the optional [, CLDIST, distance] is specified, the tool will be checked at each output point to ensure the heel or back edge of the tool is kept the "distance" away from the part surface. This tool axis mode will cause **NCL** to adjust the tilt of the tool to keep the "heel" from the part surface by approximately this "distance" amount.

If the optional [, CONTCT] is specified, the tool contact point will maintain an ON condition to the Drive Surface, Part Surface and Check Surface thus allowing the tool end point to "float" as the Part Surface normal changes.

The area of the part that is actually cut is the intersection of the Drive Surface and Part Surface, thereby leaving smooth and even scallops when "kellering" a part with wavy surfaces. **TLLFT**, **TLRGT**, TO and PAST conditions are ignored when using this option. This option is used primarily for 4 and 5-axis machining of airfoil type surfaces. If the optional [, PERPTO, vector] is specified, the tool will also remain perpendicular to the vector. This is useful for 4-axis contouring situations.

"LAST" allows the tool axis modifiers "FWD", "RIGHT", "GUIDE" and "GOUGCK" to be used while still maintaining the perpendicularity to the specified vector. If "LAST" is not specified, the output tool axis would not repeat the

6 CUTTER AND TOOL AXIS STATEMENTS

perpendicular vector if a tool axis modifier is specified with the perpendicular vector.

This construct specifies that the tool axis is to be maintained tangent to the drive surface at a height specified by height and perpendicular to the part surface.

If the optional surface or plane is specified, the tool axis will remain perpendicular to this entity (the tool axis control surface) instead of the actual part surface.

If the optional perpto-vector is specified, the tool axis will also remain perpendicular to the specified vector.

“LAST” allows the tool axis modifiers “FWD”, “RIGHT”, “GUIDE” and “GOUGCK” to be used while still maintaining the perpendicularity to the specified vector. If “LAST” is not specified, the output tool axis would not respect the perpendicular vector if a tool axis modifier is specified with the perpendicular vector.

6.2.7 TLAXIS/ TANTO, DS, height, FAN [, CENTER, OFF] \$
ON
AUTO

[, SMOOTH [, d, r]] [, modify-clause]

This construct is similar to TLAXIS/ TANTO, DS, height except that the tool will approach the check surface with a fanning motion.

At the end of a pass, the tool will be tangent to the check surface at the specified height. As the tool iterates towards the check surface, the maximum angle of change between points is governed by the **MAXANG** statement.

"CENTER,OFF" specifies the fanning motion is calculated relative to the tool end. This is the default condition.

6 CUTTER AND TOOL AXIS STATEMENTS

"CENTER,ON" specifies the fanning motion is calculated relative to the center of the tool ring (the ball center for a ball-nose cutter), instead of the tool end. For most cases this results in a motion which may have numerically different steps but is essentially the same fanning motion. In some cases, this results in a good motion while the "tool end" calculation fails.

"CENTER,AUTO" specifies the motion calculation will change to the "Center of the tool ring" method if the "tool end" method is likely to result in error.

"CENTER,..." has no effect for a flat bottom cutter with no corner radius.

“SMOOTH [,d,r]”:

- This allows an optional smooth interpolation between the starting and ending tool axis vectors.
- “d” is a positive integer, specifying the degree of the interpolating polynomial (used to calculate the transition between tool axis modes). Specifying d=0 or d=1 disables smooth interpolation and uses the standard fanning algorithm.
- “r” is a positive real number no more than 1, specifying how steep the interpolating polynomial is in the middle of the fanning or interpolating motion.
- The “d” and “r” parameters are intended to allow the user to fine tune the way the tool axis changes with the SMOOTH modifier. Making the degree “d” higher leads to a more gradual transition between the fanning (interpolating) motion and the preceding and following motions. A higher rate “r” makes for an extra smooth transition, but a steeper change in the middle of the fanning (interpolating) motion.
- The default values are d=5, r=1, which creates a nice transition curve.

6.2.8 TLAXIS/ TANTO, DS, height, PARELM [, modify-clause]

This construct causes the tool axis to be maintained parallel to the iso-parametric lines of a surface being used as the drive surface. The cutter will contact the surface at the distance up the cutter specified by height. **NCL** will align the tool axis with the iso-parametric (U or V) lines which are closest to the current tool axis position. If this is not the desired direction, the [REDEF/ sf, PARELM, n](#) command may be used to indicate the desired PARELM direction.

The following statement may be added to set the PARELM mode back to automatic if previously modified by a REDEF command:

```
REDEF/surface, PARELM, -1
```

6 CUTTER AND TOOL AXIS STATEMENTS

[PS, surface,] leave-dist [, approach-dist]
plane

[, SMOOTH [, d, r]] [, modify-clause]

This command COMBINES the functionality of "FAN" and "TANTO, DS".

The format is similar to "TLAXIS/ TANTO, DS, height". The difference is that the tool will FAN away from the previous drive surface as it starts the move. It will generate this FAN type of motion until it reaches an internally calculated plane.

This plane is defined by calculating where the tool would be if it was driven in the "native" mode to a position **THICK** a distance of "leave-dist" from the previous cutter position (native mode would be TANTO the drive surface).

The plane would be constructed using the tool end point, a point at the top center of the tool at this driven to position and normal to the forward direction.

At that point, the tool will have been fanned approximately normal to the part surface and TANTO the drive surface. It will then maintain this TANTO the drive surface tool axis as it moves along the part/drive geometry until it reaches a point where the tool is THICK to the check surface a distance of "approach-dist." At this point, the tool will FAN into the check surface to complete the move. If the "approach-dist" is not specified, the value of "leave-dist" is used. If the overall distance between the current tool position and the next check surface is smaller than the sum of "leave-dist" and "approach-dist" then the entire move will be made in the FAN mode.

If the optional surface or plane is specified, this entity (the tool axis control surface) will be used to control the tool axis instead of the actual part surface.

"CENTER,OFF" specifies the fanning motion is calculated relative to the tool end. This is the default condition.

"CENTER,ON" specifies the fanning motion is calculated relative to the center of the tool ring (the ball center for a ball-nose cutter), instead of the tool end. For most cases this results in a motion which may have numerically different steps but

6 CUTTER AND TOOL AXIS STATEMENTS

is essentially the same fanning motion. In some cases, this results in a good motion while the "tool end" calculation fails.

"CENTER,AUTO" specifies the motion calculation will change to the "Center of the tool ring" method if the "tool end" method is likely to result in error.

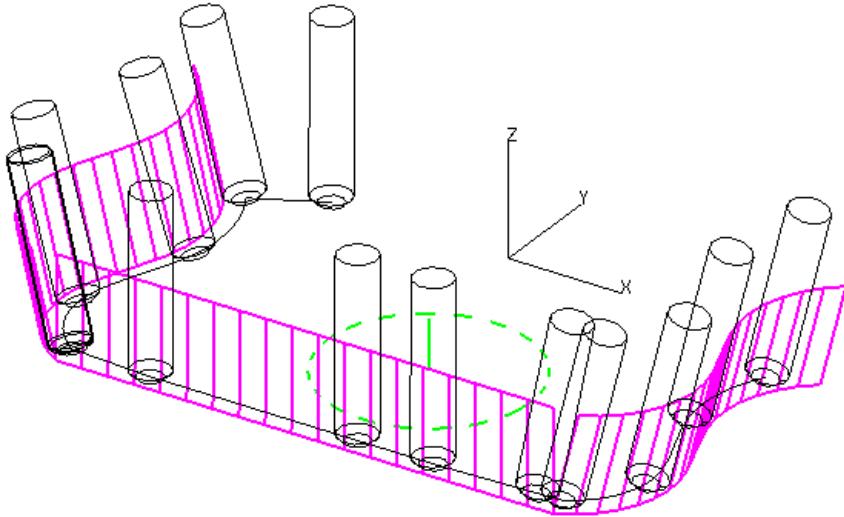
"CENTER,..." has no effect for a flat bottom cutter with no corner radius.

“SMOOTH [,d,r]”:

- This allows an optional smooth interpolation between the starting and ending tool axis vectors.
- “d” is a positive integer, specifying the degree of the interpolating polynomial (used to calculate the transition between tool axis modes). Specifying d=0 or d=1 disables smooth interpolation and uses the standard fanning algorithm.
- “r” is a positive real number no more than 1, specifying how steep the interpolating polynomial is in the middle of the fanning or interpolating motion.
- The “d” and “r” parameters are intended to allow the user to fine tune the way the tool axis changes with the SMOOTH modifier. Making the degree “d” higher leads to a more gradual transition between the fanning (interpolating) motion and the preceding and following motions. A higher rate “r” makes for an extra smooth transition, but a steeper change in the middle of the fanning (interpolating) motion.
- The default values are d=5, r=1, which creates a nice transition curve.

The purpose of using this TLAXIS mode, is to allow an easier transition when going from a FAN to a TANTO, DS mode or vice versa. See the figure on next page for the resultant tool motion.

6 CUTTER AND TOOL AXIS STATEMENTS



**6.2.10 TLAXIS/ COMBIN, height, PARELM, [CENTER, OFF ,] \$
ON
AUTO**

leave-dist [, approach-dist][, modify-clause]

This format is the same as the previous format except that the tool will remain parallel to the rulings of the drive surface instead of TANTO the drive surface during the middle portion of the move when it is not fanning away from the previous drive surface and towards the check surface.

The purpose of using this TLAXIS mode, is to allow an easier transition when going from a FAN or TANTO, DS mode to a PARELM mode or vice versa. If the overall distance between the current tool position and the next check surface is smaller than the sum of "leave-dist" and "approach-dist" then the entire move will be made in FAN mode.

6.2.11 TLAXIS/ [. . . ,] [RIGHT, right-angle] [[,] FWD, fwd-angle]

In addition to the modify clause all variable tool axis modes may have a modification applied to them by adding the parameters shown in the above syntax.

6 CUTTER AND TOOL AXIS STATEMENTS

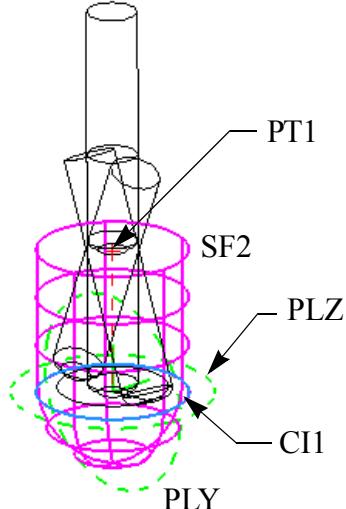
Where RIGHT will cause the calculated tool axis to be tilted right by the specified right-angle (a negative angle will cause a tilt to the left) and FWD by fwd-angle (a negative angle will cause a backward tilt).

This parameter may be added to the end of any variable tool axis command or may be specified in a statement by itself in which case the currently active tool axis mode will be modified

6.2.12 TLAXIS/ THRU, point

This statement will cause the tool axis to always pass through a specified point as shown in the following illustration on next page.

```
FROM/PT1  
MULTAX  
RAPID  
GOTO/(POINT/CENTER,CI1)  
TLAXIS/THRU,PT1  
INDIRV/1,0,0  
GO/TO,SF2,PLZ  
TLLFT  
GOLFT/SF2,ON,2,INTOF,PLY  
RAPID  
GOTO/PT1,0,0,1
```



6 CUTTER AND TOOL AXIS STATEMENTS

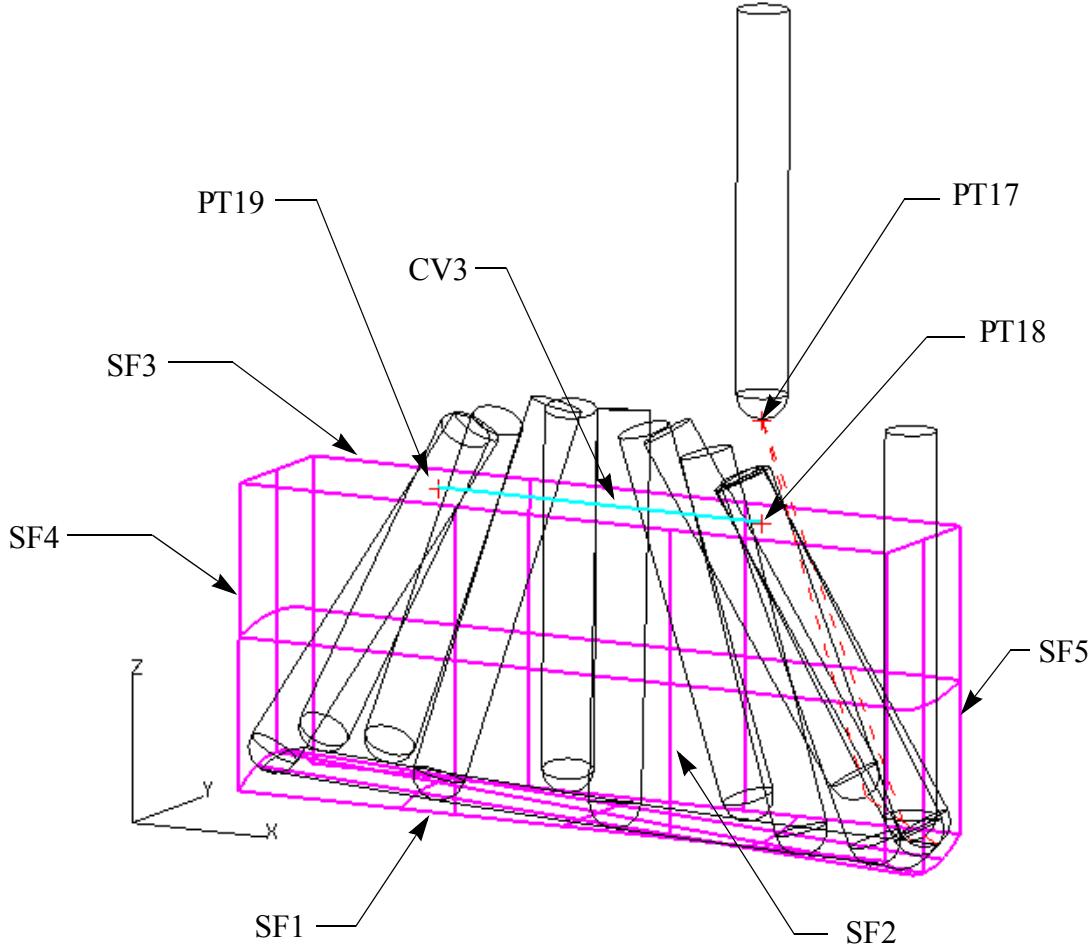
6.2.13 TLAXIS/ THRU, curve [, dist]

This statement will cause the tool axis to always pass through a specified curve.

Where curve (any wireframe entities - line, arc, etc.) is the curve the tool axis will always pass through. The tool axis will pass through the closest end point of the specified curve at the start of a move and will move proportional along the curve during the move until it passes through the other end point of the curve at the end of the move. *dist* is an optional distance representing the approximate length of the move and should be used if **NCL** gets an error when the length of the move is much greater than the initial distance of the cutter to the check surface.

The following group of commands and illustration on next page show how this statement may be used in a part program:

```
MULTAX
TA/0,0,1
FROM/(PT17=PT/PT18,0,0,1)
TH/.1,.1,.1
RP,GO/SF3,SF1,SF5
TH/0
TA/THRU,CV3
RP,GO/SF3,SF1,SF5
IV/-1,0,0
TL,GF/SF3,SF4
TA/THRU,PT19
GL/SF4,SF2
TA/THRU,CV3
GL/SF2,SF5
TA/THRU,PT18
GL/SF5,SF3
RP,GD/- .5,- .5,.5
RP,GT/PT17
```



**6.2.14 TLAXIS/ [. . . ,] GUIDE, curve, [, CONTCT] [, TLLFT] \$
 OFFSET TLRGT
 TLON**

[, thick]

All variable tool axis modes may be modified to maintain a relationship with a specified GUIDE curve.

Where in the above syntax:

- | | |
|-------|--|
| curve | - Is a valid curve class entity (curve, spline, circle, etc.) that will modify the current tool axis mode according to the specified parameters. |
|-------|--|

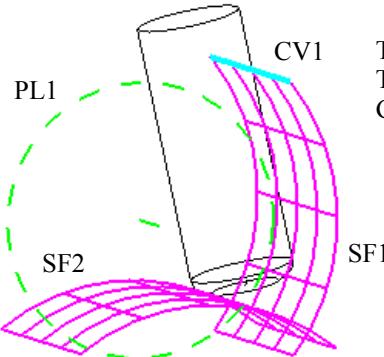
6 CUTTER AND TOOL AXIS STATEMENTS

- | | |
|--------|---|
| CONTCT | - Specifies that the tool should always be in contact with the GUIDE curve. |
| OFFSET | - Specifies that the tool will only be in contact with the GUIDE curve when the current tool axis would cause the tool to violate the GUIDE curve. If TLON is specified, CONTCT is assumed otherwise the default is OFFSET. |
| TLLFT | Specifies the side of the GUIDE curve the cutter is to contact. |
| TLRGT | - If none of these parameters are given then the current tool |
| TLON | condition is used. |
| thick | - Specifies an optional thickness to apply to the GUIDE curve. If no value is given, the current drive surface thick is used. |

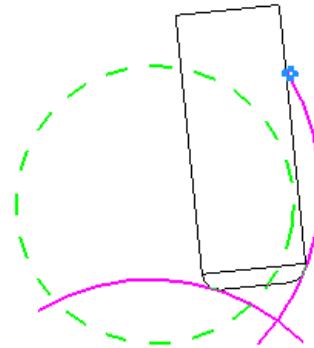
GUIDE curve parameters may be added to the end of any tool axis command or specified in a statement by itself in which case the currently active tool axis mode is modified. The GUIDE curve feature is automatically deactivated once another tool axis command is given.

Figures on next page illustrate the different usage of a guide curve in position mode.

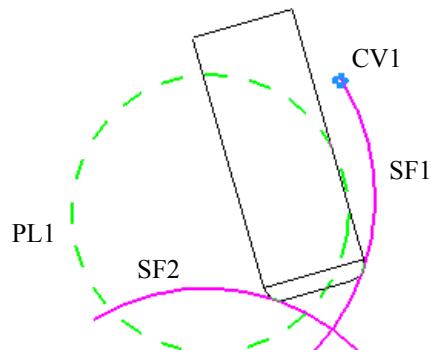
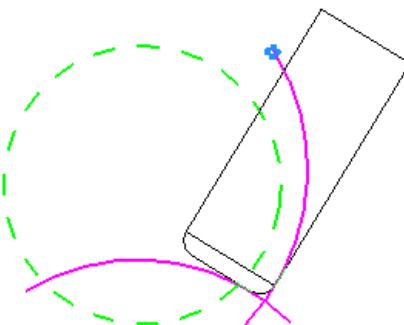
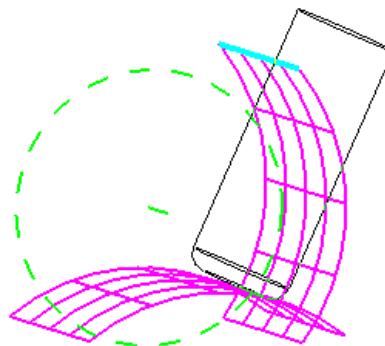
6 CUTTER AND TOOL AXIS STATEMENTS



TA/TT,DS,0.1
TA/GUIDE,CV1,TLLFT
GO/SF1,SF2,ON,PL1

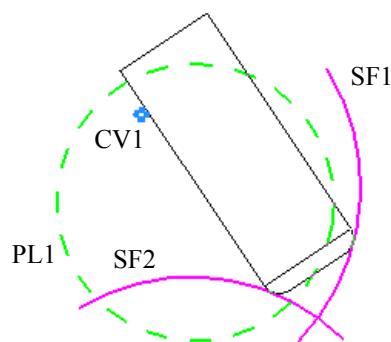


Same position mode as above without using guide curve



Guide curve with a thick value specified.

TA/TT,DS,0.1
TA/GUIDE,CV1,TLLFT,0.1
GO/SF1,SF2,ON,PL1



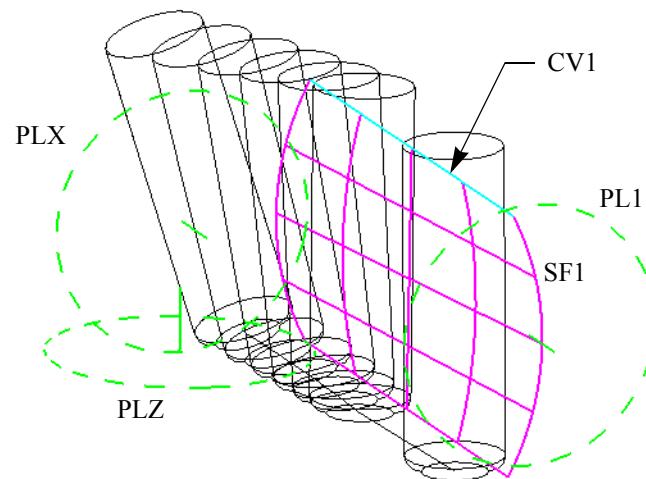
Guide curve with CONTCT specified.

TA/TT,DS,0.1
TA/GUIDE,CV1,CONTCT,TLRGRT
GO/SF1,SF2,ON,PL1

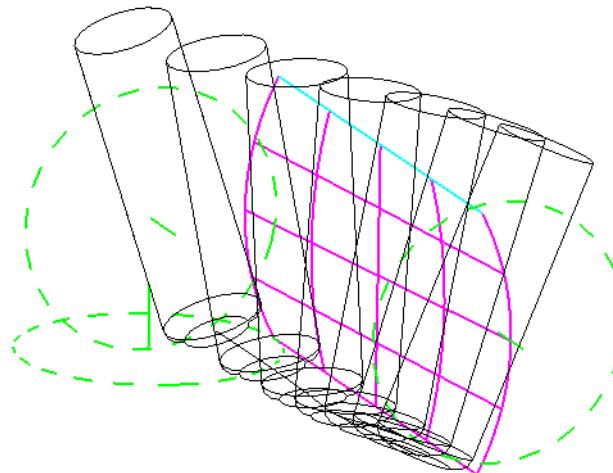
6 CUTTER AND TOOL AXIS STATEMENTS

Following figures illustrate one of the usage of a guide curve in continuous motion.

CU/0.75,0.125,3
MULTAX
TA/TT,DS,0.125
SRFVCT(VE/0,1,0)
GO/SF1,PLZ,ON,PLX
TA/GUIDE,CV1
IV/1,0,0
TLRG
GF/SF1,ON,PL1



Tool path without specifying TA/GUIDE,...



6.2.15 TLAXIS/ [. . . ,] GOUGCK, ON OFF

Allows gouge checking to be turned ON or OFF as applied to variable tool axis modes. When added to the TLAXIS command, the tool will be tilted away from the drive surface as necessary to prevent gouging. For a more detailed use of the GOUGCK statement, see the section in this chapter entitled "[GOUGCK](#)."

6 CUTTER AND TOOL AXIS STATEMENTS

6.2.16 TLAXIS/ INTERP, vector [, SMOOTH [, d, r]]
point-vector
i, j, k

This statement will cause the tool to start at the current tool axis vector and interpolate to a user defined ending vector.

“SMOOTH [,d,r]”:

- This allows an optional smooth interpolation between the starting and ending tool axis vectors.
 - “d” is a positive integer, specifying the degree of the interpolating polynomial (used to calculate the transition between tool axis modes). Specifying d=0 or d=1 disables smooth interpolation and uses the standard fanning algorithm.
 - “r” is a positive real number no more than 1, specifying how steep the interpolating polynomial is in the middle of the fanning or interpolating motion.
 - The “d” and “r” parameters are intended to allow the user to fine tune the way the tool axis changes with the SMOOTH modifier. Making the degree “d” higher leads to a more gradual transition between the fanning (interpolating) motion and the preceding and following motions. A higher rate “r” makes for an extra smooth transition, but a steeper change in the middle of the fanning (interpolating) motion.

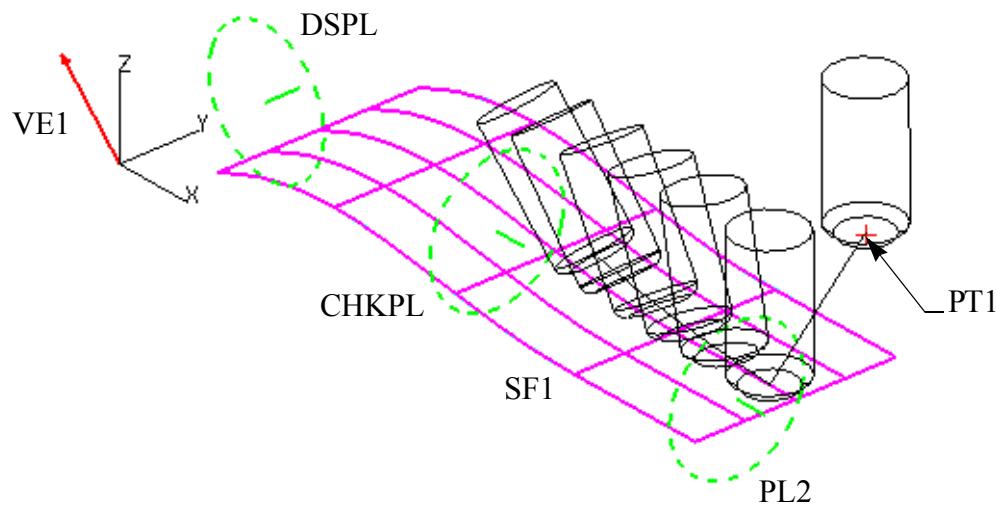
The default values are $d=5$, $r=1$, which creates a nice transition curve.

Example:

```
FROM    /PT1
TA      /0,0,1                      $$ start vector
SRFVCT/(VE/1,0,0)
GO      /PL2,SF1,ON,DSPL
TLAXIS/INTERP,VE1                  $$ ending vector
INDIRV/(VE/-1,0,0)
TLON   ,GOFWD/DSPL,CHKPL
```

The figure on next page shows the result of above example.

6 CUTTER AND TOOL AXIS STATEMENTS



6.2.17 TLAXIS/ [. . . ,], LOCK, OFF

END

**ds1 [, LINEAR] [, ds2 [, INTERP]]\$
RADIUS FAN**

[, OMIT]
RETAIN

This statement will cause the tool axis to be locked or unlocked for a user specified distance at the start and end of a move.

"ds1" is the move distance that the tool axis will remain locked at the start and end of the move and "ds2" is the move distance for the tool to transition from the locked tool axis to the tool axis currently in effect. LINEAR specifies that "ds1" is given as a linear distance. RADIUS specifies that "ds1" is given as a circular radius. An actual linear distance will be estimated from this value and the drive surface and check surface normals at the end of the move. The default value is LINEAR. If INTERP is specified, the transition move will be made by interpolating to a vector. If FAN is specified, the transition move will be made by fanning. INTERP is the default. If "ds2" is not specified or is set to zero, there will be no transition move. If OMIT is specified, the tool axis will not be locked at the start of the first move following the taxis lock statement.

If OFF is specified, tlaxis lock mode is turned off. This is the same as specifying the first distance as zero. If END is specified, the next move will be made with the tool axis locked at the start of the move only and lock mode will then be turned off.

6 POINT TO POINT MOTION STATEMENTS

Point to Point Motion Statements

This chapter deals with Point to Point Motion generation and related statements.

6.3 FROM

The FROM statement is used to establish an initial tool location. The valid syntax constructs for the FROM statement are:

6.3.1 FROM/ x, y [, z] [, vector] [, feedrate]

In this statement "x" specifies the initial X coordinate, "y" specifies the initial Y coordinate and the optional "z" specifies the initial Z coordinate. If omitted, the Z coordinate does not change. The optional vector specifies the [TLAXIS](#). If omitted, the TLAXIS does not change. Note that if the optional vector and/or the optional feedrate operators are used, the optional Z coordinate must be specified. The optional "feedrate" value sets the primary feed rate exactly as if it were a [FEDRAT](#) statement.

6.3.2 FROM/ point-vector [, feedrate] point [, vector]

This designates the point or point-vector origin to be the initial point. If specified, the optional vector sets the TLAXIS.

Examples:

```
FROM/1.5,13  
FROM/-3,-3,5  
FROM/-1,-1,0,0,0,1  
FROM/PT1  
FROM/PT1,0,1,0  
FROM/PT1,VE1,100  
FROM/PV1
```

6.4 GO

The GO statement causes the tool to move from its current location to a location specified by the 1 to 3 locating type geometry items. The first and third locating geometric items may be of the entity type line, plane, circle, curve, spline or

6 POINT TO POINT MOTION STATEMENTS

surface. The second locating geometric item which is a part surface can be of the entity type plane, surface or sspline. The GO statement generates the same motion that the following general list of statements would generate:

```
DNTCUT
    PSIS/ second-locating-geometry
    TLRGT-TLON-TLLFT (depending on direction-modifier)
    INDIRV/ -- (based on direction of move from starting position to
              first contact with the first-locating-geometry)
    THICK/ -- (values in effect)
    TLAXIS/ -- (values in effect)
    TOLER/ -- (values in effect)
    MAXDP/ -- (value in effect)
    GOFWD/ first-locating-geometry, modifier, third-locating-geometry
CUT
```

An INDIRV or INDIRP command points toward the third locating geometry (from the current location of the cutter) must be specified before the GO command if this locating geometry is of the type curve or spline.

If THICK is specified, the first locating geometric item will respect the drive surface thick value, the second locating geometric item will respect the part surface thick value and the third locating geometric item will respect the check surface thick value.

Example GO Statements:

```
NOPS,GO/SF1
GO/SF1,SF2,15
GO/LN3,PL44
GO/PAST,SF2,PLZ12,ON,LNX4
GO/ON,SF3,PLC12,TO,SF3A
```

Suggestions for using the GO statement:

- Proceed the GO statement with a SRFVCT statement. This will ensure that the cutter is positioned consistently regardless of the cutter location prior to the GO statement.
- An INDIRV or INDIRP command pointing toward the third locating geometry (from the current location of the cutter) can help in ambiguous cases.
- If the GO statement fails to execute after trying the above suggestions it means that the combination of settings (i.e. THICK, TLAXIS, TOLER)

6 POINT TO POINT MOTION STATEMENTS

and the geometric entities specified have created an ambiguous situation. In such cases it may be necessary to position the tool within a **DNTCUT/CUT** sequence.

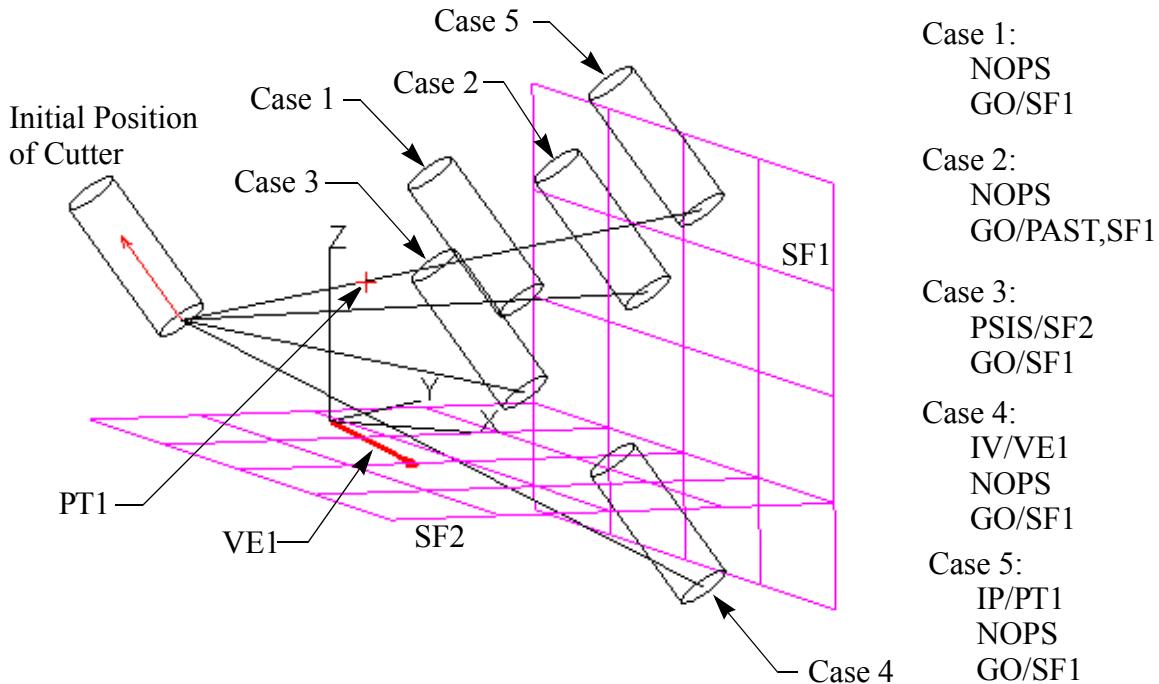
6.4.1 One Surface GO

```
[NOPS, ]GO/[first-entity-modifier,]first-entity      $  
[ , feedrate]
```

This statement is a one surface GO that specifies the shortest move to the first-entity.

This move is calculated depending upon the modifiers INDIRV and **NOPS** (no part surface). If NOPS is specified, the tool is moved to the first-entity in the shortest possible distance on the XY-plane passing through the current tool end point disregarding what is the current part surface, or along the direction of the vector specified by an immediately preceding **INDIRV** or **INDIRP**. If NOPS is not specified, then the tool is moved to the intersection of the first-entity and the last part-surface specified in the possible shortest distance.

The optional first-entity-modifier indicates whether the cutter will be located **TO**, **ON** or **PAST** the first-entity. The default is **TO**.



6 POINT TO POINT MOTION STATEMENTS

6.4.2 Two Surface GO

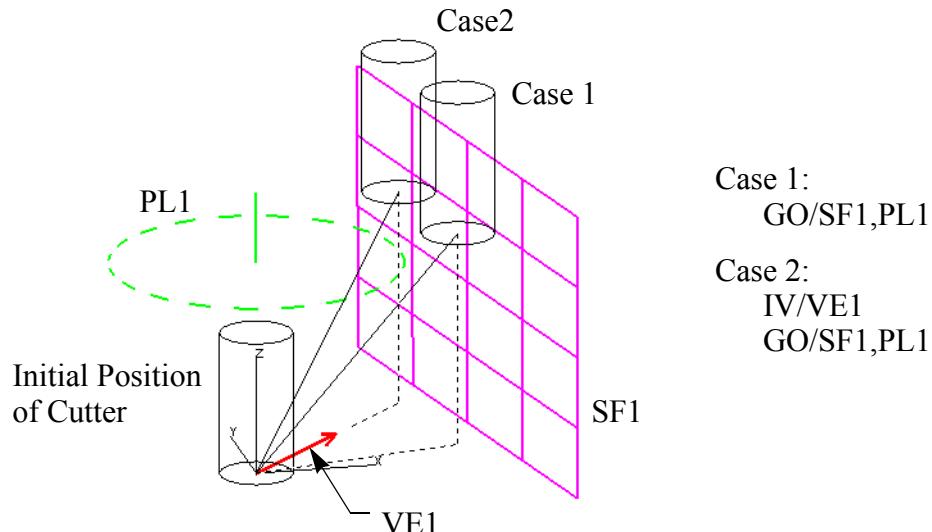
```
GO/ [first-entity-modifier,]first-entity, $  
[second-entity-modifier,]second-entity $  
, feedrate]
```

This statement is a two surface GO that specifies the shortest move to the intersection of the first-entity and the second-entity or along the direction of the vector specified by an immediately preceding **INDIRV** or **INDIRP** statement.

The second-entity is the name of the geometry item that will be or currently is the part surface (part-surface).

The optional first-entity-modifier indicates whether the cutter will be located TO, ON or PAST the first-entity. The default is TO.

The optional second-entity-modifier indicates whether the cutter will be located ON or TO the second entity. The default is TO. ON means **TLONPS** and TO means **TLOFPS**.



6 POINT TO POINT MOTION STATEMENTS

6.4.3 Three Surface GO

```
GO/ [first-entity-modifier,]first-entity, $  
      [second-entity_modifier,]second-entity $  
          NOPS  
      [,third-entity-modifier],third-entity $  
      [, feedrate]
```

This statement is a three surface GO that specifies a move to the intersection of the first-entity, the second-entity and the third-entity.

The optional first-entity-modifier indicates whether the cutter will be located TO, ON or PAST the first-entity. The default is TO.

The first-entity is the name of the geometry item that will act as the drive surface for the internal [GOFWD](#) statement in the [DNTCUT/ CUT](#) command list noted above.

The optional secondary modifier indicates whether the cutter will be located ON or TO the second-entity. TO is the default. ON means [TLONPS](#) and TO means [TLOFPS](#).

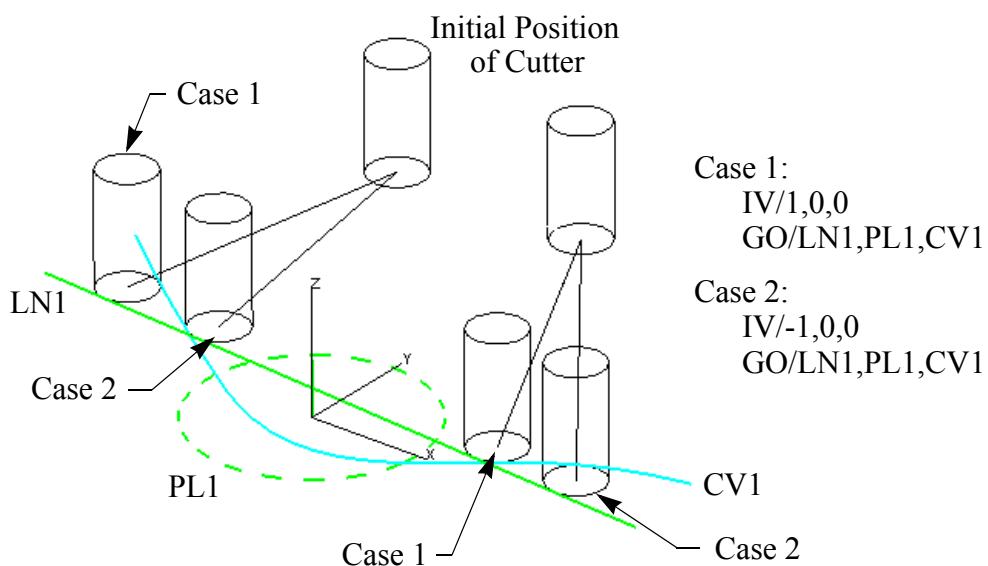
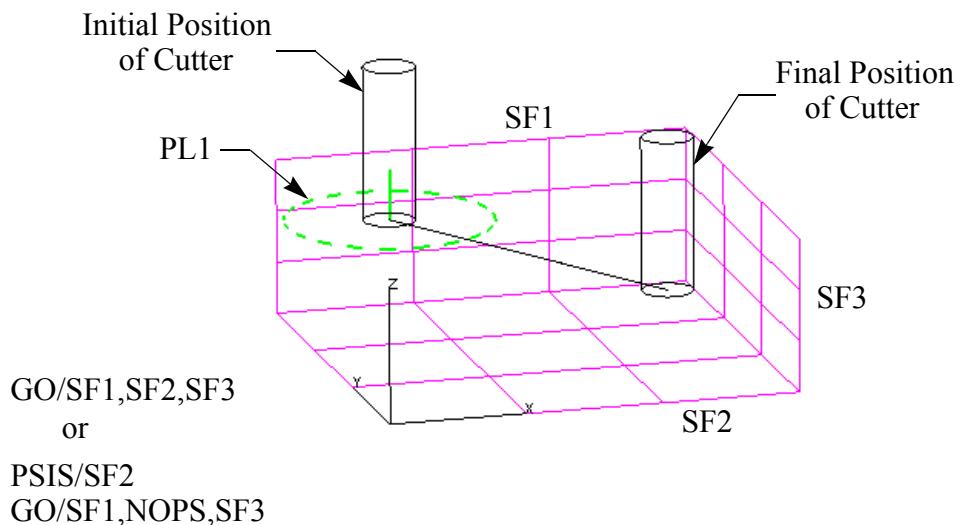
The second-entity is the name of the geometry item that will be or currently is the part surface (part-surface). If “[NOPS](#)” is specified in stead of second-entity, the current part surface will be used, otherwise, the Z zero plane will be used.

The optional third-entity-modifier indicates whether the cutter will be located TO, ON or PAST the third-entity. The default is TO.

The third-entity is the name of the geometry item that will act as the check surface for the internal GOFWD statement in the DNTCUT/ CUT command list noted above.

If the third entity is of the type curve or spline, an [INDIRV](#) or [INDIRP](#) statement must be specified before the GO statement. If there are several intersections between this third entity and the first entity, the cutter will move to the first entity in the shortest path internally and then to the closest intersection which would satisfy the INDIRV or INDIRP condition. Under this condition, the INDIRV or INDIRP will have the same meaning as the cs-vector of the [SRFVCT](#) statement.

6 POINT TO POINT MOTION STATEMENTS



Note that the final position of the cutter depends on the initial position of the cutter relative to the locating geometry even if the same INDIRV statement is specified before the same GO statement.

6 POINT TO POINT MOTION STATEMENTS

6.5 The SRFVCT Statement

The SRFVCT (surface vector) statement is used to specify what is to be considered the 'TO' side of the entities specified in a **GO** statement that immediately follows the SRFVCT statement.

The syntax for the SRFVCT statement is:

SRFVCT / [ds-vector] [, cs-vector]

Where:

"ds-vector": A vector that points in the general direction of the desired 'TO' side of the first entity specified in the GO statement.

"cs-vector": A vector that points in the general direction of the desired 'TO' side of the third entity specified in the GO statement.

When determining which vectors to use, imagine standing in the center of the tool, as it would be in the desired position, and looking toward the entities to which you want to position. The direction you must look to see the first entity would be the general direction for the "ds-vector". The direction you must look to see the third entity would be the general direction for the "cs-vector".

The ds-vector and cs-vector may be specified either as a vector, point vector or as three scalar values representing vector coordinates.

When using SRFVCT with a One or Two surface GO statement only the cs-vector is considered. This is because when a One or Two surface Go statement is processed the system internally creates a drive surface and uses the first entity specified in the GO statement as the check surface.

It is highly recommended that a SRFVCT statement be used prior to any GO statement. This will ensure that the cutter is positioned consistently regardless of the cutter location prior to the GO statement.

Example SRFVCT Statements:

SRFVCT/V1	"ds-vector" Only
SRFVCT/1, 0, 0	"ds-vector" Only
SRFVCT/, V2	"cs-vector" Only

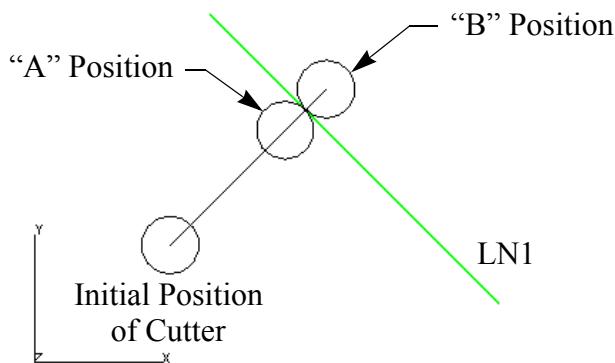
6 POINT TO POINT MOTION STATEMENTS

SRFVCT/, 0, 1, 0
SRFVCT/V1, V3

"cs-vector" Only
"ds-vector" and "cs-vector"

The following examples show how to use the SRFVCT statement with different type of GO statements. All examples below assume THICK/0 condition. Note that reversing the direction of the vector changes the TO side to the PAST and vice versa.

1) One surface GO

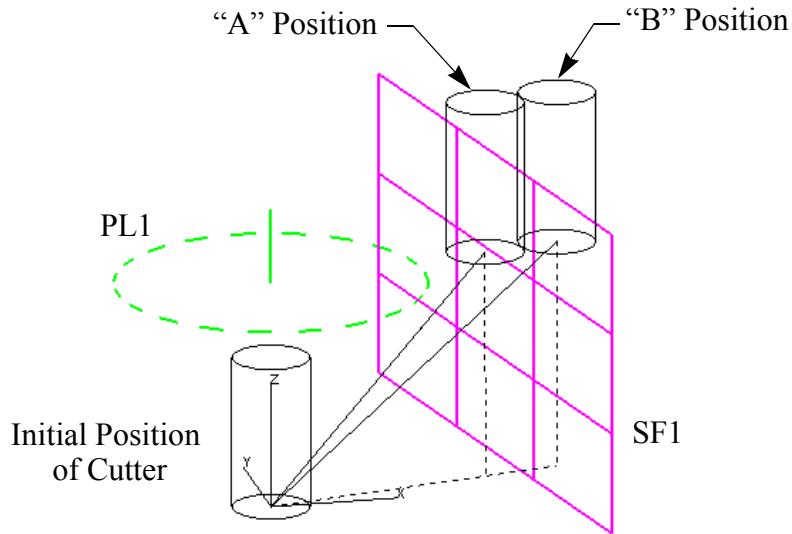


- A) Any set of the following commands will move the cutter from the initial position to the "A" position.
 - i. GO/LN1
 - ii. SRFVCT/, 1, 0, 0 or SRFVCT/, 0, 1, 0
GO/LN1
 - iii. SRFVCT/, -1, 0, 0 or SRFVCT/, 0, -1, 0
GO/PAST, LN1
 - iv. SRFVCT/1, 0, 0, 1, 0, 0 *Only the last 3 values will be used
GO/LN1

- B) Any set of the following commands will move the cutter from the initial position to the "B" position.
 - i. GO/PAST, LN1
 - ii. SRFVCT/, -1, 0, 0 or SRFVCT/, 0, -1, 0
GO/LN1
 - iii. SRFVCT/, 1, 0, 0 or SRFVCT/, 0, 1, 0
GO/PAST, LN1
 - iv. SRFVCT/1, 0, 0, -1, 0, 0 *Only the last 3 values will be used
GO/LN1

6 POINT TO POINT MOTION STATEMENTS

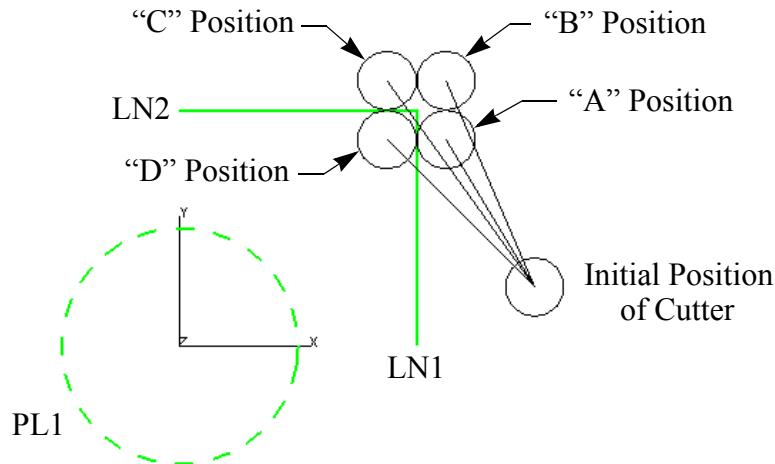
2) Two surface GO



- A) Any set of the following commands will move the cutter from the initial position to the "A" position.
- i. GO/SF1, PL1
 - ii. SRFVCT/, 1, 0, 0
GO/SF1, PL1
 - iii. SRFVCT/, -1, 0, 0
GO/PAST, SF1, PL1
 - iv. SRFVCT/1, 0, 0, 1, 0, 0 *Only the last 3 values will be used
GO/SF1, PL1
- B) Any set of the following commands will move the cutter from the initial position to the "B" position.
- i. GO/PAST, SF1, PL1
 - ii. SRFVCT/, -1, 0, 0
GO/SF1, PL1
 - iii. SRFVCT/, 1, 0, 0
GO/PAST, SF1, PL1
 - iv. SRFVCT/1, 0, 0, -1, 0, 0 *Only the last 3 values will be used
GO/SF1, PL1

6 POINT TO POINT MOTION STATEMENTS

3) Three Surface GO



- A) Any set of the following commands will move the cutter from the initial position to the "A" position.
- i. GO/LN1, PL1, LN2
 - ii. SRFVCT/-1, 0, 0, 0, 1, 0
GO/LN1, PL1, LN2
 - iii. SRFVCT/1, 0, 0, 0, 1, 0
GO/PAST, LN1, PL1, LN2
 - iv. SRFVCT/-1, 0, 0, 0, -1, 0
GO/LN1, PL1, PAST, LN2
 - v. SRFVCT/1, 0, 0, 0, -1, 0
GO/PAST, LN1, PL1, PAST, LN2
- B) Any set of the following commands will move the cutter from the initial position to the "B" position
- i. GO/LN1, PL1, PAST, LN2
 - ii. SRFVCT/-1, 0, 0, 0, -1, 0
GO/LN1, PL1, LN2
 - iii. SRFVCT/1, 0, 0, 0, -1, 0
GO/PAST, LN1, PL1, LN2
 - iv. SRFVCT/-1, 0, 0, 0, 1, 0

6 POINT TO POINT MOTION STATEMENTS

- GO/LN1, PL1, PAST, LN2
- v. SRFVCT/1,0,0, 0,1,0
GO/PAST, LN1, PL1, PAST, LN2
- C) Any set of the following commands will move the cutter from the initial position to the "C" position.
- i. GO/PAST, LN1, PL1, PAST, LN2
 - ii. SRFVCT/1,0,0, 0,-1,0
GO/LN1, PL1, LN2
 - iii. SRFVCT/-1,0,0, 0,-1,0
GO/PAST, LN1, PL1, LN2
 - iv. SRFVCT/1,0,0, 0,1,0
GO/LN1, PL1, PAST, LN2
 - v. SRFVCT/-1,0,0, 0,1,0
GO/PAST, LN1, PL1, PAST, LN2
- D) Any set of the following commands will move the cutter from the initial position to the "D" position
- i. GO/PAST, LN1, PL1, LN2
 - ii. SRFVCT/1,0,0, 0,1,0
GO/LN1, PL1, LN2
 - iii. SRFVCT/-1,0,0, 0,1,0
GO/PAST, LN1, PL1, LN2
 - iv. SRFVCT/1,0,0, 0,-1,0
GO/LN1, PL1, PAST, LN2
 - v. SRFVCT/-1,0,0, 0,-1,0
GO/PAST, LN1, PL1, PAST, LN2

6.6

GODLTA

The GODLTA statement causes the tool to move the amount specified. The word GODLTA may be abbreviated as GD.

Example GODLTA Statements:

```
GODLTA/1
GD/1,1
GODLTA/-.2,1.34,5
GODLTA/PL1
```

6 POINT TO POINT MOTION STATEMENTS

GODLTA/VE3
GODLTA/PV1
GODLTA/SF1

6.6.1 GODLTA/ distance [, feedrate]

This syntax causes the tool to move the distance specified along the current tool axis. A positive distance moves the tool up the tool axis (the direction the tool axis is pointing). A negative distance moves the tool down the tool axis.

6.6.2 GODLTA/ delta-x, delta-y, delta-z [, feedrate]

Delta-x is the distance in the X direction to move the tool. Delta-y is the distance in the Y direction to move the tool. Delta-z is the distance in the Z direction to move the tool.

6.6.3 GODLTA/ plane [, feedrate]

This statement causes the tool to be moved along the current tool axis until the tool end point is *ON* the plane specified by the plane. The current tool axis may not be within 5 degrees of parallel with the plane.

6.6.4 GODLTA/ vector [, feedrate]

This syntax causes the tool to move in the direction of the vector for a distance given by the magnitude of the vector.

6.6.5 GODLTA/ point-vector [, feedrate]

This syntax causes the tool to move in the direction of the point-vector for a distance given by the magnitude of the point-vector.

6.6.6 GODLTA/ surface [, feedrate]

This syntax causes the tool to move directly up or down the tool axis until the tip of the tool is in contact with the specified surface.

If the tool moves up the tool axis, it will finish past the surface and if it moves down the tool axis, it will finish *TO* the surface. Note this is different from the way GODLTA/*plane* works which always goes *ON* the specified plane. It should also

6 POINT TO POINT MOTION STATEMENTS

be noted that any thick statement in effect at the time the GDLTA statement is processed will be applied and the resulting new surface will become the current part surface.

6.7 GOTO

The GOTO statement causes the tool to move from its current location to the location specified. The word GOTO may be abbreviated as GT.

Example GOTO Statements:

```
GOTO/0,0  
GT/-3.45,.010,1  
GOTO/12,12,2.5,-1,0,0  
GOTO/PT1,VE1  
GOTO/PT1,0,0,1,100  
GOTO/PN1,INVERS  
GOTO/PN2,OMIT,2,3,7  
GOTO/PN3,RETAIN,4,6,8,10  
GT/PN4,AVOID,1,1,2,5  
GT/PV1
```

6.7.1 GOTO/ x, y [, z [, i, j, k]] [, feedrate]]

"x" specifies the X coordinate, "y" specifies the Y coordinate and "z" specifies the Z coordinate. If omitted, the Z coordinate is unchanged. The optional i, j, k clause specifies the new [TLAXIS](#). If omitted, the TLAXIS vector is unchanged.

If either the i, j, k clause or the optional feedrate is specified, the optional Z coordinate must be specified.

6.7.2 GOTO/ point [, vector] [, feedrate]]

The point is that to which the tool is moved. If specified, the optional vector represents the new TLAXIS vector.

6.7.3 GOTO/ point-vector [, feedrate]]

The point component of the point-vector is the point at which the tool will be positioned, the vector component of the point-vector will become the new tool axis.

6 POINT TO POINT MOTION STATEMENTS

6.7.4 GOTO/ patern-id [, INVERS] [, CONST] [, AVOID-clause] \$ [, RETAIN-clause] [, OMIT-clause]

This GOTO statement generates a GOTO for every point in the **PATERN** specified by patern-id. The motion will start at the first point or point-vector in the PATERN, and will end with the last point or point-vector unless INVERS is specified. If INVERS is specified, the order will be reversed.

The OMIT modifier is used to skip points in a pattern when using cutter motion.

Example:

```
GOTO/PN1, OMIT, 2
```

In the above example, the cutter goes to the first points in PN1, skips the 2nd point and resumes with the next point in the pattern.

RETAIN means that only the points listed are to be used.

Example:

```
GOTO/PN2, RETAIN, 2, 4, 6
```

In the above example, the cutter will only go to points 2, 4 and 6.

- OMIT and RETAIN cannot both be specified in the same GOTO statement.
- When specifying the list of point positions, the THRU clause may be used to specify a range of points.

Example:

```
GOTO/PN3, RETAIN, 3, THRU, 7
```

The AVOID clause causes the tool to retract along the current tool axis between the points specified. The distance of retraction is the number immediately following the AVOID statement.

Example:

```
GOTO/PN3, AVOID, 2.75, 3, 5, 9
```

6 POINT TO POINT MOTION STATEMENTS

The CONST modifier is used in conjunction with the INVERS modifier. It allows the point positions in the OMIT, AVOID and RETAIN clauses to be specified in the original order that they were defined (or constructed), even though the motion will be in the INVERS direction. For example:

```
PN4=PATERN/ARC,CI3,45,315,CCLW,7  
GOTO/PN4,INVERS,CONST,RETAIN,1,4,5,7
```

will cause motion to the points on CI3 at 45°, 180°, 225° and 315°.

```
GOTO/PN4,INVERS,RETAIN,1,4,5,7
```

will cause motion to the points on CI3 at 315°, 180°, 135° and 45°.

6.8 NOPS

The NOPS statement (No Part Surface) turns off the current Part Surface for a single SURFACE GO/ statement as a one-shot command. The cutter will Go to the drive surface specified in the minimum distance. If NOPS is specified, the tool is moved to the drive-surface in the shortest possible distance or along the direction of the vector specified by an immediately preceding INDIRV or INDIRP. NOPS may be at the beginning of a statement or stand alone.

Examples:

```
NOPS,GO/SF1  
NOPS  
GO/SF1
```

Continuous Motion Statements

This chapter deals with Continuous Path Motion generation and related statements. In continuous path motion there are three surfaces that guide the cutter: the Part Surfaces, the Drive Surface and the Check Surface.

6.9

Part Surface

The Part Surface normally controls the depth of cut; it guides the bottom of the cutter. The part surface usually remains the same for a sequence of motion statements; therefore, it only needs to be specified in a motion statement when it changes. **GO** or **PSIS** statements are used to change the part surface. The **GO** statement is actually a three-surface statement that designates the **Drive Surface**, the **Part Surface** and the **Check Surface**.

Example GO and PSIS statements:

```
GO/ON, SF3 , PL3 , TO , SF4  
GO/CV1 , PL1 , CV2  
PSIS/SF7
```

6.9.1

AUTOPS

The AUTOPS statement causes **NCL** to use a plane, defined as through the current tool end point and perpendicular to the current tool axis vector, as the part surface on subsequent motion statements. The valid syntax for the AUTOPS statement is:

```
AUTOPS
```

6.9.2

PSIS

The PSIS statement is used to establish a part surface. The valid syntax construct for the PSIS statement is:

```
PSIS/ part-surface
```

The part-surface becomes the new part surface. Only planes, surfaces and ssplines may be part surfaces.

6 CONTINUOUS MOTION STATEMENTS

Example PSIS Statements:

PSIS/SF1
PSIS/PL2
PSIS/SF2 [.25 , .5]

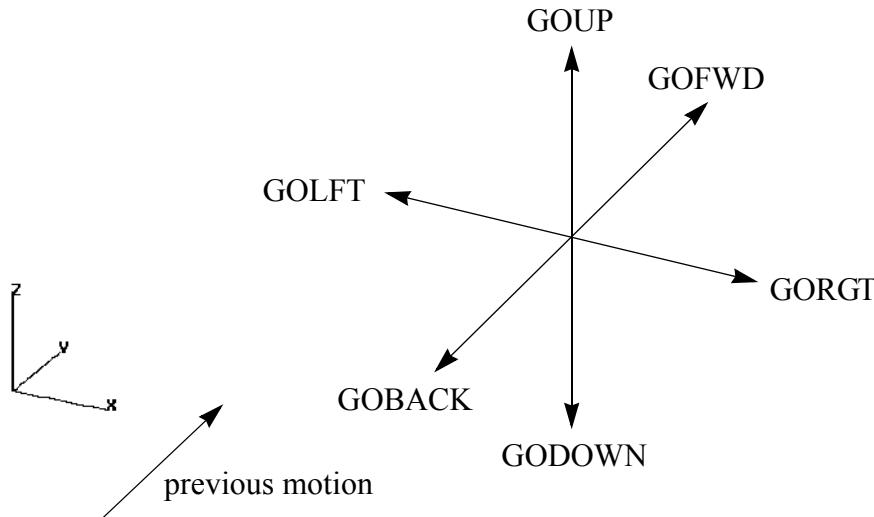
see "User Specified U and V Values
for CURVES and SURFACES"

6.10 Drive Surface

The Drive Surface guides the side of the cutter. In continuous path motion, the drive surface must be specified in each motion statement. The motion statements used to specify the drive surface are **GOBACK**, **GOFWD**, **GORGT**, **GOLFT**, **GOUP**, **GODOWN** and **GO** as mentioned above.

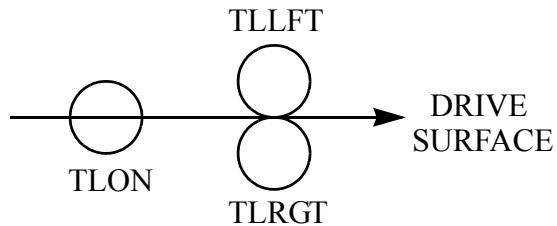
Example Drive Surface statements:

GOBACK/LN2 , TO , LN1
GOFWD/LN1 , PAST , CI1
GOLFT/ PL1 , TO , PL2
GORGT/ LN1
GOUP/ PL3 , PAST , PL4
GODOWN/ PL3 , PL5



6.11 Tool Conditions

TLRGT (tool right), **TLLFT** (tool left) and **TLON** (tool on), specify the position of the tool in relation to the drive surface during a move. If omitted, then the last tool condition specified is used. If none has been specified, the default is **TLON**.



Example motion statements with drive surface modifiers:

```
TLON, GORGT/LN1, ON, LN2  
TLLFT, GOLFT/LN2, TO, LN3  
TLRGT, GORGT/L1, TO, L3
```

Note: **TLRGT**, **TLLFT** and **TLON** may appear alone as a complete statement.

6.12 Combined Drive And Part Surface

NCL has the ability to drive a curve as both the drive and part surface. The syntax is as follows:

```
NOPS  
GOFWD/curve
```

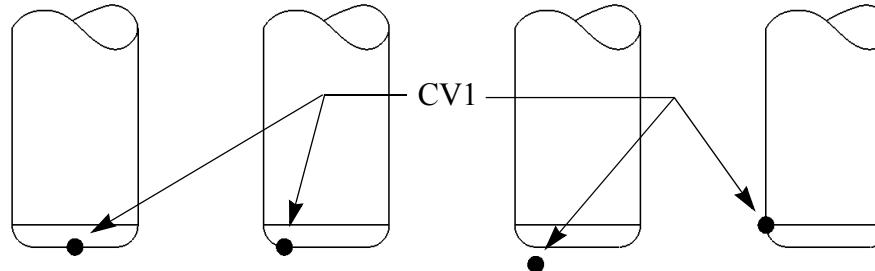
These statements will cause the tool to go to the start point of the curve and move along the curve to the end point. When the tool condition is **TLON**, the center of the tool will be ON the curve. When the tool condition is **TLLFT** or **TLRGT**, the contact point of the tool will be the tangent point of the corner radius and the bottom of the tool. If the tool has no corner radius, the contact point will be the intersection of the side and bottom of the tool.

If part surface **THICK** is in effect, it will be honored. **THICK** is applied relative to the contact point previously described. For example, if the value of **THICK** (when

6 CONTINUOUS MOTION STATEMENTS

a negative number is specified) is equal to the corner radius of the tool, the contact point will be at the tangent point of the corner radius and the side of the tool.

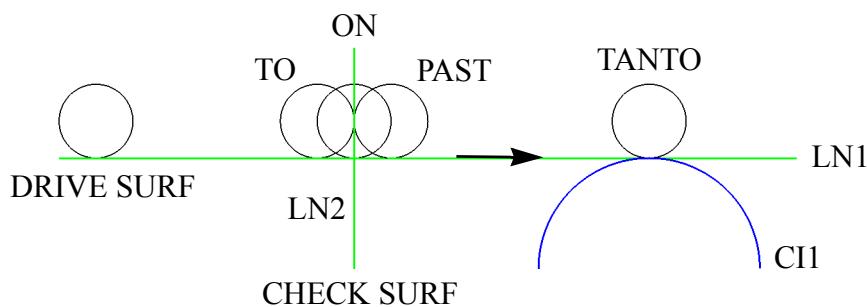
The current tool axis mode is ignored. The following examples may help to clarify the use of this command.



CU/0.75,0.12,2	CU/0.75,0.12,2	CU/0.75,0.12,2	CU/0.75,0.12,2
TLRG	TLRG	TLRG	TLRG
NOPS	NOPS	TH/0.1,0	TH/-0.12,0
GF/CV1	GF/CV1	NOPS	NOPS
		GF/CV1	GF/CV1

6.13 Check Surface

The Check Surface defines the end of a move along the present drive surface. It must be specified along with the drive surface unless the IMPLIED CHECK SURFACE technique is used. The Check Surface modifiers, TO, ON, PAST, PSTAN (part surface tangent to check surface) and TANTO (drive surface tangent to check surface), specify the position of the tool in relation to the check surface at the end of a move. If the modifier is omitted, TO is assumed.



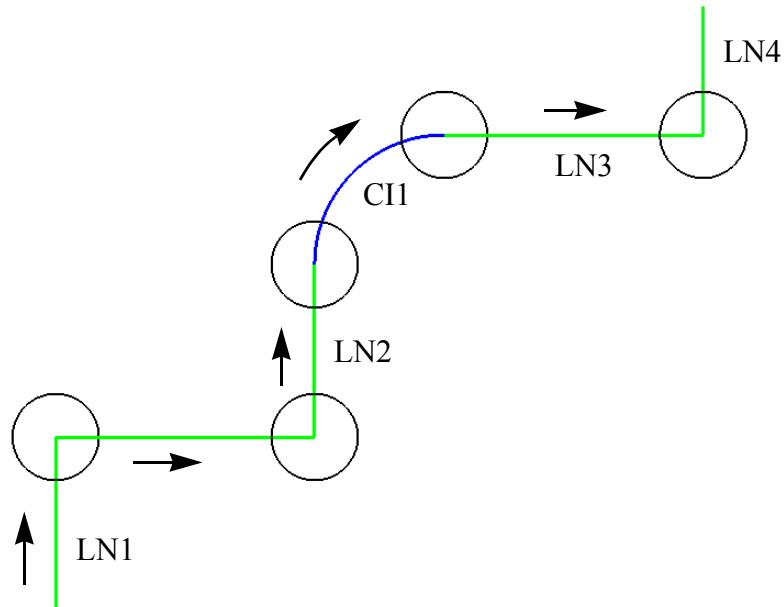
Example motion statements with Check Surface modifiers:

6 CONTINUOUS MOTION STATEMENTS

```
TLON, GORGT/LN1, ON, LN2  
TLLFT, GOLFT/LN2, TO, LN3  
TLRGT, GORGT/L1, TO, L3
```

The following four statements with the check surface provided will produce the motion as shown below:

1. TLON, GORGT/LN1, ON, LN2
2. GOLFT/LN2, TANTO, CI1
3. GOFWD/CI1, TANTO, LN3
4. GOFWD/LN3, ON, LN4



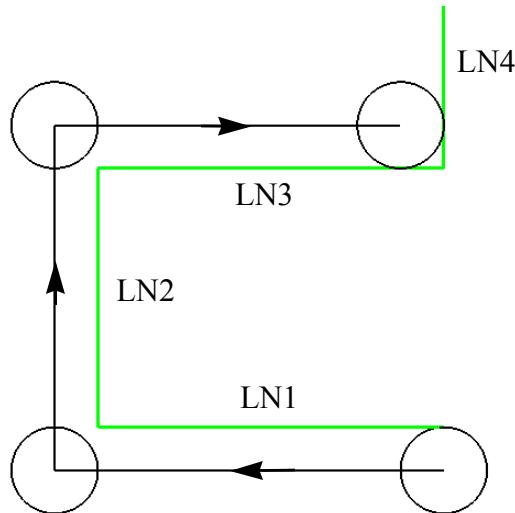
Using the IMPLIED CHECK SURFACE technique to drive the tool along geometry eliminates the Check Surface. This means that a sequence of motion statements may be used that only specify the Drive Surface. The following four statements will produce the same motion shown in the previous diagram using the Implied Check Surface technique.

1. TLON, GORGT/LN1
2. GOLFT/LN2
3. GOFWD/CI1
4. GOFWD/LN3, ON, LN4

6 CONTINUOUS MOTION STATEMENTS

When using IMPLIED CHECK SURFACES, **NCL** will choose the appropriate check surface modifier for each move, based upon the current tool condition (**TLLFT**, **TLRGT** or **TLON**). The following three statements will produce the motion in the diagram below.

1. TLLFT , GOLFT/LN1
2. GORGT/LN2
3. GORGT/LN3 , TO , LN4



The following 3 statements with the check surfaces specified will generate the exact same motion as the diagram above:

1. TLLFT , GOLFT/LN1 , PAST , LN2
2. GORGT/LN2 , PAST , LN3
3. GORGT/LN3 , TO , LN4

The check surface modifier **PAST** is used in statements "1" and "2" because the tool needs to be **TLLFT** for the following moves.

When running **NCL** interactively, using implied check surfaces causes the tool motion to be delayed one or more statements. This is because a motion statement cannot be completed until the statement containing the next drive surface (and therefore, the current statement's check surface) is seen. In the following sequence,

1. GORGT/LN1
2. GOLFT/LN2

6 CONTINUOUS MOTION STATEMENTS

the motion generated by statement 1: cannot be completed until the second statement has been seen because **NCL** doesn't know where to stop the move along LN1 until it sees that the check surface will be LN2. Because of this, certain statements are not allowed while using implied check surfaces because they would cause a possible confusing situation.

If an error occurs during an implied check surface move, **NCL** reports the error and displays the motion statement that was generating the motion. This will be the motion statement prior to the last one entered. Both motion statements will then need to be reprocessed.

This is because **NCL** doesn't know which statement was actually in error; the statement containing the drive surface, or the statement containing the check surface.

It is possible to place any valid **NCL** statement between motion statements in an implied check surface range, however this may be dangerous. For example:

1. PSIS/SF1
2. GOFWD/PL1
3. GORGT/PL2
4. PSIS/SF2
5. GOLFT/PL3, PAST, PL4

Attempting to change the part surface during an implied check surface range will probably result in a tool out of position warning. It is a good idea to use an explicit check surface when doing anything like this. However, there are many valid statements which may be inserted in an implied check surface range, for example:

1. TLLFT,GL/LN1,30
2. GR/LN2
3. TLRGT
4. FEDRAT/10
5. GORGT/LN3, TO, LN4

6.13.1 Multiple Check Surfaces

Often in the process of programming motion statements it becomes difficult to determine which of several possible check surfaces will be reached at the end of the move. For example, when taking rough and finish cuts using different size cutters, a large diameter cutter may not reach into a fillet radius. In these instances an additional check surface can be referred to, and if the cutter finds the alternate check surface first, a different series of motions can be performed.

6 CONTINUOUS MOTION STATEMENTS

The format for multiple check surface motion statements is

```
GOFWD /drive surface      $
GOBACK
GORGT
GOLFT
GOUP
GODOWN

[, TO    ], check-surface-1,statement-label-1      $
ON
PAST
PSTAN
TANTO

[, TO    ], check-surface-2,statement_label_2      $
ON
PAST
PSTAN
TANTO

[ [, TO], [ . . . ] [ . . . ] ]
```

Where "statement-label-1" represents a statement label to which the program should transfer control if "check-surface-1" is reached first. "Statement-label-2" represents the statement label of the transfer point when "check-surface-2" is reached first - and "[. . .]" represents the statement label of the transfer point when additional check surfaces "[. . .]" are reached first. The maximum number of check surfaces and statement labels allowed is five.

Example:

```
MAC1=MACRO
FROM/ PT2
GO/ CI1
TLRGT, GORGT/ CI1, TANTO, LN1
GOFWD/ LN1, TO, LN3, ID1, TO, LN2, ID2
ID1:   TLLFT, GOLFT/ LN3, TO, LN4
JUMPTO/ ID3           $$ SKIP NEXT TWO STATEMENTS
ID2:   TLLFT, GOFWD/ LN2, TO, LN3
GOLFT/ LN3, TO, LN4
ID3:   GO.../...       $$ CONTINUE
TERMAC
CALL/ MAC
```

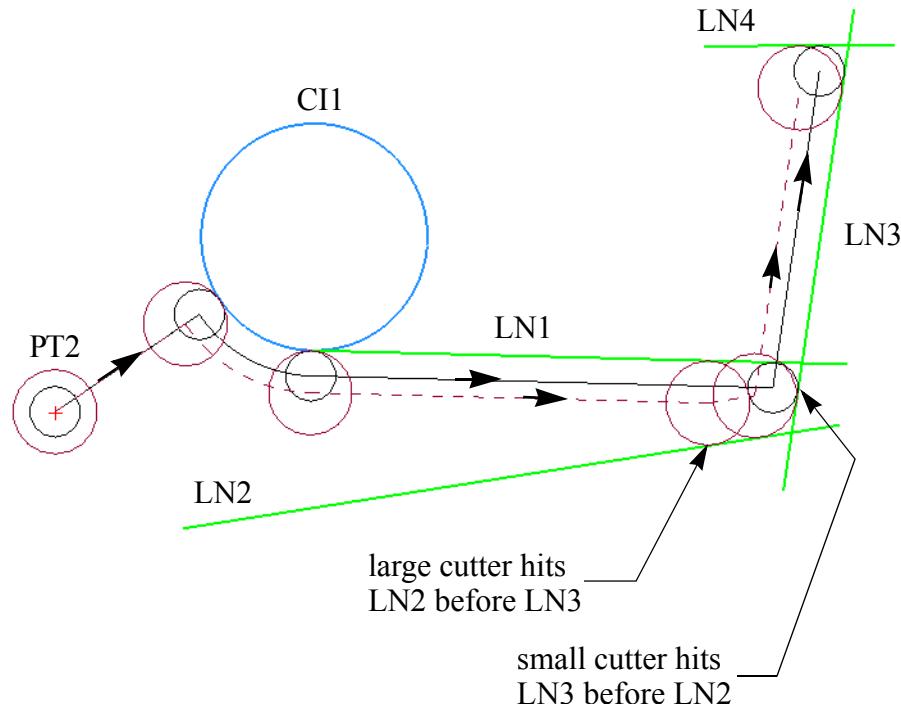
6 CONTINUOUS MOTION STATEMENTS

In this example, LN2 may be defined in such a manner that the exact location in relation to LN1 is not known. If it is imperative that LN2 not be gouged, the multiple check surface feature can be used.

In the event that LN2 should interfere with the cutter motion along LN1 to LN3 (as with a large cutter), **NCL** will generate cutter motions to move the cutter along LN1 to LN2 and then along LN2 to LN3 thereby preventing LN2 from being gouged.

If LN2 does not interfere (as with a smaller cutter), the motion will be along LN1 directly to LN3 and along LN3 to the rest of the part. In this case, the actual tool path would depend upon the size of the cutter being used. A smaller diameter cutter would take the path indicated in "ID1" and a larger cutter would take the path indicated in "ID2."

This feature may only be used within a LOOP or a **MACRO**.



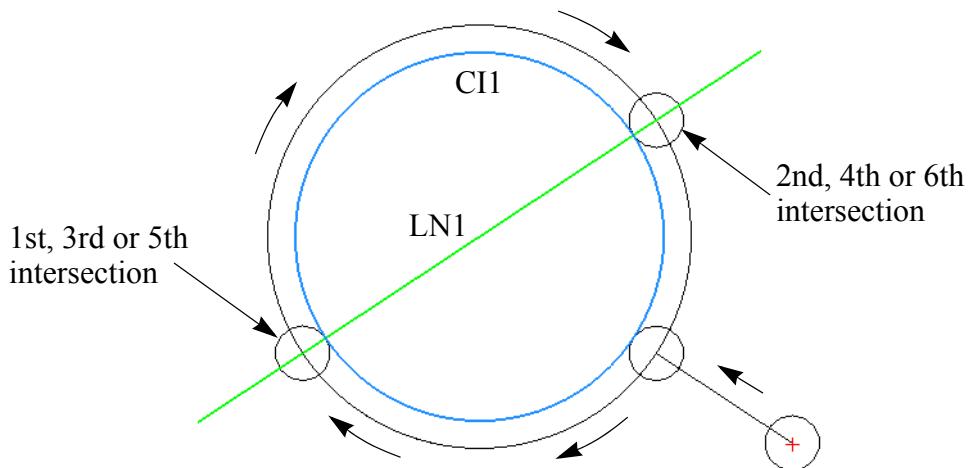
6 CONTINUOUS MOTION STATEMENTS

6.13.2 Multiple Intersection Motion Statements

The format for the multiple intersection type motion statement is:

```
GOFWD /drive-surface[, TO      ], n, INTOF, check-surface  
GOBACK          ON  
GORGT           PAST  
GOLFT           PSTAN  
TANTO
```

Where "n" represents the "nth" intersection of the drive-surface and check-surface. An intersection is actually where the requested tool to check-surface condition applies. In the case of a line intersecting a circle, it is possible that more than two intersections could be requested. If the user wanted to go around the circle more than once, the first intersection could be specified as the 1st, 3rd or 5th intersection. By the same token, the second intersection could be specified as the 2nd, 4th or 6th intersection. This is illustrated in the sketch as shown below.

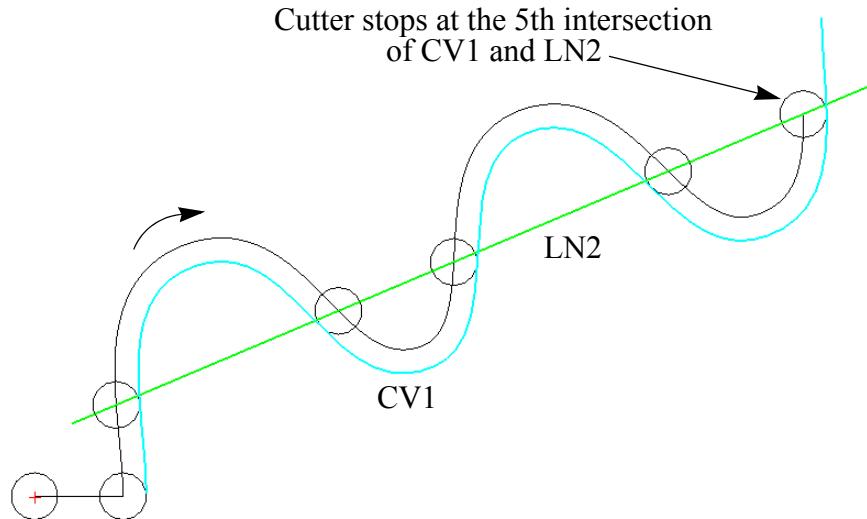


This format has application when either the drive-surface or check-surface is a circle or curve. In such cases, the cutter may intersect the check-surface more than once in its motion along the chosen drive-surface (note that the drive-surface/part-surface and check-surfaces need not actually intersect, it is only necessary that the cutter can be positioned TO, ON, PAST, PSTAN or TANTO the check-surface).

To select a parameter for "n," the programmer must count the number of possible stopping positions of the cutter, including the desired one, and enter this count as

6 CONTINUOUS MOTION STATEMENTS

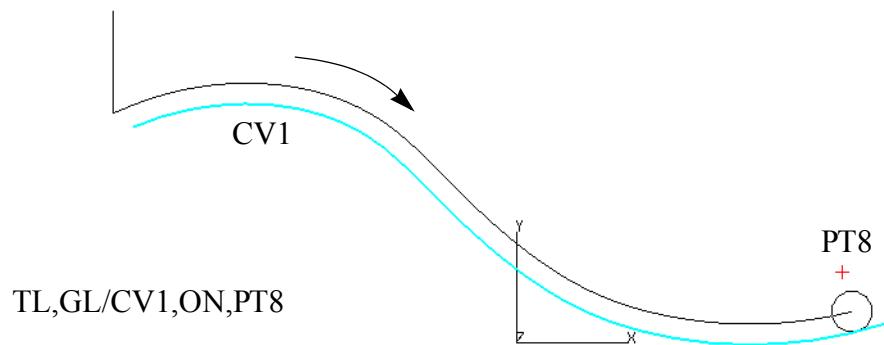
the value of "n" in the statement. The value of "n" is unlimited. The following sketch shows an example of multiple intersections of a curve and a line.



6.13.3 Point As A Check Surface

NCL has the ability to use a point as a check surface. The tool will stop when the point lies on a plane through the tool end point and perpendicular to the forward direction and is contained within a cylinder having its axis as the tool axis and its radius equal to the maximum radius of the tool.

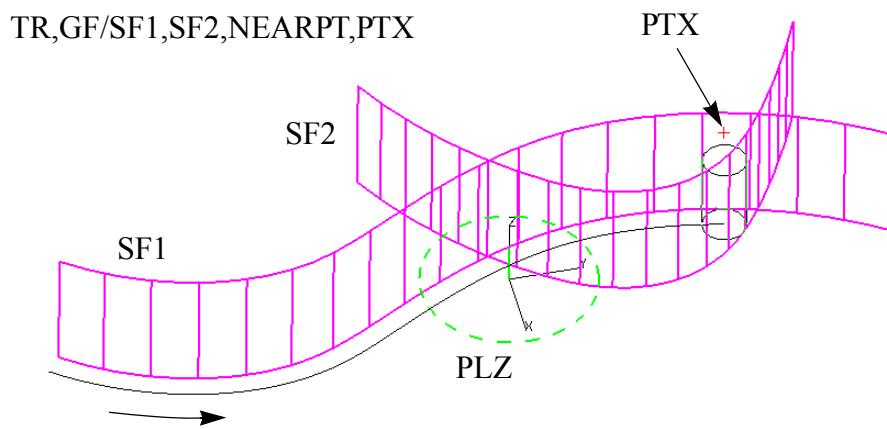
The following tool path illustrates using a point as a check surface:



6 CONTINUOUS MOTION STATEMENTS

6.13.4 Check Surface At A Near Point

NCL also has the ability to check to a surface at a near point. In this case, the tool will check to the check surface and then determine if the near point lies within a cylinder having its axis as the tool axis and its radius equal to the maximum radius of the tool. If it does not, the tool will continue forward expecting to find another intersection with the check surface where the near point condition is satisfied. The following tool path illustrates using a check surface at a near point.



6.13.5 Check Surface Tangent To A Surface

NCL has the ability to check to a tangent surface. The syntax is as follows:

```
GOFWD  
GOBACK/sf1,TANTO,sf2  
GOLFT      PSTAN  
GORGT  
GOUP  
GODOWN
```

Where “sf1” and “sf2” represent the name of a valid surface or plane.

Caveats:

- The tangency point is calculated at the contact point of the cutter.
- The modifier "PSTAN" can only be used when both the part surface and the check surface are of the entity type "**TRIMMED-SURFACE**". Other entity types will not be allowed.

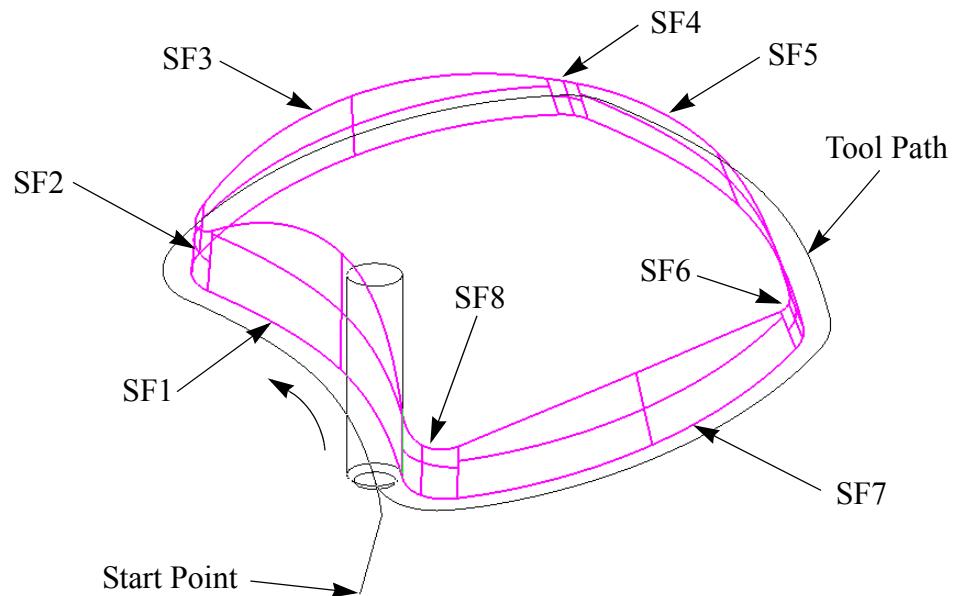
6 CONTINUOUS MOTION STATEMENTS

- The "tangent" feature will work reliably only when the tangent surfaces share a common edge. Trimmed surfaces are supported for this feature.
- Results can be unpredictable when the contact point of the cutter is on the extension of either of the control surfaces.

Example:

```
TLAXIS/COMBIN,.06,PARELM,.15
GO      /TO,SF1,PLZ,TO,SF7
IV      /VPY
TLLFT
GOFWD/SF1,TANTO,SF2
GOFWD/SF2,TANTO,SF3
GOFWD/SF3,TANTO,SF4
GOFWD/SF4,TANTO,SF5
GOFWD/SF5,TANTO,SF6
GOFWD/SF6,TANTO,SF7
GOFWD/SF7,TANTO,SF8
GOFWD/SF8,TANTO,SF1
```

Picture below shows the motion generated by above statements.



6 CONTINUOUS MOTION STATEMENTS

6.14

User Specified U And V Values For Curves And Surfaces

The user may set the "U" and/or "V" values that **NCL** uses to initially position the cutter to a CURVE or SURFACE. CURVES have only a U value associated with them. For example, U=0 is the start point of a CURVE and U=1 is the last point of a CURVE. SURFACES have both a U and a V value associated with them. For example, U=0 and V=0 is the first point of the first defining element of a surface. U=1, and V=1 is the last point of the last defining element of a surface. The U direction is along the defining elements of the surface, the V direction is perpendicular to the defining elements. Each time **NCL** processes a motion statement it must determine where the cutter contacts the control surfaces (*Drive Surface, Part Surface and Check Surface*). When CURVES and/or SURFACES are specified, **NCL** has traditionally began its search from the middle of the control surfaces (U=.5 for CURVES and U=.5, V=.5 for SURFACES). **NCL** would then try to make contact with the control surface at the first point where the control surface's normal (perpendicular) vector passes through the tool end point. In the case of U shaped or helical type CURVES and SURFACES this would often be an undesirable position which would cause **NCL** to *jump* through the surface, much to the user's dismay. To avoid this problem the user may specify where **NCL** is to start looking for the contact point, typically a value close to the current location of the tool.

The valid syntax for specifying where the tool is to look for a contact point on a curve or surface is:

CV[u] or SF[u,v]

For example:

GOFWD/SF1 [0,0],SF2 [.75,.25]

Would cause **NCL** to begin its search for the Drive Surface at U=0, V=0 and the Check Surface at U=.75, V=.25.

D1=DIST/PT1,SF4 [.8,1]
GOFWD/CV1 [.25],PAST,SF3 [.25,.5]

When CURVES are specified, only the "U" value should be given. U and V values must always be in the range of 0-1. This feature is only valid with the following statements.

PSIS

GOFWD, GOBACK, GOLFT, GORGT, GOUP, GODOWN

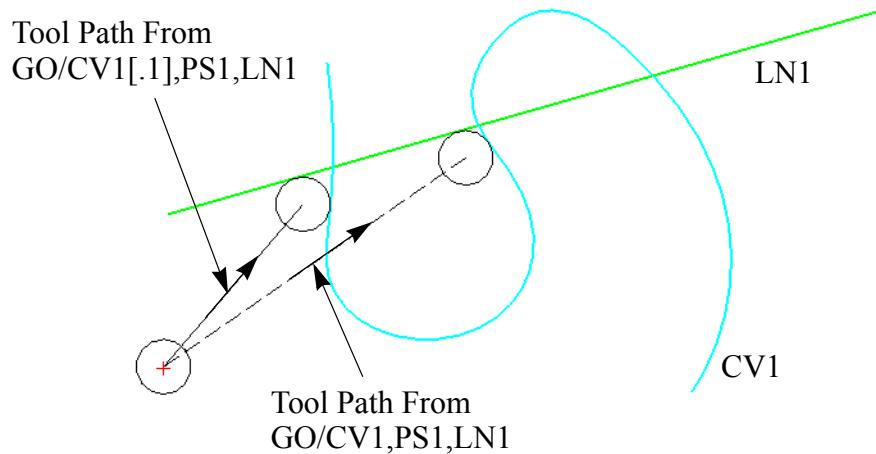
6 CONTINUOUS MOTION STATEMENTS

```
GO  
VECTOR/point,surface  
VECTOR/TANTO,curve,point  
DIST/point,surface
```

The following figure shows the result of a normal three-surface start up command attempting to position the tool at the first intersection of the curve CV1 and the line LN1 from the starting position at STPT.

The dashed line indicates the resultant move which is not as intended.

The problem is eliminated by specifying the approximate U distance (enclosed in square brackets) as a percentage value between 0 and 1 from the start of the curve.



6.14.1 Automatic Use Of Previous U And V Values During Continuous Path Motion

Generally, the manual setting of U and V values, as previously described, will only be necessary for initially positioning the tool to the control surfaces. This is because **NCL** automatically uses the U and V values calculated in the previous motion statement for the current motion statements. This will prevent the tool from *jumping* through U shaped, helical or 360 degree **CURVES** and **SURFACES** when processing a series of continuous path moves along such surfaces. The automatic tracking applies to the following commands:

GO , GOFWD , GOBACK , GOUP , GODOWN , GORGT , and GOLFT

6 CONTINUOUS MOTION STATEMENTS

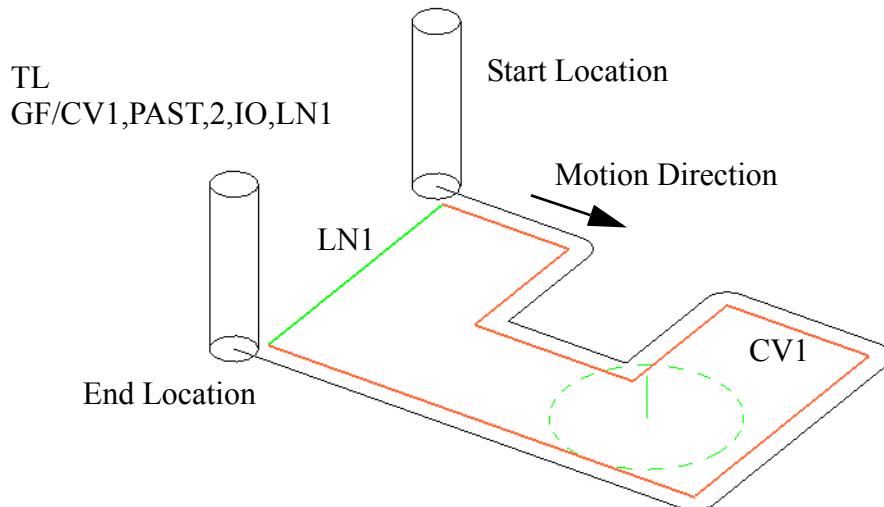
If any other motion command is processed between the above commands, the automatic tracking is discontinued until the next series of continuous path moves. It should be noted that programs processed with older versions of **NCL** may process differently. For this reason the user may disable or enable this feature with the following commands:

*RESET/AUTOUV
*SET/AUTOUV (default)

6.14.2 Driving A Composite Curve With Components Not Tangent To Each Other

NCL has the ability to drive a composite curve with any 3-D planar or non-planar components which are not tangent to each other.

When the tool condition is **TLLFT** or **TLRGT** **NCL** will create fillet-like motion hugging an outside corner, or “sharp-cornered” motion (with abrupt change of direction) when driving an inside corner. With **TLON**, the motion will follow the curve shape exactly (no fillet-like motion will be created).



Note:

1. Motion would be more likely to succeed if started some distance away from an inside corner (so that the correct curve projection could be found at the beginning).

6 CONTINUOUS MOTION STATEMENTS

2. The fillet-like motion is only created if using the **GOFWD**, **GOBACK**, **GOFWD**, **GOLFT**, **GOUP** and **GODOWN** commands. No fillet-like motion will be created if using the **GOFWDA** command.

6.15 GOBACK

The GOBACK statement generates tool motion from the current location along the drive surface to the check surface, in the opposite direction of the forward sense. The word GOBACK may be abbreviated as GB.

The valid syntax for the GOBACK statement is:

```
GOBACK/drive-surface          $  
      [,check-surface-modifier],check-surface]    $  
      [,feedrate]
```

Drive-surface and check-surface geometry may be lines, planes, circles, curves or surfaces.

The tool is assumed to be in contact with the drive surface at the start of the move. "TO" is the default check-surface-modifier.

Example GOBACK Statements:

```
GOBACK/PL1, TO, PL2  
TLRGT, GB/SF1, PL3, 10  
GB/LN1, PAST, CI1  
TLLFT, GOBACK/SF1, ON, SF2
```

6.16 GODOWN

When GODOWN is used the sense of direction established by the previous move is ignored. Instead, the tool axis is used to establish the sense of direction.

The valid syntax for the GODOWN statement is:

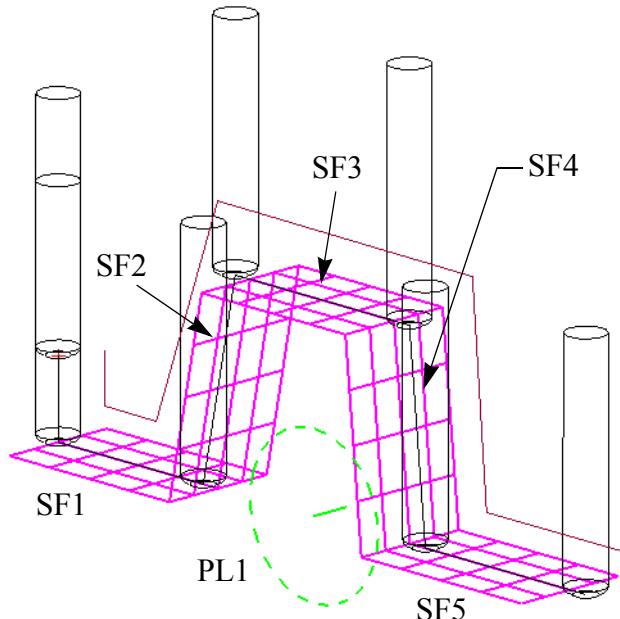
```
GODOWN/drive-surface          $  
      [,check-surface-modifier],check-surface]    $  
      [,feedrate]
```

6 CONTINUOUS MOTION STATEMENTS

GODOWN is used if the desired direction is along the negative tool axis (from top to bottom). GODOWN is also used when the desired direction has a component along the negative tool axis even though it is not parallel to the axis.

The example on next page shows how GODOWN and GOUP statements may be used in a typical application.

```
CU/.5,.1,3  
FROM/-4,-.5,1  
GO/ON,PL1,SF1  
IV/1,0,0  
GF/PL1,SF2  
PSIS/SF2  
GOUP/PL1,PAST,SF3  
PISS/SF3  
GF/PL1,PAST,SF4  
PSIS/SF4  
GODOWN/PL1,SF5  
PSIS/SF5  
IV/1,0,0  
GF/PL1,PAST,(PL1,0,0,4)
```



6.17 GOFWD

The GOFWD statement generates tool motion from the current location along the drive surface to the check surface, in the direction of the forward sense. The word GOFWD may be abbreviated as GF. The valid syntax for the GOFWD statement is:

```
GOFWD/drive-surface $  
[ [ , check-surface-modifier ] , check-surface ] $  
[ , feedrate ]
```

Valid drive-surface geometry may be lines, planes, circles, curves or surfaces. Valid check-surface geometry may be lines, planes, circles, curves, surfaces or points

6 CONTINUOUS MOTION STATEMENTS

The tool is assumed to be in contact with the drive surface at the start of the move. "TO" is the default check-surface-modifier.

Example GOFWD Statements:

GOFWD/PL1, TO, PL2, 30
TLRGFT, GF/SF1, PL3
GF/LN1, PAST, CI1
TLLFT, GOFWD/SF1, ON, SF2

6.18 GOLFT

The GOLFT statement generates tool motion from the current location along the drive surface to the check surface, in the general left direction from the forward sense. The word GOLFT may be abbreviated as GL. The valid syntax for the GOLFT statement is:

```
GOLFT/drive-surface $  
    [,check-surface-modifier [,check-surface]] $  
    [,feedrate]
```

Valid drive-surface geometry may be lines, planes, circles, curves or surfaces.
Valid check-surface geometry may be lines, planes, circles, curves, surfaces or points.

The tool is assumed to be in contact with the drive surface at the start of the move. "TO" is the default check-surface-modifier.

Example GOLFT Statements

GOLFT/PL1, TO, PL2
TLRGRT, GL/SF1, PL3, 25
GL/LN1, PAST, CI1
TLLFT, GOLFT/SF1, ON, SF2

6.19 GORGT

The GORGT statement generates tool motion from the current location along the drive surface to the check surface, in the general right direction from the forward sense. The word GORGT may be abbreviated as GR. The valid syntax for the GORGT statement is:

6 CONTINUOUS MOTION STATEMENTS

GORGT/drive-surface	\$
[, check-surface-modifier [, check-surface]]	\$
[, feedrate]	

Valid drive-surface geometry may be lines, planes, circles, curves or surfaces. Valid check-surface geometry may be lines, planes, circles, curves, surfaces or points.

The tool is assumed to be in contact with the drive surface at the start of the move. "TO" is the default check-surface-modifier.

Example GORGT Statements:

```
GORGT/PL1, TO, PL2
TLRGT, GR/SF1, PL3, 25
GR/LN1, PAST, CI1
TLLFT, GORGT/SF1, ON, SF2
```

6.20 GOUP

When GOUP is used, the sense of direction established by the previous move is ignored. Instead, the tool axis is used to establish the sense of direction.

The valid syntax for the GOUP statement is:

GOUP/drive-surface	\$
[, check-surface-modifier], check-surface	\$
[, feedrate]	

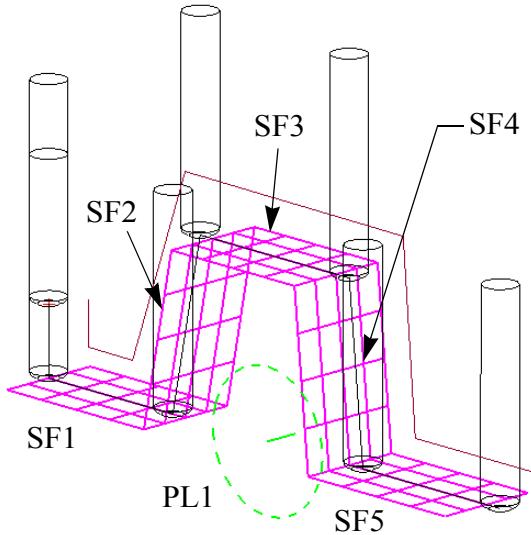
GOUP is used if the desired direction is along the *positive* tool axis (from bottom to top). GOUP is also used when the desired direction has a component along the *positive* tool axis even though it is not parallel to the axis.

The example on next page shows how GOUP and [GODOWN](#) statements may be used in a typical application.

```

CU/.5,.1,3
FROM/-4,-5,1
GO/ON,PL1,SF1
IV/1,0,0
GF/PL1,SF2
PSIS/SF2
GOUP/PL1,PAST,SF3
PISS/SF3
GF/PL1,PAST,SF4
PSIS/SF4
GODOWN/PL1,SF5
PSIS/SF5
IV/1,0,0
GF/PL1,PAST,(PL/1,0,0,4)

```



6.21 INDIRP

The INDIRP statement is used to set the forward sense for cutter motion. The word INDIRP may be abbreviated IP.

Example INDIRP Statements:

```

INDIRP / PT
INDIRP / 0 , 1 , 0

```

6.21.1 INDIRP/ point

The forward sense is calculated from the current tool location to the point specified.

6.21.2 INDIRP/ x, y, z

The three coordinates define a point which specifies the new forward sense.

6.22 INDIRV

The INDIRV statement is used to set the forward sense for cutter motion. INDIRV may be abbreviated as IV.

6 CONTINUOUS MOTION STATEMENTS

Example INDIRV Statements:

```
INDIRV/VE1  
INDIRV/.7,-5,-.34  
INDIRV/PV1
```

6.22.1 INDIRV/ vector

The vector specified becomes the new forward sense.

6.22.2 INDIRV/ x, y, z

The three coordinates define a vector which becomes the new forward sense.

6.22.3 INDIRV/ Point-Vector

The direction of the point-vector becomes the new forward sense.

6.23 TLLFT

The TLLFT statement specifies that the tool position shall be maintained on the left side of the drive surface with respect to the forward direction. TLLFT may be appended to the front of a GOFWD, GOBACK, GORGT or GOLFT motion statement. The word TLLFT may be abbreviated as TL. The valid syntax construct for the TLLFT statement is:

```
TLLFT
```

Example TLLFT Statements:

```
TLLFT  
TLLFT,GL/LN1,TO,LN2  
TL,GF/CI1,LN1
```

6.24 TLON

The TLON statement specifies that the tool position shall be maintained ON the drive surface. TLON may be appended to the front of GOFWD, GOBACK, GOLFT or GORGT motion statements. The word TLON may be abbreviated TN. The valid construct for the TLON statement is:

6 **CONTINUOUS MOTION STATEMENTS**

TLON

Example TLON Statements:

```
TLON  
TLON, GL/LN1, PAST, LN2
```

6.25

TLONPS

The TLONPS statement specifies that the tool end is to be maintained on the part surface. This statement will only apply to the GO, RMILL and all continuous path motion statements such as GOFWD, GORGT, GOFWDA, etc. This command will not apply to the GODLTA, FMILL, SCRUB and POCKET motion statements.

Example TLONPS Statement:

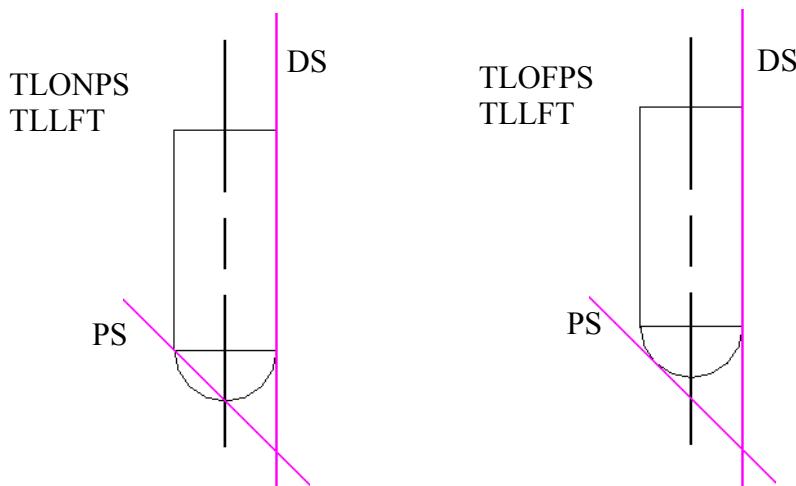
```
GO/SF1, ON, SF2  
TLONPS, TLLFT, GOLFT/SF1, SF3
```

6.26

TLOFPS

The TLOFPS statement specifies that the tool is to be offset from the part surface. This is the default setting.

The following illustration shows the relationship of the tool to the drive surface and part surface.



6 CONTINUOUS MOTION STATEMENTS

6.27 TLRGT

The TLRGT statement specifies that the tool position shall be maintained on the right side of the drive surface with respect to the forward direction. TLRGT may be appended to the front of **GOFWD**, **GOBACK**, **GOLFT** or **GORG** motion statements. The word TLRGT may be abbreviated as TR. The valid construct for the TLRGT statement is:

TLRGT

Example TLRGT Statements:

```
TLRGT  
TLRGT, GL/LN1, ON, LN2  
TR
```

Automatic Motion Routine Statements

This section deals with all the automatic motion routine available in **NCL**.

6.26 FMILL

The FMILL statement causes tool motion to be generated over a surface along the U or V direction, effectively milling the entire surface or a portion of it bounded by a user specified curve. If the surface specified is of the type "Trimmed", the underlying surface will not be used as the obsolete **SCRUB** command. Only the region of the surface bounded by the outside boundary and the inside boundary(ies) of the trimmed surface is used for this FMILL command.

The valid syntax construct for the FMILL statement is shown below.

```
FMILL/sff,HEIGHT,scallop-height          $  
    PASS,num-passes  
  
    [,CS,ccs1[,thk1],[ccs2[,thk2]][...]]      $  
  
    [,START,pt1][,FWD,0][,TOLER,tol][ OMIT,OFF ]      $  
        u,v           1     STEP,stp      START  
                                         END  
                                         BOTH  
  
    [,ON,curve][,RAPTO,p11[,fr1   ]]  
        sf1   RAPID  
        ds1  
        0  
  
    [,RETRCT,p12[,fr2   ]][,CONTCT][,AVOID,      $  
        sf3   RAPID      TO  
        ds2       ON  
        0       PAST  
  
        OFF  
        THRU  
        DOWN[,dir]  
    [BOTH,]p13[,FEDRAT,f3],p14[,FEDRAT,fr4]  
        sf3   RAPID      sf4   RAPID  
        ds3  
                                         ds4  
  
    [,STOP          ]  
    LAST[,FEDRAT,fr5]
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

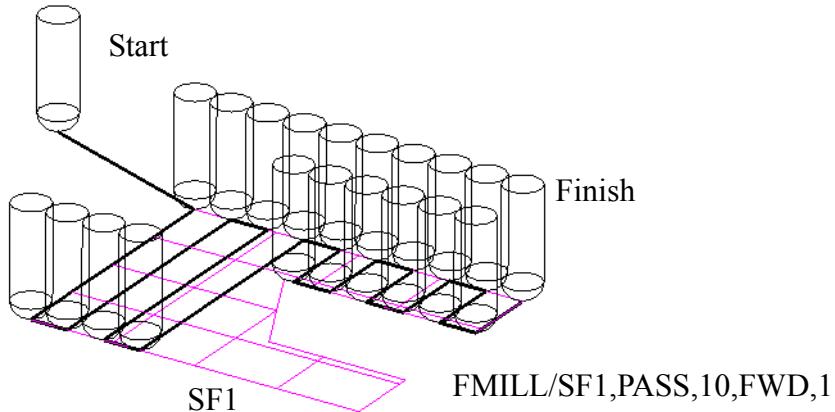
Where:

- "sff" is the surface to be milled and can be of any surface types except **QUILT**.
 - Net surface is allowed and it must be defined prior to issuing the FMILL command.
 - The "START" and "FWD" qualifiers will apply to the first component surface. Surfaces will be machined in order, starting with the first one. Beyond the first surface, the sequence of machining does not necessarily match the sequence used to define the net surface. The order is determined similar to the surface chaining logic: the algorithm attempts to continue each flowline beyond a surface boundary on to the next surface, trying to preserve the flowline direction.
- If "HEIGHT" is specified, the number of passes made will be calculated using the scallop-height specified. Otherwise, the number of passes must be specified using the "PASS" clause.
 - The "scallop_height" parameter can be any positive number, it is not restricted by the current machining tolerance. Note that for a ball-nose cutter a very small "scallop-height" translates into a very large number of passes (for other types of cutters the distance between passes is not less than the flat part of the cutter bottom).
 - One-pass cut is allowed. A one-pass cut can be specified either by specifying "PASS,1" or by specifying an appropriate "scallop-height".
- The "CS" option specifies the Avoidance Entities (ccs1,ccs2,...ccs40).
 - A flowline pass motion will stop at an Avoidance Entity, then move along it to the next flow and continue.
 - An Avoidance Entity can be any of the following geometry types: surface, plane, curve, line, circle.
 - Up to 40 Avoidance entities are allowed in a single FMILL command.
 - Each Avoidance Entity can have an optional thick parameter, i.e. "thk1" for "css1", "thk2" for "css2", etc.
 - The working assumption is that the FMILL motion starts on the "good" side of the Part Surface "sff". This is used to decide between two sides of an Avoidance Entity.
 - The cutting algorithm will not try to avoid an extension of a finite Avoidance Entity - only the entity itself. Planes and lines are considered infinite entities. Surfaces, curves and circles are considered finite entities.
- If "START" is specified, milling will begin nearest to the point or surface U and V values given. The default start position is the U = 0, V = 0 point on the surface.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- If "FWD" is specified, motion will be along the U flowlines of the surface if 0 is given or along the V flowlines if 1 is given. The default motion is along the U flowlines.
- If "TOLER" is specified, the value given will be used to create the points on the flowlines and on the surface boundary. If "STEP" is specified, the value given will be the number of points in each flowline and the current machining tolerance will be used to create the points on the boundary. If neither is given, the current machining tolerance will be used for both.
- "OMIT" specifies whether the first pass and/or the last pass can be eliminated. "OFF" is the default condition and no passed will be eliminated. "START" will eliminate the first pass. "END" will eliminate the last pass. "BOTH" will eliminate both the first and last passes.
- If "ON" is specified, the curve given will be projected normal onto the surface and used to limit the motion. Otherwise, the motion will be limited by the boundary of the surface. If a section of the curve is projected outside the surface, that section of the projected curve will not be used to limit the motion, instead the original surface boundary at that section will be used.
- If "RAPTO" is specified, an initial move will be made to a point above the first point on the surface. If a plane/surface is specified, the move will be on the plane/surface. If a distance is specified, the move will be made to that distance above the first point on the surface. If the "RAPTO" clause is omitted or a distance of 0 is given, no initial move will be made.
- If "RETRCT" is specified, a final move will be made to a point above the last point on the surface. If a plane/surface is specified, the move will be on the plane/surface. If a distance is specified, the move will be made to that distance above the final point on the surface. If the "RETRCT" clause is omitted or a distance of 0 is given, no final move will be made.
- If "CONTCT" is specified, the tool is in contact with the surface boundary and this is the default condition. If "ON" is specified, the tool end is positioned above the boundary, while the tool is in contact with the surface itself. If "TO" or "PAST" is specified, the tool is positioned on the inside or outside of the boundary, as if the boundary with this modifier were used as a Check Surface during each milling pass.
- The "AVOID" option describes the behavior with multiple flowline-boundary intersections for the outside boundary.
 1. "OFF" is the default. Motion will only be generated before the first intersection and no motion will be generated for areas after the first intersection. See example on next page.

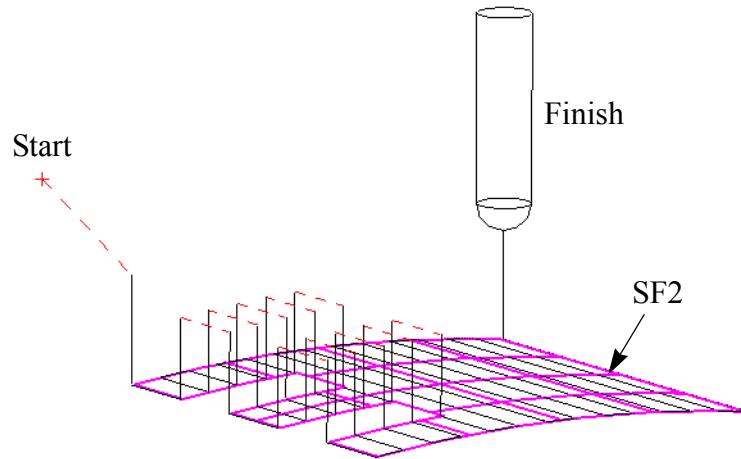
6 AUTOMATIC MOTION ROUTINE STATEMENTS



2. If "THRU" is specified, the intersections are ignored - each pass goes from the first intersection right through to the last one.
3. If "DOWN" is specified, when an intermediate boundary intersection is encountered, the pass leaves the flowline ($u=\text{const}$ or $v=\text{const}$), and takes the shortest way around the boundary until it is reached again at the next intersection, then the pass continues along the flowline.
4. If "BOTH" is specified, it works as the DOWN option (when an intermediate boundary intersection is encountered, the pass leaves the flowline and goes around the boundary until the flowline is regained), until the already cut part of the boundary is reached. Instead of going around the already cut pass (as it would with the DOWN option), the motion follows the intermediate boundary retract option: the cutter is retracted and moved to the other side of the boundary, where it is moved back to the surface.
5. If "DOWN" or "BOTH" is specified as the AVOID option, the direction of the cut along the boundary between two intermediate boundary intersections can be specified. The options are: SHORT, SAME, CCLW and CLW.
 - If "SHORT" is specified (the default option), the pass takes the shortest way around the boundary.
 - If "SAME" is specified, the shortest path is calculated only once for each boundary curve - each subsequent cut around the boundary is made toward the previously cut part of the boundary (so that if "BOTH" is specified the cutter stays down as little as possible).
 - If "CCLW" is specified, the cut goes counter-clockwise (as viewed along the tool axis) around the outer boundary and clockwise around the inner boundaries.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- If “CLW” is specified, the cut goes clockwise (as viewed along the tool axis) around the outer boundary and counter-clockwise around the inner boundaries.
- 6. If a plane “pl3”, a surface “sf3” or a distance ”ds3” is specified, when an intermediate boundary intersection is encountered, the tool will be retracted on the specified plane, to the specified surface or to a distance above the milled surface, then moved to the point above the next intersection, after which the tool is moved back on the milled surface and the pass continues.
- An optional retract feedrate can be specified as “FEDRAT,fr3” or “FEDRAT,RAPID” if “pl3”, “sf3” or “ds3” is specified. If it is not specified, the final RETRCT feedrate will be used if present in the command, otherwise the current feedrate will be used.
- The “pl4”, “sf4” or “ds4” specifies the subsequent RAPTO location when positioning back onto the milled surface. The tool will be rapid on the specified plane, to the specified surface or at a distance above the milled surface. This parameter must be specified whenever “pl3”, “sf3” or “ds3” is specified. This parameter is not allowed for “THRU” or “DOWN” specification.
- An optional “FEDRAT,fr4” or “FEDRAT,RAPID” can be specified for this RAPTO motion. If it is not specified, the final RAPTO feedrate will be used if present in the command, otherwise the current feedrate will be used. See example below:



```
FMILL/SF2,HEIGHT,.05,FWD,1,RAPTO,1,RETRCT,1 $  
50,AVOID,.75,FEDRAT,100,.75,FEDRAT,100
```

- If “STOP” is specified at the end, no final pass around boundaries is made.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- If “LAST” is specified at the end of the command, a final pass around the boundary is performed. This option is active only when “AVOID” is specified with “DOWN”, intermediate boundary retract, or “BOTH” options. If there is an inner boundary, the pass around it is performed when the tool touches it for the last time.
- An optional feedrate can be specified for this final pass motion as “FEDRAT, fr5”. If not specified, the current feedrate will be used.

Not all types of cutter and taxis modes can be used with the FMILL statement. Only a ball-end cutter (i.e. the corner radius is equal to half the diameter) or a flat-nosed cutter (i.e. the corner radius is less than half the diameter) can be specified with the FMILL command. The tool axis must either remain constant ([TLAXIS/ SAME](#)), be normal to the part surface ([TLAXIS/ NORMAL, PS](#)) or be normal to a secondary control surface ([TLAXIS/ NORMAL, PS, surface](#)). An error will occur if, when using a fixed tool axis, the portion of the surface being machined contains a normal that is greater than 90 degrees from the current tool axis.

If the tool axis is fixed, the motion boundary is determined by where the cutter contacts the boundary of the surface or the internal projected curve if a boundary curve is specified. It is not determined by the bottom center of the cutter.

If the tool axis is not fixed, the motion boundary is determined by the bottom center of the cutter, i.e. the motion will stop at where the bottom center of the cutter contacts the boundary of the surface or the internal projected curve if a boundary curve is specified.

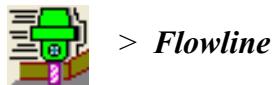
Note:

1. Generated motion for boundary with a narrow “bottle neck” spot could result in gouging of the part or incomplete milling.
2. For version prior to 9.6:
 - NET surface is not allowed.
 - the inner boundary(ies) of the trimmed surface will be disregarded if they ever exist.
 - “TLAXIS/ NORMAL, PS, surf” taxis mode is not allowed.
 - Fixed axis mode is not allowed for flat-nosed cutter.
 - Due to the changes in the motion algorithm, the generated motion might not be the same for program written prior to 9.6. See the [“*SET/VER, vflag”](#) command for compatibility details.
3. For version prior to 9.7.
 - Avoidance entity does not allow net surface.

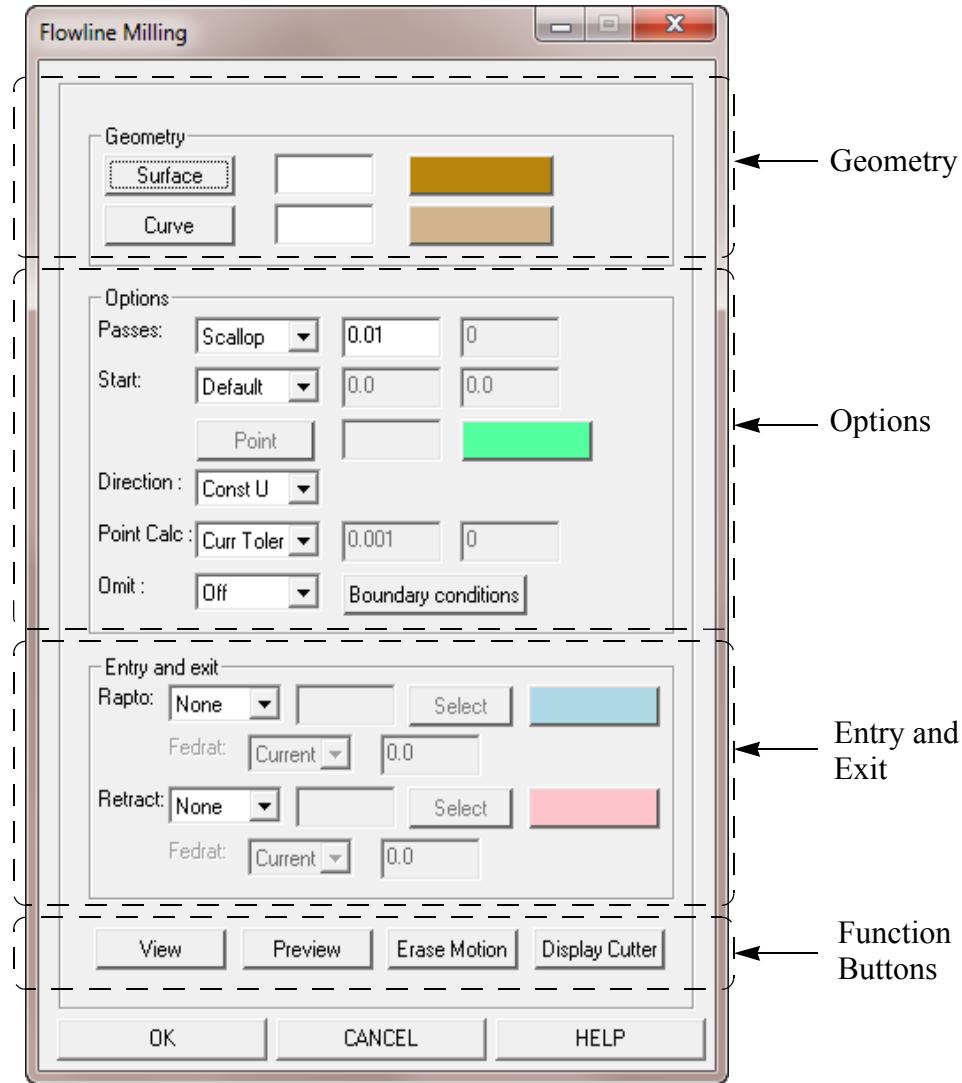
6 AUTOMATIC MOTION ROUTINE STATEMENTS

4. For version after 9.8.
 - FMILL allows continuous direction cutting when the start and end of surface(s) are the same (the surface(s) are closed). The toll will not reverse direction when it reaches the end of the surfaces, but will rather automatically transfer from the end to the beginning of the surface(s) and continue on in the same direction.

Following pages show the graphical interactive interface for this **FMILL** routine and a description of how to use this interface. This interface can be activated by using the following the following on screen menu icon sequences:



6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form composed of the following sections: Geometry, Options, Entry and Exit, and Function Buttons.

Geometry:

This section composed of two items: Surface, Curve They are described as follows:

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Surface:

Click to select the surface to be machined, or enter the surface name in the adjacent field. A selected surface will be highlighted with the color shown. This corresponds to the “sff” parameter of the command syntax.

Curve:

Click to select an optional curve to limit the motion or enter the curve name in the adjacent field. A selected curve will be highlighted with the color shown. If no curve is selected the motion will be limited by the surface outer boundary. This corresponds to the “ON,curve” parameters of the command syntax.

Options:

This section composed of five items: Passes, Start, Direction, Point Calc and Boundary Conditions. They are described as follows:

Passes:

Toggle between Scallop and Number. Select Scallop to have **NCL** calculate the number of passes using the scallop height specified in the next field. Select Number to enter the number of passes required. This item corresponds to either the “HEIGHT,scallop-height” or “PASS,num-passes” parameters of the command syntax.

Start:

This specifies where the milling motion will be started. Toggle between “Default”, “u,v” or “Point”.

• Default

The start position is the U=0, V=0 point on the milled surface. This corresponds to the “START,0,0” parameters of the command syntax.

• u,v

The milling motion will start near to the specified u and v values on the surface. This corresponds to the “START,u,v” parameters of the command syntax.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **point**

The milling motion will start near to the specified point. Pick the point by using the mouse or enter the point label into the blank space besides the activated point item. The highlight color can also be changed by toggle the color button at the right. This corresponds to the “START,pt1” of the command syntax.

Direction:

This specifies the milling direction and can be toggled between “Const U” or “Const V”.

- **Const U**

The milling motion will be along the U flowlines. This corresponds to the “FWD,0” parameters of the command syntax.

- **Const V**

The milling motion will be along the V flowlines. This corresponds to the “FWD,1” parameters of the command syntax.

Point Calc:

This specifies how the motion points will be created on the flowline and on the boundary. Toggle between “Curr Toler”, “Toler” or “Step”.

- **Curr Toler**

The current machining tolerances will be used both for the points creation along the flowline and the boundary. This is the default condition and there is no corresponding parameters of the command syntax.

- **Toler:**

Specifies to have the points calculated along each flowline and the boundary to be within the tolerance specified. This corresponds to the “TOLER,tol” parameters of the command syntax.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- Step

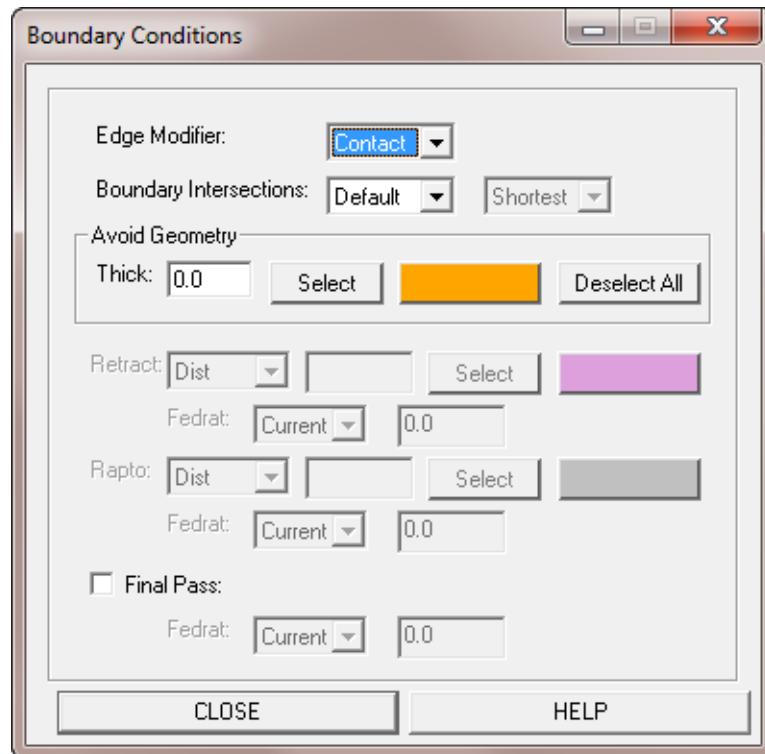
This specified the maximum number of points created along any flowline and the current machining tolerance will be used to create the points on the boundary. This corresponds to the “STEP,stp” parameters of the command syntax.

Omit:

OFF will keep all calculated passes of the FMILL motion, START will eliminate the first pass only, END will eliminate the last pass only, or BOTH will eliminate the first and last passes of the FMILL motion.

Boundary Conditions:

Click this button to specify how to generate motion for multiple intersections along the flowline with the outside boundary. The “Boundary Conditions” form as shown below will appear.



6 AUTOMATIC MOTION ROUTINE STATEMENTS

Edge Modifier:

This controls the relationship of the cutter to the outside boundary. Toggle between Contact, To, On, and Past.

- Contact**

Specifies the tool is in contact with the surface boundary. This is the default condition and corresponds to the “CONTCT” parameter of the command syntax.

- To**

Specifies the tool is positioned on the inside of the boundary, as if the boundary with this modifier were used as a Check Surface during each milling pass. This corresponds to the “TO” parameter of the command syntax.

- On**

Specifies the tool end is positioned above the boundary, while the cutter is in contact with the surface itself. This corresponds to the “ON” parameter of the command syntax.

- Past**

Specifies the tool is positioned on the outside of the boundary, as if the boundary with this modifier were used as a Check Surface during each milling pass. This corresponds to the “PAST” parameter of the command syntax.

Boundary Intersections:

The controls the motion behavior when there are multi-intersection between the flowlines and the outside boundary. Toggle between Default, Thru, Retract, Down and Both.

- Default**

Specifies motion will only be generated along the flowlines before the first intersection. There will be no motion generated after the first intersection. This is the default condition and

6 AUTOMATIC MOTION ROUTINE STATEMENTS

corresponds to the “AVOID,OFF” parameters of the command syntax.

- **Thru**

Specifies all the intersections are ignored. Motion will be generated from this end of the outside boundary to the other end of the outside boundary. This corresponds to the “AVOID,THRU” parameters of the command syntax.

- **Retract**

Specifies the tool will retract at the first intersection, move to the point above the next intersection and move down on the surface along the flowline. This is repeated until the other end of the outside boundary is reached. Specifies this option will activate the Retract and Rapto options.

- **Down**

Specifies when an intermediate boundary intersection is encountered; the pass leaves the flowline and takes the shortest way around the boundary until it is reached again at the next intersection, then the pass continues along the flowline.

- **Both**

Specifies when an intermediate boundary intersection is first encountered; the pass leaves the flowline and goes around the boundary until the flowline is regained. The next time the same boundary is encountered, instead of going around the already cut pass; the cutter is retracted and moved to the other side of the boundary, where it is moved back to the surface. Specifies this option will activate the Retract and Rapto options.

The boundary cut direction can be specified if “DOWN” or “BOTH” is specified. The choices are: Shortest, Same, CCLW or CLW.

- **Shortest**

The pass takes the shortest way around the boundary. This is the default condition.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Same**

The shortest path is calculated only once for each boundary - each subsequent cut around the boundary is made toward the previously cut part of the boundary (so that the cutter stays down as little as possible).

- **CCLW**

The cut goes counter-clockwise (as viewed along the tool axis) around the outer boundary and clockwise around inner boundaries.

- **CLW**

The cut goes clockwise (as viewed along the tool axis) around the outer boundary and counter-clockwise around inner boundaries.

Avoid Geometry:

A flowline pass motion will stop at an Avoidance Entity, then move along it to the next flowline and continue. An Avoidance Entity can be any of the following geometry types: surface, plane, curve, line, circle. Up to 40 Avoidance Entities are allowed in a single FMILL command.

- **Thick:**

The thick parameter to apply to selected Avoidance Entities. Each Avoidance Entity can have an optional thick parameter.

- **Select:**

Pick this button to select the geometric entity to be avoided. The currently set thick parameter will apply to the selected entity. Change the thick parameter before selecting the next entity.

- **Color:**

The avoidance geometry will be highlighted with the color shown.

- **Deselect All:**

Empty the list of Avoidance Entities for this FMILL command.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Retract:

This is only activated if the Boundary Intersections is specified as “Retract” or “Both”. This specifies how the tool retracts when the intersection is encountered the first time. This also specifies how the tool retracts for subsequent encountering of the same intersection when “Both” is specified. Toggle between Dist, Plane and Surface.

- **Dist**

Specifies the tool is retracted to a distance above the milled surface. This corresponds to the “ds3” parameter of the command syntax.

- **Plane**

Specifies the tool is retracted on a plane. This corresponds to the “pl3” parameter of the command syntax.

- **Surface**

Specifies the tool is retracted to a surface. This corresponds to the “sf3” parameter of the command syntax.

- **Text Field:**

Holds the label of the specified plane, surface or the distance value.

- **Select:**

Use this one to pick the specified plane or surface.

- **Color:**

Toggle this to select the highlight color of the selected plane or surface.

- **Fedrat:**

Specifies the feedrate value of the retract motion. Toggle between Current, Rapid and Value.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- | | |
|----------|---|
| Current- | Specifies the retract feedrate is same as the current feedrate value. |
| Rapid - | Specifies the retract is done at RAPID mode. This corresponds to the first “FEDRAT, RAPID” parameters of the command syntax after the AVOID parameter. |
| Value - | Specifies the retract feedrate is a user specified value. This activates the next Text Field to allow the user to enter the specified feedrate value. This corresponds to the “FEDRAT, fr3” parameters of the command syntax after the AVOID parameter. |

Rapto:

This is only activated if the Boundary Intersections is specified as “Retract” or “Both”. This controls how the tool move down on the surface on the next intersection after the retract motion. Toggle between Dist, Plane and Surface.

• Dist

Specifies the tool is moved to a distance above the milled surface on the next intersection along the flowline. This corresponds to the “ds4” parameter of the command syntax.

• Plane

Specifies the tool is moved on a plane above the milled surface on the next intersection along the flowline. This corresponds to the “pl4” parameter of the command syntax.

• Surface

Specifies the tool is moved to a surface above the milled surface on the next intersection along the flowline. This corresponds to the “sf4” parameter of the command syntax.

• Text Field:

Holds the label of the specified plane, surface or the distance value.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Select:**

Use this one to pick the specified plane or surface.

- **Color:**

Click this to open the Color Form to select the highlight color of the selected plane or surface.

- **Fedrat:**

Specifies the feedrate value of the Rapto motion. Toggle between Current, Rapid and Value.

Current - Specifies the Rapto feedrate is same as the current feedrate value.

Rapid - Specifies the Rapto is done at RAPID mode. This corresponds to the second “FEDRAT, RAPID” parameters of the command syntax after the AVOID parameter.

Value - Specifies the Rapto feedrate is a user specified value. This activates the next Text Field to allow the user to enter the specified feedrate value. This corresponds to the “FEDRAT, fr4” parameters of the command syntax after the AVOID parameter.

Final Pass:

Click this button to perform a final pass around the boundary. This option is active only when Down, Retract, or Both is selected for Boundary Intersections. If it is an inside boundary, the last pass around it is performed when the tool touches it for the last time.

- **Fedrat:**

Specify the feedrate for the final pass motion. Toggle between Current and Value.

Current - Specifies the final pass is done at the current primary feedrate.

Value - Specifies the final pass is done at a user specified value. This activates the next Text Field to allow the

6 AUTOMATIC MOTION ROUTINE STATEMENTS

user to enter the specified feedrate value. This corresponds to the “FEDRAT, fr5” parameters of the command syntax after the LAST parameter.

CLOSE:

Click this button to accept the entries and close the Boundary Condition form.

HELP:

Click this button to display a brief description of the Boundary Condition form.

Entry and Exit:

This section composed of four items: Rapto, Rapto Feedrate, Retract, Exit Feedrate. They are described as follows.

Rapto:

This controls how the milling motion start. Toggle between None, Dist, Plane and Surface.

• None

This is the default condition. No entry move is specified. This corresponds to the “RAPTO,0” parameters of the command syntax.

• Dist

The motion will start at a distance above the first point on the surface. This corresponds to the “RAPTO,ds1” parameters of the command syntax.

• Plane

The motion will start on the specified plane above the first point on the surface. This corresponds to the “RAPTO,pl1” parameters of the command syntax. The plane can be picked by the mouse or the plane label can be entered into the blank space. The highlight color of this specified plane can be changed by toggle the color button to the right.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Surface**

The motion will start on the specified surface above the first point on the surface. This corresponds to the “RAPTO,sf1” parameters of the command syntax. The surface can be picked by the mouse or the surface label can be entered into the blank space. The highlight color of this specified surface can be changed by toggle the color button to the right.

- **Text Field:**

Holds the label of the specified plane, surface or the distance value.

- **Select:**

Use this one to pick the specified plane or surface.

- **Color:**

Toggle this to select the highlight color of the selected plane or surface.

- **Entry Feedrate:**

Toggle between Current, Rapid, or Value.

Current - The entry move will be done at the current feedrate.

Rapid - The entry move will be done at a rapid feedrate. This corresponds to the “RAPID” parameter after the “RAPTO” parameter of the command syntax.

Value - The entry move will be done at the feedrate specified. This corresponds to the “fr1” parameter of the command syntax.

Retract:

This controls how the tool move after the milling motion. The four choices are None, Dist, Plane and Surface.

- **None**

This is the default condition. No final move is specified, the tool will stay at the final point on the surface. This corresponds to the “RETRCT,0” parameters of the command syntax.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Dist**

The motion will end at a distance above the final point on the surface. This corresponds to the “RETRCT,ds2” parameters of the command syntax.

- **Plane**

The motion will end on the specified plane above the final point on the surface. This corresponds to the “RETRCT,pl2” parameters of the command syntax. The plane can be picked by the mouse or the plane label can be entered into the blank space. The highlight color of this specified plane can be changed by toggle the color button to the right.

- **Surface**

The motion will end on the specified surface above the final point on the surface. This corresponds to the “RETRCT,sf2” parameters of the command syntax. The surface can be picked by the mouse or the surface label can be entered into the blank space. The highlight color of this specified surface can be changed by toggle the color button to the right.

- **Text Field:**

Holds the label of the specified plane, surface or the distance value.

- **Select:**

Use this one to pick the specified plane or surface.

- **Color:**

Click this to open the Color Form to select the highlight color of the selected plane or surface.

- **Exit Feedrate:**

Toggle between Current, Rapid, or Value.

Current - The exit move will be done at the current feedrate.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- | | |
|-------|---|
| Rapid | - The exit move will be done at a rapid feedrate. This corresponds to the “RAPID” parameter after the “RETRCT” parameter of the command syntax. |
| Value | - The exit move will be done at the feedrate specified. This corresponds to the “fr2” parameter of the command syntax. |

Function Buttons:

This section composed of four items: View, Preview, Erase Motion and Display Cutter. They are described as follows:

View:

Click to activate dynamic viewing.

Preview:

Click to preview the motion without output the FMILL command or data.

Erase Motion:

Click to erase the displayed motion.

Display Cutter:

Click to display the current cutter.

OK:

Click this button to accept the entries, generate the motion, output the FMILL command and close the Flowline Milling form.

CANCEL:

Click this button to cancel all the changes and close the Flowline Milling form.

HELP:

Click this button to display a brief description of the Flowing Milling form.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.27 GOFWDA

The GOFWDA statement will initialize a profiling routine to generate motion for driving a single composite curve or multiple drive surfaces.

6.27.1 GOFWDA For Driving A Single Composite Curve

```
GOFWDA / [LOOK, n, ] [AVOID, ] cmopc  
          FIND
```

Where:

- | | |
|-------|---|
| cmopc | Is a composite curve. In the case of a closed composite curve, the tool will stop at where it started. In the case of an open composite curve the tool will stop past a plane that is at the end and perpendicular to the curve. The composite curve will be driven in the direction at which it has been defined, thus the direction of the curve is significant. |
| LOOK | Specifies to look ahead “n” elements from the current element of the composite curve. If the next element cannot be successfully reached the system will look ahead as many as “n” elements for a successful combination. This is useful in cases where an element is too small for the cutter to reach because of a large value of thick or when using a large cutter for roughing. For example, a radius that is too small for the cutter to fit in to. |
| AVOID | Specifies that the system will ignore the extensions of composite curve elements. |
| FIND | Specifies that the system will respect the extensions of composite curve elements. |

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.27.2 GOFWDA For Driving Multiple Entities

```
GOFWDA/ds1, $  
[LOOK,n,] [AVOID,] [modifier,] [n,INTOF,] ds2 $  
    FIND  
  
[ ,NEARPT,pt] [ [ ,LOOK,n] [ ,AVOID] [ ,modifier] $  
    FIND  
  
[ ,n,intof],dsi[,NEARPT,pt]] [...] $  
[ ,CHECK, [AVOID,] [modifier,] [n,INTOF,] csi $  
    FIND  
  
[ ,NEARPT,pt,] [ [ ,AVOID] [ ,modifier] $  
    FIND  
  
[ ,n,INTOF] ,csi[,NEARPT,pt]] [...] ]
```

Where:

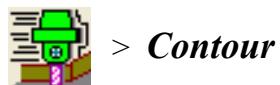
ds1 ... ds _i	Specify the drive surfaces, which can be any combination of lines, circles, curves, planes or surfaces. The rules governing standard motion apply for tangency. Up to 500 drives surfaces can be specified.
LOOK	Specifies to look ahead “n” drive surfaces from the current drive surface. If the next drive surface cannot be successfully reached, the system will look ahead as many as “n” drive surfaces for a successful combination. This is useful in cases where a surface is small for the cutter to reach because of a large value of thick or when using a larger cutter for roughing. “LOOK” stays in effect for all subsequent drives surfaces.
AVOID	Specifies that the system will ignore surfaces that are hit by the tool on the extension of the surface.
FIND	Specifies that the system will attempt to drive surfaces that are hit by the tool on the extension of the surface.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

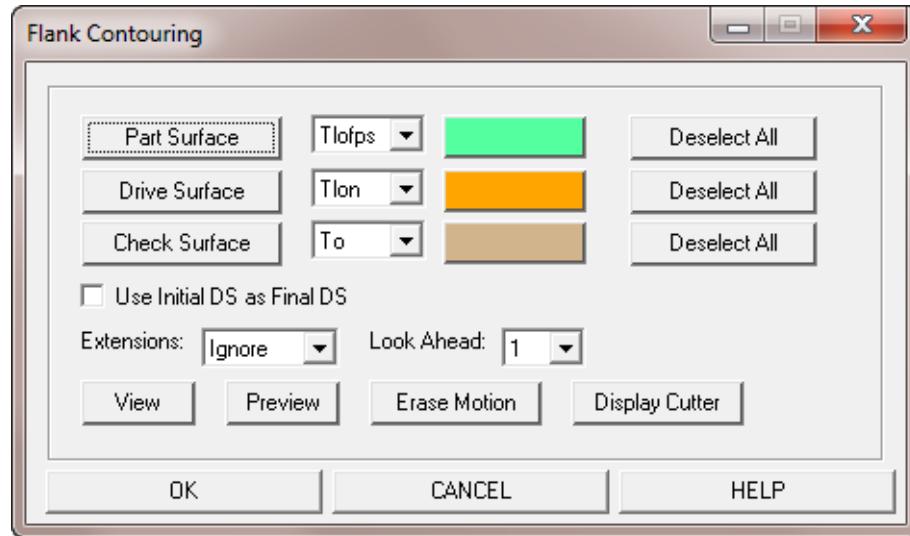
modifier	Check surface modifier: TO, PAST, ON, TANTO or PSTAN.
n,INTOF	Multiple intersection clause where “n” specifies which of the several possible intersections between the current drive surface and the next drive surface or the check surface at which the tool will stop. Specify 2 for the second intersection, 3 for the third intersection, and so on.
NEARPT,pt	This allows to specify a point near the intersection of the current drive surface and the next drive surface or the check surface at which the tool will stop. The near point is used in cases where many intersections of the control surfaces exist.
CHECK	Used to specify the final check surface(s). GOFWDA will end when the tool reaches any of the surfaces specified after CHECK. If CHECK is not specified, then the last specified drive surface will be used as the final check surface.
cs1 ... csi	Specify the final check surface(s). Up to 5 check surfaces may be specified and the tool will stop at the first one encountered.

It should be noted that while the command can be entered as shown, this statement is best produced by using the graphical interactive interface with which the user has the ability to preview the motion before generating the statement.

Following pages show the graphical interactive interface for this **GOFWDA** routine and a description of how to use this interface. This interface can be activated by using the following on screen menu icon sequences:



6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form composes the following buttons:

Part Surface:

Click this button to select the part surface. Multiple surfaces can be selected for the part surface.

Part Surface Condition:

The tool can either be positioned to the part surface (*TLOFPS*) or with the tool center always on the Part Surface (*TLONPS*).

Part Surface Color:

The color choice button specifies which color to use to highlight the part surface(s).

Deselect All:

The Deselect All button will deselect all currently selected part surfaces.

Drive Surface:

Click this button to select the surfaces to machine. These surfaces must be selected in the order in which you would like them machined. The Surface Chain method can be used for selecting the surfaces.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Tool Condition:

The tool can be positioned to the left of the drive surfaces (*TLLFT*), on the drive surfaces (*TLON*), or to the right (*TLRGRT*) in reference to the current forward direction of the tool.

Drive Surfaces Color:

The color button specifies which color to use to highlight the drive surfaces.

Deselect All:

The Deselect All button will deselect all currently selected drive surfaces.

Check Surface:

Click this button to select the final check surface. Up to five check surfaces can be selected.

Check Surface Condition:

The tool position for the final check surface can also be chosen. The tool conditions for intermediate drive/check surfaces are automatically calculated by the Contour routine.

Check Surfaces Color:

The color button specifies which color to use to highlight the check surface(s).

Deselect All:

The Deselect All button will deselect all currently selected check surfaces.

Use Initial DS as Final DS:

It is sometimes required that the initial drive surface be used as the last drive surface, for example when machining completely around a part and starting in the middle of a surface.

Since **NCL** does not allow the same geometry to be selected twice during a Selection operation and the nature of this form is to allow a geometry that

6 AUTOMATIC MOTION ROUTINE STATEMENTS

is selected a second time (during a separate selection process) to be deselected, enabling this field will automatically use the first selected drive surface as the last drive surface. If the initial drive surface is to be used as the final check surface, then do not check this box, but rather select this surface as the check surface also.

Extensions:

The drive surface extensions that are checked to by the tool can either be *Ignored* or *Respected*. It is typical to ignore the surface extensions when driving connected drive surfaces.

Look Ahead:

The tool can be instructed to look ahead up to five surfaces when driving each surface. This feature is useful when the tool may be too large to fit in some areas, so it will attempt to check to multiple surfaces, checking to the surface that it reaches first.

View:

Takes down the form and enters dynamic viewing.

Preview:

Previews the Contour motion. No command is output.

Erase Motion:

Erases all motion.

Display Cutter:

Displays the current cutter position.

OK:

Click this button to accept the form, generate the motion, output the command and close the form.

Cancel:

Click this button to disregard all the changes (if any) and close the form.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Help:

Click this button to open the online help for this Contour Form.

6.28 POCKETing

There are two routines in **NCL** that generate motion to clear out a bounded area (pocket). The simple pocket defined by a series of points can be cleared with the "APT" like POCKET syntax. More complex pockets with islands and perimeters defined with geometry other than points require the use of the advanced POCKET statements.

6.28.1 POCKET

The POCKET statement generates tool motion to clean out an area bounded by up to 20 points. The sides of the area are considered to be straight lines between the input points. Therefore, the minimum number of points is 3 and the points must be in sequence around the perimeter of the area. The pocket must not be "concave," i.e. a line drawn between any 2 of the defining points must lie within the pocket.

The bottom of the area is a plane or a surface of primitive type "**PLANAR**" specified by the last **PSIS** statement. This plane (or the planar surface) must be at an angle greater than 12 degrees to the tool axis of the tool.

Coverage testing logic is available to ensure that no islands will be left in the pocket. **Multax** may be ON but the tool axis must be in some "fixed" mode. The valid syntax construct for the POCKET statement is:

```
POCKET/radius,distance,finish-cut,plunge-feedrate, $  
      general-feedrate,finish-feedrate, $  
      coverage,type,point-list
```

Where:

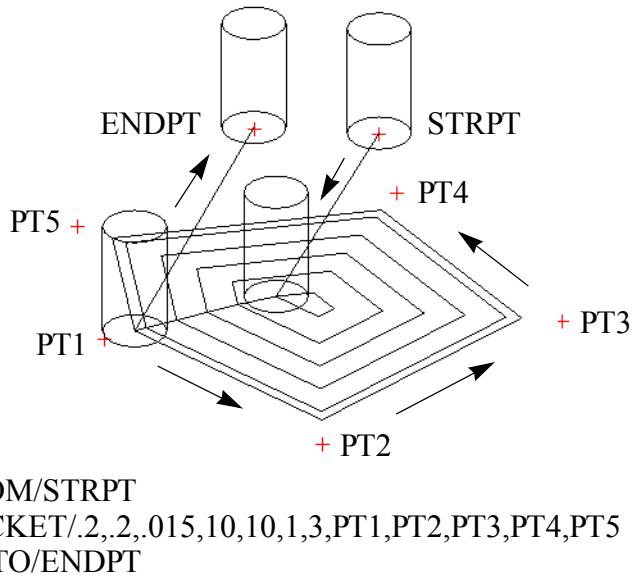
- | | |
|--------|---|
| radius | - specifies the effective radius of the tool for coverage testing. It should be as large as possible to eliminate unnecessary motion. This will normally be the radius of the cutter minus its corner radius. |
|--------|---|

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- | | |
|------------------|--|
| distance | - specifies the maximum distance between passes during pocketing. |
| finish-cut | - specifies the thickness of material for the final pass cut. If this value is zero, no finish pass will be done. |
| plunge-feedrate | - specifies the feedrate for the plunge into the pocket. |
| general-feedrate | - specifies the feedrate for the general clean out cutting. |
| finish-feedrate | - specifies the feedrate for the finish pass. |
| coverage | - is the coverage testing flag. NCL always does coverage testing. This parameter is included to provide compatibility with APT. Valid values are 0 and 1. |
| type- | - specifies the pocket type. If a 2 is entered, the input points are tool center locations. If a 3 is entered, the input points are the pocket vertices. |
| point-list | - is a list ranging from 3 to 19 points which define the outline of the area to be cleaned out |

Example POCKET Statements:

```
POCKET/.25,.5,.05,5,25,10,1,2,      $  
      PT1,PT2,PT3,PT4,PT5  
POCKET/.5,1,.1,10,10,10,1,2,PT1,THRU,PT19
```



6.28.2 Advanced POCKET

The Advanced POCKET feature will generate a set of NC motion instructions to remove material in a "spiraling in/out fashion" from within an irregular shaped perimeter with the inclusion of a number of islands. The spiraling is a series of concentric paths with a stepover between each path. There may be one or more "sections" of pocket motion generated from a single POCKET statement.

A section is a set of concentric paths that clears out an area that is part of the total area. A pocket may be broken up into sections if the routine determines that it is a more efficient way to clear the total area. A "dog-bone" or "dumbbell" type area is an example of this.

The following is the syntax and functionality of the Advanced Pocket feature.

The POCKET routine uses two types of **NCL** statements. The POKMOD statement sets modal values that apply to all subsequent Advanced POCKET type statements. The Advanced POCKET statement causes motion to be generated to clear out material in the POCKET.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.28.2.1 POKMOD

This sets values used by the Advanced POCKET motion generating statement. The valid syntax for this statement requires at least 20 parameters to be specified. The valid syntax of the POKMOD command is shown below.

```
POKMOD/num-ramp/rev/ent-ang,           $
entry-method, [CLW/CCLW, ]             $
[PERPTO/TANTO, ]                      $
[warning, ]                           $
ramp-dist/helical-rad,                $
[OUT,ARC, ]                           $
[ISLAND,ARC, ]                        $
RETRCT,ON/OFF                         $
clearance-level,[INCR, ]               $
step-down-dist,[DIST/DEPTH]           $
rapto-dist,[retrct-dist, ]             $
machining-method,                     $
retract-option,                       $
arc-corner-option,[IN,rad,][ATANGL,ang,] $
[TRANS,ARC, ]                         $
max-step-dist,                        $
min-step-dist,                        $
[HEIGHT,scallop, ]                   $
general-fedrat,                      $
position-fedrat,                     $
retract-fedrat,                      $
entry-fedrat,                        $
transition-fedrat,                  $
finish-fedrat                         $
[,first-pass-fedrat]                 $
[,CUTCOM,arg1,...,argn[,NOMORE]]    $
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Where:

num-ramp/rev/ent-ang

- A scalar defines the number of ramps/revolutions to use to reach the pocket depth, or the maximum entry angle (scalar):
- When "entry-method" is set to RAMP or HELIX.
 - a) A positive value indicates how many ramping moves or helix revolution to make until the current pocketing level is reached. The number will be rounded to the closest integer value. If the value is from zero to one, a single ramping move or helix revolution will be made.
 - b) A negative value specifies the maximum entry angle allowed. When this option is specified, the number of ramp moves or helix revolutions is the smallest such that the entry angle does not exceed the maximum entry angle specified. If the entry method is RAMP, the number of ramps is made even so that the entry motion always ends at the point where it started.

entry-method = RAMP, PLUNGE, HELIX, OMIT, COUPLE, CYCLE or a text string variable:

- RAMP specifies the cutter will make a series of linear ramping motions, starting at the retract level, until the current pocketing level is reached. This will output point-to-point entry motion to the CL file.

If SCRUB/LACE is specified as the machining method, the last ramp ends at the pocket entry point and moves in the direction opposite to the first cut.

- PLUNGE specifies the cutter will move directly down the tool axis to the current pocketing level when entering the pocket. This will output point-to-point entry motion to the CL file.
- HELIX specifies the cutter will make a spiraling move at a fixed radius until the current pocketing level is reached. The cutter will complete its helical move on the bottom of the pocket at the starting point of the spiral motion path. This will output point-to-point motion to the CL file.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

If SCRUB/LACE is specified as the machining method, the entry motion ends at a distance from the original entry point tangent to the perimeter (calculated to not gouge the pocket geometry), then the tool moves to the original entry point along the perimeter.

- OMIT specifies that the automatic entry move will be omitted.
- COUPLE will generate the same kind of motion as HELIX except the CL output is different. See Notes at end of this section for details.

```
GOTO/x1,y1,z1  
COUPLE/zi,ATANGL,360  
GF/(CIRCLE/xc,yc,0,0,0,1,d)  
GOTO/x2,y2,z2
```

where:

x1,y1,z1 - First point of the helical entry
zi - Depth per one helical rotation
xc,yc,zc - Center of the circle defined by projecting the entry helix to the pocket plane
d - Helix radius
x2,y2,z2 - Last point of the helical entry

- CYCLE will generate the same kind of motion as “HELIX”, but the CL output is different. See Helix Output as [CYCLE in POKMOD Command](#) section for details.
- “*ext String Variable*” will generate the same kind of motion as “HELIX”, but the CL output is different. See [Helix Output as Text String in POKMOD Command](#) section for details.

Note:

- i. For version prior to 9.6, when LACE/SCRUB is specified as the “machining-method”, only PLUNGE or OMIT entry-method will be allowed. If an entry method other than PLUNGE or OMIT is specified, it will be converted to PLUNGE.
- ii. Due to the changes in the behavior of the LACE/SCRUB entry method, the generated motion might not be the same for

6 AUTOMATIC MOTION ROUTINE STATEMENTS

program written prior to 9.6. See the “[*SET/VER, vflag](#) command for compatibility details.

CLW / CCLW:

- This is an optional parameter that specifies the orientation of the helix defined when utilizing the HELIX entry method. CCLW is the default condition if not specified.
- This option is ignored if “ANTO is specified with the HELIX entry method since the cut direction right after the entry will decide the helical direction.
- This option is ignored by all other entry methods.

PERPTO / TANTO:

- This is an optional parameter that specifies the direction of the entry relative to the first move of the pocketing motion. PERPTO is the default value.
- PERPTO specifies that the entry should be made so the forward direction of the cutter is perpendicular to the forward direction of the first move of the cut.
- “ANTO specifies that the entry should be made so the forward direction of the entry is in the forward direction of the first move of the cut when the cutter reaches the cutting plane.
- When TANTO is specified, the CLW/CCLW modifiers will be ignored since the direction of the first cut after the entry will decide the direction.
- This option is ignored by PLUNGE or OMIT entry method.

warning = WARN, NOWARN, AVOID or “AVOID,NOWARN”:

- This is an optional parameter, WARN is the default.
- WARN specifies **NCL** should check whether the entry path violates the pocket geometry. If such violations occur while **NCL** is run interactively, **NCL** displays the pocketing motion, generate the warning message *POCKET ENTRY VIOLATES GEOMETRY*, and stops.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- NOWARN specifies that **NCL** will not check for possible geometry violations during the pocket entry.
- AVOID specifies **NCL** will try to avoid geometry violations during pocket entry. In the case of RAMP or HELIX entry method, **NCL** tries to calculate an entry path that does not violate the pocket geometry. The ramp distance or helical radius may be reduced iff necessary and **NCL** will output a warning message if NOWARN is not specified. If such a valid path is not found, the pocket motion ends and **NCL** will output an error message. In the case of OMIT or PLUNGE entry, AVOID has the same meaning as WARN.

ramp-dist/helical-rad

- A scalar defines the Ramp distance or the /Helical radius:
- When Entry Method is set to HELIX, this value will indicate the radius generating the path the center point of the tool will follow during its downward helical entry.
- If this value is set to 0, it will default to the distance between the first point and second point of the pocket motion.
- A negative value is not allowed.
- This parameter is ignored if the Entry Method is set to PLUNGE or OMIT.

“OUT, ARC:”

- These are optional parameters that will cause **NCL** to generate a 90 degrees arc exit off the pocket instead of the default exit along a straight segment from the pocket plane at 45 degrees to the last perimeter move.
- The height and the radius of the helical arc move or the length of the side of the 45-degree linear move is defined by the **retract-dist** parameter.
- If the Machining-Method is set to LACE or SCRUB, these parameters will be ignored. Only straight exit move will be allowed.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

"ISLAND, ARC:"

- These are optional parameters that will cause **NCL** to generate a 90 degrees arc entry and exit motion for pocket islands.
- If the Machining-Method is set to LACE or SCRUB, these parameters will be ignored. Only straight plunge in entry and exit move will be allowed.

RETRCT, ON/OFF/STEP:

- When ON is specified, the tool will automatically be retracted to the clearance plane at the end of the pocketing motion.
- When OFF is specified, the tool will remain at the pocket floor bottom at the end of the pocketing motion.
- When STEP is specified, the tool will still retract to the clearance plane for all intermediate retract moves, but not for the final retract. On the final retract, the tool will retract from the part using the *retrct-dist* values, but will not retract to the clearance plane.

clear-level = Clearance level (scalar or plane):

- Distance/plane specified from the top plane of the current pocket that will be used as the Z-level the tool is:
 - retracted to upon exiting a pocket section.
 - at when positioning before entering pocket.
- A PLANE (or a SURFACE of primitive type **PLANAR**) geometry identifier may be given.

"INCR"

- Specifies that the tool will not be retracted to the clearance plane between each level of a multiple level pocket. The tool will instead traverse above the top plane for the current level by the clearance distance for successive pocketing levels.
- The clearance distance cannot be specified as a plane when INCR is used.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

step-down-dist = Distance between each cutting level (scalar):

- Used to calculate the number of pocketing levels (passes) to machine. Each cutting level will be defined using the plane defining the pocket bottom and will be offset by the step-down distance.
- If set to a positive number:
 - a) When *DIST* is specified, this value will be divided into the distance between the top and bottom planes to determine the number of levels. The step-down distance will be modified to the nearest number of levels if it does not divide evenly into the total distance between the top and bottom planes. This is the default condition.
 - b) When *DEPTH* is specified, the step-down distance specifies the maximum depth of cut for each level. The number of cutting levels is calculated as the largest that does not exceed this specified maximum depth of cut value.
- If set to a negative number, the step-down distance defines the number of cutting levels to use. The actual distance between levels will be the total distance between the top and bottom planes divided by the number of levels.
- If set to 0, the step-down distance will be set to half the current cutter.

rapto-dist = Rapto distance (scalar):

- Defines the height above the cutting plane he tool will move to down the tool axis at the retraction feed rate. At the rapto-dist height, the cutter will enter the pocket using the specified entry method and the entry feed rate.
- The rapto-dist is used as the default value for the retract-dist parameters.
- A negative value is not allowed.

retrct-dist = Retract distance(s) (scalar)

- Specifies the retract parameters when the tool is made away from the wall and off the floor prior to retracting the tool to the clearance plane. A negative value is not allowed.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Linear Exit Type: hretdis, [vretdis,]

- *i* specifies the horizontal distance of the side of the 45-degree linear move that is made away from the wall.
- *vretdis* specifies the distance off the floor along the tool axis of the 45-degree linear move that is made away from the wall. The default value is *hretdis* if not specified.

Arc Exit Type: radius, [vretdis,]

- *radius* specifies the radius of helical arc move that is made away from the wall.
- *vretdis* specifies the distance off the floor along the tool axis of the helix move that is made away from the wall and off the floor. The default value is *radius* if not specified.

machining-method = Specifies how the pocket is machined. The options are:

1. “[COLAPS,] pocket-direction, spiral direction”
 - COLAPS is an optional parameter. This will cause **NCL** to generate a standard collapsing pocket motion.
 - pocket-direction (CCLW or CLW) indicates the direction of the spiraling pocketing motion.
 - spiral direction which can be:
 - “OUT” specifies the tool will start in the center of each pocket section and work its way out towards the periphery.
 - “IN” specifies the tool will start at the periphery of each pocket section and work its way in towards the center,
2. “**LACE** , **cut-direction** [, **FINISH** , **dir**] [, **BOTH** [, **bndir**]]
SCRUB
 - **NCL** will generate a lace or scrub style pocket motion.
 - *cut-direction* specifies the general direction the cutter will follow:
 - POSX - the positive X direction

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- NEGX - the negative X direction
 - POSY - the positive Y direction
 - NEGY - the negative Y direction
 - LONG - along the long side of the smallest-area rectangle around the perimeter
 - SHORT - along the short side of the smallest-area rectangle around the perimeter
 - vector - along a vector or point-vector geometric entity
- FINISH applies to the perimeter final pass. For a pocket with islands, the final pass around islands is always in the direction opposite that of the perimeter. The direction modifier can be one of the following:
- SAME - along the last cutting direction
 - CCLW - counter-clockwise
 - CLW - clockwise
 - OMIT - no final pass is made
 - REVERS - reverse of the last cutting direction
- If the optional parameter BOTH is not specified, the tool will move along the boundary until the pass is regained if a cutting pass encounters an intermediate boundary intersection. This way some parts of the pocket boundaries could be traversed several times. This is the default setting.
- BOTH is an optional parameter and specifies that the tool is retracted when going around a boundary it reaches a section that has already been cut. When retracted, the tool is repositioned above the beginning of the next uncut portion of the boundary and then reenters the cutting level.
- The modifier *bndir* specifies the direction of the cut along the boundary between two intermediate boundary intersections. This can be one of the following:
- SHORT - The pass takes the shortest way around the boundary. This is the default setting.
 - SAME - The shortest path is calculated only once for each boundary curve and each subsequent

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- cut around the boundary is made toward the previously cut port of the boundary (so that the cutter stays down as little as possible).
- CCLW - The cut goes counter-clockwise (as viewed along the tool axis) around the perimeter and clockwise around islands.
 - CLW - The cut goes clockwise (as viewed along the tool axis) around the perimeter and counter-clockwise around islands.
 - REVERS - The final pass will go in the opposite direction of the last lace/scrub move.

NOTE:

- i. When the LACE/SCRUB is specified, the POCKET command can have an optional start-point at the end:

```
POCKET/bottom,top,dir-mod,perimeter-geo, $  
      dir-mod,island-geo,..... $  
      [,START,start-point]
```

- ii. If the starting point is specified, the cutting starts as close to it as possible. In particular, if the lace-direction is given as LONG or SHORT, the starting point will be at the closest point on the perimeter-enveloping rectangle to the start point. If the lace-direction is POSX, the starting point will be at either the max-Y or min-Y level, depending on which is close to start-point.
- iii. If LACE/SCRUB is in effect and the POCKET command ends with the traditional [END, end-point] and/or [,max-loops] options, these options will be ignored.
- iv. The LACE/SCRUB mode is respected by the Waterline Pocket command, however the command syntax has not been changed, so a start point cannot be specified in this command.
- v. If *IN, corner-rad* is specified, it has no effect for the LACE mode, and is used to fillet corners for the SCRUB mode.
- vi. When LACE/SCRUB is specified, only a PLUNGE or OMIT entry method will be allowed. If an entry method other than PLUNGE or OMIT is specified, it will be converted to PLUNGE.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- vii. When LACE/SCRUB is specified, only straight exit moves will be allowed. All other exit method will be ignored.
- viii. When LACE/SCRUB is specified, only straight plunge entry and exit move will be allowed for islands. All other entry/exit method will be ignored.

retract-option = Tool position between pocket sections (UP or DOWN):

- Specifies the tool position between sections.
- UP specifies the tool is to be retracted to the clearance plane, repositioned, and enter the next pocket section using the specified entry method and associated parameters' values.
- DOWN specifies that the tool will migrate to the various sections without retraction when possible. If DOWN is specified and the tool cannot move to a section without violating the periphery or an island, the UP option will be used for that section only.

arc-corner-option = Machine outside corners with arcs (SHARP or ARC):

- Specifies the corner handling method for the pocket boundary.
- ARC specifies motion at sharp corners on the periphery will be rounded to generate a circular move around the sharp corners. The angle of the corner must be less than 130 degrees.
- "SHARP" indicates sharp/outside corners will not be rounded.

"IN, rad":

- Specifies to generate filleting motion around inside corners. This is unlike the arc-corner-option for outside corners, the filleting is done after the motion is calculated, so the radius is the same for all the loops.
- *rad* specifies the fillet radius to be generated.

"ATANGL, ang":

- Specifies the secondary feedrate to apply to the last pass around the pocket perimeter and islands. The secondary feedrate will be applied when the angular change of direction is greater than or equal to *ang*.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

TRANS, ARC:

- These are optional parameters.
- This will cause **NCL** to generate a “double arc” transitions between concentric loops. As a result, the cutter will arc off the current loop and onto the next loop so the cutter exits the current loop tangent to the loop and enters the next loop tangent to the loop.

max-step-dist = Maximum stepover distance (scalar):

- Indicates the maximum distance between concentric motion loops.
- If set to 0, the current effective cutter diameter will be used. This is the cutter diameter minus twice the corner radius. This is calculated based on the cutter parameters in effect at the time the POCKET statement is processed, not at the time the POKMOD statement is processed.

min-step-dist = Minimum stepover distance (scalar):

- Indicates the minimum distance between concentric motion loops.
- If set to 0, the system will use the default of .1 of the "Maximum stepover distance" value. The minimum step-over distance can be used to increase the likelihood of complete coverage. The smaller the value, the better chance no areas will be left uncut. A small value can greatly increase the number of concentric paths.
- The minimum step-over distance must be at least five times larger than the TOLER value.

HEIGHT, scallop = Maximum height of left over material (scalar):

- This is only an optional parameter that is valid only when the cut method is specified as COLAPS. The height given specifies the maximum height for material not removed between concentric loops.
- *scallop* specifies the maximum height. Internally this value is restricted to be in the range between zero and the corner radius of the tool.
- This is ignored if the tool does not have a corner radius.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

general-fedrat = General pocketing feed rate (scalar):

- The feed rate used for generate material cutting motion in the pocket.
- The general feedrate will also be used as the base value for other feed rate parameters. It can be a multiplied times factor as explained below.
- A negative value is not allowed.
- A value of zero (0) causes the feed rate in effect at the time the POCKET statement is processed to be used.

position-fedrat = Positioning feed rate (RAPID or scalar):

- Specifies the feed rate at which the tool will travel while moving at the clearance level.
- If a positive number is specified, it will be used as the positioning feed rate.
- If a negative number is specified, it will be considered as a factor of the *general-fedrat*. The value given will be multiplied by the *general-fedrat* and the results used for the positioning feed rate.

retract-fedrat = Retract feed rate (RAPID or scalar):

- Specifies the feed rate at which the tool will travel while retracting to the clearance level and moving to the rapto plane.
- The same rules as for *position-fedrat* apply for positive and negative values.

entry-fedrat = Entry feed rate (scalar):

- Specifies the feed rate at which the tool will travel while entering a pocket section.
- The same rules as for *Position-fedrat* apply for positive and negative values.
- A value of zero (0) causes the *General-fedrat* to be used.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

transition-fedrat = Transition feed rate (scalar):

- Specifies the feed rate at which the tool will travel while moving to the next concentric loop.
- The same rules as for *Positioning-feedrat* apply for positive and negative values.
- A value of zero (0) causes the *General-feedrat* to be used.

finish-fedrat = Finish pass feed rate (scalar):

- Specifies the feed rate at which the tool will travel while making the final pass around the periphery and islands.
- The same rules as for *Position-feedrat* apply for positive and negative values.
- A value of zero (0) causes the *General-feedrat* to be used.

first-pass-fedrat = "First pass" feedrate (scalar):

- Specifies the feed rate at which the tool will travel while making the first pass around the periphery and islands.
- If a positive number is specified, it will be used as the first pass feed rate.
- If a negative number is specified, it will be considered as a factor of the *general-feedrat*. The value will be multiplied by the *general-feedrat* and the results used for the first pass feed rate.
- A value of zero (0) causes the RAPID feed rate to be used.

"CUTCOM,arg1,...,argn,NOMORE" = "Finish pass" Cutter compensation:

- These optional parameters cause **NCL** to output a "CUTCOM/arg1,...,argn" postprocessor statement to the CL file just before the tool making the pass around the periphery and/or each island.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- "arg1,...,argn" are parameters (separated by comma) used to specify the CUTCOM condition which may compose of valid postprocessor vocabulary words, scalers or scalar variables.
- NOMORE is an optional parameter to denote the end of the CUTCOM parameters.
- A CUTCOM/OFF postprocessor statement will immediately output to the CL file to reset the CUTCOM condition after the geometry profiling around the periphery and/or each island.

Wherever a "scalar" is allowed in these parameters, a variable scalar name may be given. If it is, the variable will be evaluated at the time the POKMOD statement is processed and its value used as the modal setting. The name of the variable is NOT passed or used when processing the POCKET statement, so if a variable's value is changed between the POKMOD and the POCKET statements, the change will not be reflected in the processing of the POCKET statement.

Following pages show the graphical interactive interface for this **POKMOD** routine and a description of how to use this interface. This interface can be activated by using either one of the following on screen menu icon sequences:



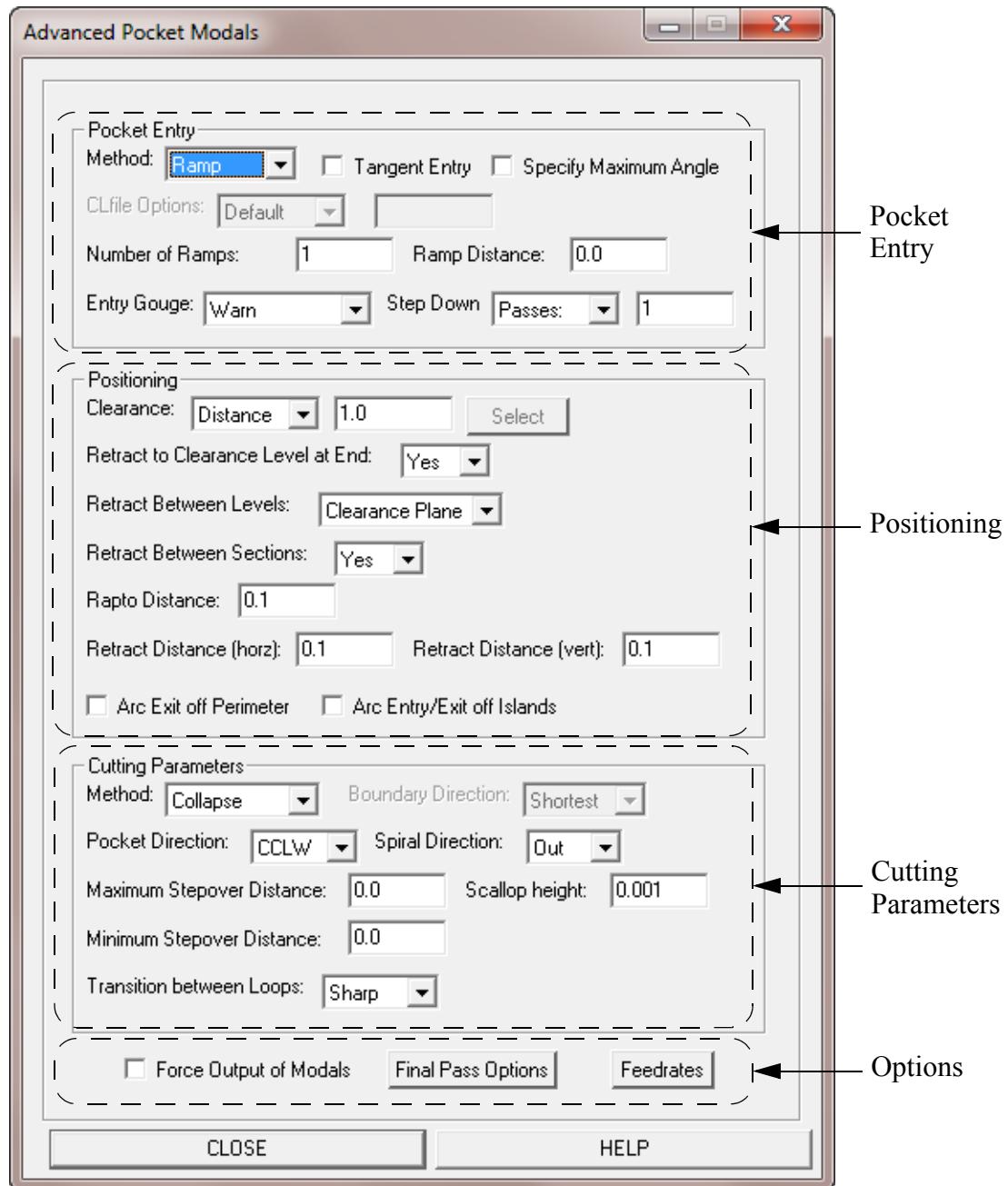
> **Pocket > Pocket Modals**

or



> **Waterline > Pocket Modals**

6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form composed of the following sections: Pocket entry, Positioning, Cutting Parameters and Options.

Pocket Entry:

This section specifies how the tool enter the pocket.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Method:

This defines the tool entry method used. Toggle between “Ramp”, “Helix”, “Arc”, “Plunge” or “Omit”.

- **Ramp**

The tool will make a series of ramping motions to enter the pocket.

- **Helix**

The tool will make a counter-clockwise spiraling move specified by number of revolution or maximum helical angle to enter the pocket.

- **Helix CLW**

The tool will make a clockwise spiraling move specified by number of revolution or maximum helical angle to enter the pocket unless Tangent Entry is selected (for the tangent helical entry the spiral direction is determined by the first cut).

- **Arc**

The tool will use a counter-clockwise helical move specified by number of degrees to enter the pocket. If this method is specified, the input degrees will be converted to number of revolutions and **NCL** will output it as HELIX to the POKMOD command.

- **Arc CLW**

The tool will use a clockwise helical move specified by number of degrees to enter the pocket unless Tangent Entry is selected (for the tangent helical entry the spiral direction is determined by the first cut). If this method is specified, the input degrees will be converted to number of revolutions and **NCL** will output it as HELIX to the POKMOD command.

- **Plunge**

The tool will move straight down the tool axis until the current pocketing level is reached.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Omit**

The tool will not perform an initial entry move into the pocket.

Tangent Entry:

Active only for Ramp, Helix, or Arc entry methods, results in an entry along the first cut for Ramp entries and tangent to the first cut for Helix or Arc entries.

Specify Maximum Angle:

Specifies the ramp/helix maximum angle as an alternative to the number of ramps/revolutions if the Ramp or Helix entry method is specified.

CLfile Options:

Specifies the optional format for CLfile output if the Entry Method is HELIX or Arc. Toggle between “Default”, “COUPLE”, “CYCLE” or “Textvar”.

- **Default**

The helical entry is represented as a sequence of GOTO points along the helix in the CLfile.

- **COUPLE**

The helical entry is represented by the following commands in the CLfile.

```
GOTO/x1,y1,z1  
COUPLE/zi,ATANGL,360  
GOFWD/ (CIRCLE,xc,yc,zc,0,0,1,d)  
GOTO/x2,y2,z2
```

where:

- | | |
|----------|---|
| x1,y1,z1 | - First point of the helical entry |
| zi | - Depth per one helical rotation |
| xc,yc,zc | - Center of the circle defined as the projection of the entry helix onto the pocket top plane |
| d | - Helix radius |

6 AUTOMATIC MOTION ROUTINE STATEMENTS

x2,y2,z2 - Last point of the helical entry, also the first point of the pocketing motion

- **CYCLE**

See [Helix Output as CYCLE in POKMOD Command](#) section for details.

- **Textvar**

See [Helix Output as Text String in POKMOD Command](#) section for details.

Number of Ramps:

Specifies the number of ramping moves to output when the entry method is set to Ramp.

Number of Revolutions:

Specifies the number of revolutions to output when the entry method is set to Helix.

Maximum Ramp Angle:

This field is only displayed when the Specify Maximum Angle box is checked and contains the maximum entry angle allowed for Ramp entries. The number of ramp moves is the smallest such that the entry angle does not exceed the maximum entry angle specified. The number of ramps will be made even so that the entry motion always ends at the point where it started.

Maximum Helix Angle:

This field is only displayed when the Specify Maximum Angle box is checked and contains the maximum entry angle allowed for Helix entries. The number of helical revolutions output is the smallest such that the entry angle does not exceed the maximum entry angle specified.

Arc Size:

Active only with the Arc entry method, specifies the angular size of the entry arc in degrees. The input number in degrees will be converted to

6 AUTOMATIC MOTION ROUTINE STATEMENTS

number of revolutions and **NCL** will output it as number of revolution to the POKMOD command.

Ramp Distance:

Active only with the Ramp entry method, Defines the length of the ramp motion at each level. A value of zero uses the distance between the first and second point of the pocket motion as the ramp distance.

Helix Radius:

Active only with the Helix entry method, defines the radius of the helix. A value of zero uses the distance between the first and second point of the pocket motion as the helix radius.

Arc Radius:

Active only with the Arc entry method, defines the radius of the arc. A value of zero uses the distance between the first and second point of the pocket motion as the arc radius.

Entry Gouge:

Specifies what action to be taken when a pocket entry violates the pocket geometry. Toggle between "Warn", "NoWarn", "Avoid" or "Avoid-NoWarn".

- Warn**

Warn the user when a pocket entry violates the pocket geometry, but will not modify the entry motion in any way.

- NoWarn**

Not output any warning when the pocket geometry is violated. This method is normally used when the pocket islands are actually "holes" in the part and cannot be physically cut by the pocket motion.

- Avoid**

Attempt to avoid the pocket geometry when it is gouged, first by altering the entry location and then if the entry method is Ramp, Helix, or Arc, by decreasing the ramp distance or circular radius if

6 AUTOMATIC MOTION ROUTINE STATEMENTS

necessary. A warning will be output if the entry move was altered to avoid the pocket geometry.

- **Avoid-NoWarn**

Same as Avoid, but a warning will not be output when the entry move is altered to avoid the pocket geometry.

Step Down:

Specifies how to determine the number of pocketing levels (passes) to machine in a pocket. Toggle between “Passes”, “Distance” or “Depth”.

- **Passes**

Specifies the actual number of passes to take.

- **Distance**

The specified value will be divided into the distance between the top and bottom planes to determine the number of levels. This number will be rounded to the nearest integer and divide that into the Z-difference to determine the actual distance between each level. In other words, this is the approximate step distance between levels. This is the default condition.

- **Depth**

Specifies the maximum depth of cut for each level. The number of cutting levels is calculated as the largest that does not exceed this specified maximum depth of cut value.

Positioning:

This section specifies the positioning of the cutter before entry of the pocket, between levels and exit of the pocket.

Clearance:

Specifies the clearance level when exiting a pocket section and positioning to a new section. Toggle between “Distance” and “Plane”.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Distance**

The distance value is based on the top plane of the pocket.

- **Plane**

When Plane is specified, the Select button can be used to interactively pick the plane.

Retract to Clearance Level at End:

Specifies will the tool retract to the clearance plane at end of pocket motion. Toggle between “Yes” and “No”.

- **Yes**

Retract the tool to the clearance plane.

- **No**

Do not retract the tool to the clearance plane.

Retract Between Levels:

Specifies the tool either be retracted to the “Clearance Plane” or an “Incremental” distance above the pocket top plane between successive pocketing levels. If the Clearance value is specified as a plane, then Incremental cannot be specified.

Retract between Sections:

Specifies whether the tool should be retracted between each pocket section.

- **Yes**

The tool to retract between each pocket section.

- **No**

The tool to remain down at the pocket level when transitioning between pocket sections.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Rapto Distance:

The tool will move down the tool axis at the retract feed rate until it is this distance above the top of the pocket or the top plane of the last machined level. This value is also referred to as the Retract Distance.

This value is also used to determine the height and the radius of the helical arc move or the side of the 45-degree linear move that is made away from the wall and off the floor prior to retracting the tool to the clearance plane.

Retract Distance (horiz):

Specifies the 45 degree horizontal linear move at the cutting level before retracting.

Retract Distance (ver):

Specifies the vertical retract distance from the cutting level before retracting.

Arc Exit off Perimeter:

Check this to generate a 90 degree arc exit off of the pocket perimeter. Uncheck this to generate a straight line exit of 45 degrees to the last perimeter move.

The height and the radius of the helical arc move or the length of the side of the 45-degree linear move is defined by the “[Retract Distance](#)” parameter.

Arc Entry/Exit off Islands:

Check this to generate a 90 degree arc exit off of the pocket islands. Uncheck this to generate a straight line exit of 45 degrees to the last island move.

Cutting Parameters:

This section specifies how the pocket motion should be generated.

Method:

Specifies the method to be used for generating the pocket motion. Toggle between “Collapse”, “Lace”, “Lace + Both”, “Scrub” or “Scrub + Both”.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Collapse**

Generate a standard collapsing pocket motion.

- **Lace**

Generate a lace style pocketing motion (back and forth), which will retract the tool each time a pocket boundary or island is reached.

- **Lace + Both**

This is similar to Lace, except the tool cuts along the boundary when intermediate boundary intersections are found, and retracts to avoid recutting an already cut piece of the boundary. When retracted, the tool is repositioned above the beginning of the next uncut portion of the boundary.

- **Scrub**

Generate a scrub style pocketing motion, which is similar to Lace, except that the tool will remain at the current level and navigate around the pocket boundary and islands when traversing to the next pass or avoiding pocket geometry.

- **Scrub + Both**

This is similar to Scrub, except the tool is retracted to avoid recutting an already cut piece of the boundary. When retracted, the tool is repositioned above the beginning of the next uncut portion of the boundary and then reenters the cutting level.

Boundary Direction:

Specifies the pocketing motion boundary direction when “Lace+Both” or “Scrub+Both” is selected as the cutting method. Toggle between “Short”, “Same”, “CCLW” or “CLW”.

- **Shortest**

Specifies the cut takes the shortest way around the boundary. This is the default.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Same**

Specifies the shortest path is calculated only once for each boundary - each subsequent cut around the boundary is made toward the previously cut part of the boundary (so that the cutter stays down as little as possible).

- **CCLW**

Specifies the cut goes counter-clockwise (as viewed down the tool axis) around the perimeter and clockwise around islands.

- **CLW**

Specifies the cut goes clockwise (as viewed down the tool axis) around the perimeter and counter-clockwise around islands.

Pocket Direction / Cutting Direction:

Specifies the pocketing motion direction. Toggle between “CCLW” and “CLW” for Collapse. Toggle between “Positive X”, “Negative X”, “Positive Y”, “Negative Y”, “Long”, “Short” or “Vector” for Lace, Lace + Both, Scrub or Scrub + Both.

- **CCLW**

Specifies the pocketing motion as Counter-Clockwise direction if the Pocket Method is Collapse.

- **CLW**

Specifies the pocketing motion as Clockwise direction if the Pocket Method is Collapse.

- **Positive X**

Specifies the general pocketing direction is in the Positive X direction if the Pocket Method is Lace or Scrub.

- **Negative X**

Specifies the general pocketing direction is in the Negative X direction if the Pocket Method is Lace or Scrub.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Positive Y**

Specifies the general pocketing direction is in the Positive Y direction if the Pocket Method is Lace or Scrub.

- **Negative Y**

Specifies the general pocketing direction is in the Negative Y direction if the Pocket Method is Lace or Scrub.

- **Long**

Specifies the general pocketing direction is along the longest side of a rectangle encompassing the pocket geometry.

- **Short**

Specifies the general pocketing direction is along the shortest side of rectangle encompassing the pocket geometry.

- **Vector**

Specifies the general pocketing direction is along a specified vector. The Select button can be used to interactively pick the vector.

Spiral Direction:

Specifies the spiral direction of collapsing pocket motion and is only active when the Method field is set to Collapse. Toggle between “Out” and “In”.

- **Out**

The tool will start in the center of each pocket section and work its way out towards the pocket perimeter.

- **In**

The tool will start on the outside of the pocket and works its way in.

Maximum Stepover Distance:

Specifies the maximum stepover amount from one concentric pass to the next. If set to 0, the current effective cutter diameter will be used (dia-cr).

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Minimum Stepover Distance:

Specifies the minimum amount from one concentric pass to the next. If set to 0, a value equal to .1 of the Maximum Stepover Distance will be used.

Scallop Height:

Specifies the parameter used for coverage testing: the amount of material allowed to be left under the cutter's corner radius. This is only active when the cut method is specified as Collapse.

Transition between Loops:

Specifies the transition mode between loops. Toggle between "Sharp" and "Arc". This is only active when the cut method is specified as Collapse.

- **Sharp**

Produces a straight line move between pocket loops.

- **Arc**

Produces a double-arc or S-shaped transition between pocket loops.

Final Pass Direction

Specifies the final pass direction for "Lace", "Lace + Both", "Scrub" or "Scrub + Both" pocketing. Toggle between "Same", "Reverse", "CCLW", "CLW" or "Omit".

- **Same**

The final pass will continue in the direction of the last lace/scrub move. This is the default.

- **Reverse**

The final pass will go in the direction opposite to the last lace/scrub move.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **CCLW**

The final pass goes counter-clockwise (as viewed down the tool axis) around the perimeter.

- **CLW**

The final pass goes clockwise around the perimeter.

- **OMIT**

Specifies no final pass is performed.

If the final pass does occur (when the choice is not Omit), passes around islands are always done in the direction opposite that of the perimeter final pass. If there is no final pass around the perimeter (the choice is Omit), there will be no final passes around islands.

Options:

This section composed of three items: Force Output of Modals, Final Pass and Feedrates. They are described as follows:

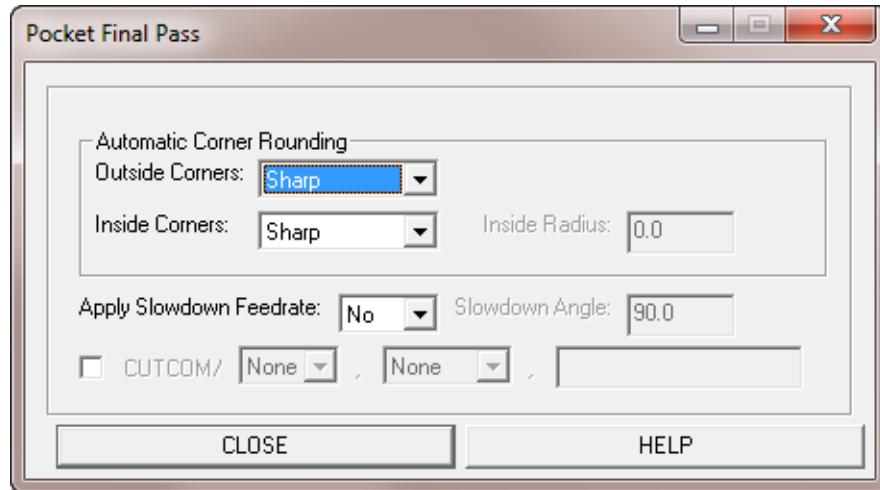
Force Output of Modals:

If this box is checked, the POKMOD statement is output unconditionally. If not checked, the POKMOD statement is output only if: 1) Pocket Modals have been changed and 2) a valid POCKET statement is also output by the main form.

Final Pass:

Specifies the final pass parameters of the pocketing motion. Click this button to display the “Final Pass” form as shown on next page

6 AUTOMATIC MOTION ROUTINE STATEMENTS



- **Outside Corners:**

Specifies to cut the sharp corners on the periphery as a circular move not. Toggle between “Sharp” and “Arc”.

- **Sharp**

Indicates sharp/outside corners will not be rounded.

- **Arc**

Indicates motion at sharp corners on the periphery will be generated as a circular move around the corner. The angle of the corner must be less than 130 degrees.

Inside Corners:

Specifies to generate filleting motion around inside corners or not. Toggle between “Sharp” and “Arc”.

- **Sharp**

Indicates no filleting motion will be generated for inside corners.

- **Arc**

Indicates to generate filleting motion around inside corners. This is unlike the Outside Corners, the filleting is done after the motion is calculated, so the radius is the same for all the loops.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Inside Radius:

Specifies the fillet radius to be generated. This is only active when Inside Corners is set to Arc.

Apply Slowdown Feedrate:

Specifies will secondary feedrate be applied to the last pass around the pocket perimeter and islands. Toggle between “Yes” or “No”.

- Yes

Applies secondary feedrate to the last pass around the pocket perimeter and islands.

- No

Do not apply secondary feedrate to the last pass around the pocket perimeter and islands.

Slowdown Angle:

Specifies the minimum angular change of direction that the secondary feedrate will be applied. Active only when Yes is chosen for “Apply Slowdown Feedrate”.

CUTCOM/:

Check this to output a CUTCOM postprocessor statement to the pass around the pocket perimeter and islands. A “CUTCOM/OFF” postprocessor statement will immediately output to the CLfile to reset the CUTCOM condition after the geometry profiling around the periphery and each island. Uncheck this not to output a CUTCOM postprocessor statement.

Check this will enable the Mode, Cutcom Plane and Text String Entry Box.

- Mode

Specifies the CUTCOM mode. Toggle between “Left”, “Right”, “On or “None”

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Cutcom Plane**

Specifies the CUTCOM plane. Toggle between “XYPLAN”, “YZPLAN”, “ZXPLAN” or “None”.

- **Text String Entry Box**

Any other required CUTCOM postprocessor command parameters can be specified in this box. Each parameters must be separated by a comma (,).

CLOSE:

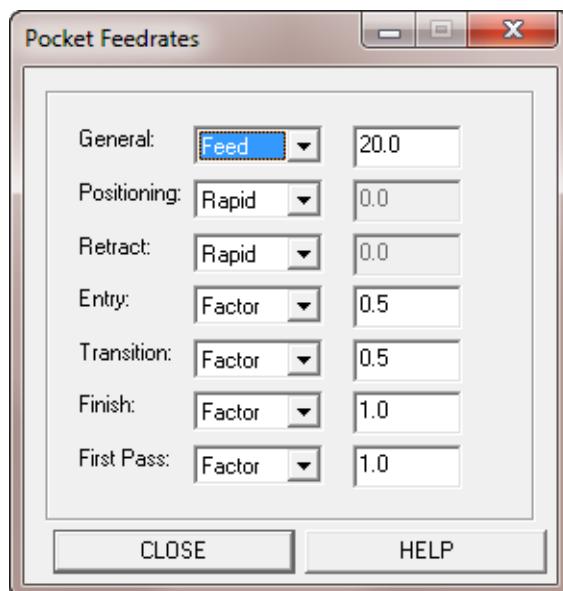
Click this button to accept the change and close the Pocket Final Pass form.

HELP:

Click this button to bring up the online help for the Pocket Final Pass form.

Feedrates:

Specifies the feedrate parameters for the pocketing motion. Click this button to display the “Feedrates” form as shown below.



6 AUTOMATIC MOTION ROUTINE STATEMENTS

General:

Specifies the general pocketing feedrate. Toggle between “Current” and “Feed”.

- **Current**

Specifies to use the current primary feedrate as the general pocketing feedrate.

- **Feed**

Specifies the general pocketing feedrate value. This is the default.

Positioning:

Specifies the feedrate at which the tool will travel while moving at the clearance level. Toggle between “Rapid”, “Feed” or “Factor”.

- **Rapid**

Specifies the tool will travel in RAPID mode. This is the default.

- **Feed**

Specifies the positioning feedrate value.

- **Factor**

Specifies the positioning feedrate as a factor of the general feedrate.

Retract:

Specifies the feedrate at which the tool will travel while retracting to the clearance level and moving to the retract depth. Toggle between “Rapid”, “Feed” or “Factor”.

- **Rapid**

Specifies the tool will travel in RAPID mode. This is the default.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Feed**

Specifies the retract feedrate value.

- **Factor**

Specifies the retract feedrate as a factor of the general feedrate.

Entry:

Specifies the feedrate at which the tool will travel while entering each pocket section. Toggle between “General”, “Feed” or “Factor”.

- **General**

Specifies the general feedrate to be used as the entry feedrate.

- **Feed**

Specifies the entry feedrate value.

- **Factor**

Specifies the entry feedrate as a factor of the general feedrate.
This is the default.

Transition:

Specifies the feedrate at which the tool will travel while moving to the next “concentric ring”. Toggle between “General”, “Feed” or “Factor”.

- **General**

Specifies the general feedrate to be used as the transition feedrate.

- **Feed**

Specifies the transition feedrate value.

- **Factor**

Specifies the transition feedrate as a factor of the general feedrate.
This is the default.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Finish:

Specifies the feedrate at which the tool will travel while making the final pass around the periphery and islands. Toggle between “General”, “Feed” or “Factor”.

- **General**

Specifies the general feedrate to be used as the finish feedrate.

- **Feed**

Specifies the finish feedrate value.

- **Factor**

Specifies the finish feedrate as a factor of the general feedrate.
This is the default.

First Pass

Specifies the feedrate at which the tool will travel while making the first pass around the periphery and islands. Toggle between “General”, “Feed” or “Factor”.

- **General**

Specifies the general feedrate to be used as the first pass feedrate.

- **Feed**

Specifies the first pass feedrate value.

- **Factor**

Specifies the first pass feedrate as a factor of the general feedrate.
This is the default.

CLOSE:

Click this button to accept the change and close the Pocket Feedrates form.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

HELP:

Click this button to bring up the online help for the Pocket Feedrates form.

CLOSE:

Click this button to accept the change and close the Advanced Pocket Modals form.

HELP:

Click this button to bring up the online help for the Advanced Pocket Modals form.

6.28.2.1.1 Helix Output as CYCLE in POKMOD Command

Following is a description of CYCLE in POKMOD command.

```
CYCLE/CIRCUL,DEPTH,z,RADIUS,d,ON,      $  
      STEP,zi,IPM,f,RAPTO,r,      $  
      ATANGL,a,DOWN,CCLW  
      GOTO/xc,yc,zc  
      GOTO/x2,y2,z2
```

where:

- | | |
|-----------------|---|
| z | - Total incremental depth of the entry |
| d | - Helix radius |
| zi | - Depth per one helical rotation |
| f | - Entry feedrate |
| r | - POKMOD retract-dist parameter |
| a | - Starting angle of the entry (relative to the positive X-axis) |
| xc,yc,zc | - Center of the circle defined as the projection of the entry helix onto the plane at retract-dist above the pocket top plane |
| x2,y2,z2 | - Last point of the helical entry, also the first point of the pocketing motion |

6 AUTOMATIC MOTION ROUTINE STATEMENTS

The entry then is interpreted as the following motion:

- i. Go to the point (xc,yc,zc) – the center of the circle
- ii. Go to the normal helical entry starting point, but at the retract-dist lower (that is, at the pocket top plane)
- iii. Perform the helical entry
- iv. Perform one horizontal circular motion at the bottom

6.28.2.1.2 Helix Output as Text String in POKMOD Command

Instead of the regular point to point motion output, the following motion and postprocessor commands will be output to the CL file.

```
text-variable
GOTO/xc,yc,zc
[text-variable-two]
GOTO/x2,y2,z2
```

where:

xc,yc,zc - Center of the circle defined as the projection of the entry helix onto the plane at retract-dist above the pocket top plane

x2,y2,z2 - Last point of the helical entry, also the first point of the pocketing motion

Here a text variable is used to format a post-processor command in the CL file. An acceptable text variable is of the form:

```
abc="CYCLE/CIRCUL,%Z,%D,%I,%F," & $  
" [IPM/IPR],%R,[UP/DOWN]," & $  
"CCLW/CLW[ ; CYCLE/OFF]"
```

An optional second text variable can be entered after a semicolon, then it is entered as the third command after “GOTO/xc,yc,zc”. In this example, the “CYCLE/OFF” command cancels the post-processor helical interpolation mode.

where:

%Z – Total incremental depth of the entry

6 AUTOMATIC MOTION ROUTINE STATEMENTS

%D	-	Helix diameter + the cutter diameter
%I	-	Depth per one helical rotation
%F	-	Entry feedrate, per minute or per rotation
%R	-	POKMOD retract-dist parameter
xc,yc,zc	-	Center of the circle defined as the projection of the entry helix onto the plane at retract-dist above the pocket top plane
x2,y2,z2	-	Last point of the helical entry, also the first point of the pocketing motion

The entry then is interpreted as the following motion:

- i. Go to the point (xc,yc,zc) – the center of the circle at retract-dist above the pocket top plane
- ii. Go to the point on the circle at 45 degrees from the center and at the pocket top plane
- iii. Perform the helical entry, CCLW by default
- iv. Perform one horizontal circular motion at the bottom
- v. If UP is specified, retract by the retract-dist up at the RAPID rate

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.28.2.2 POCKET

The Advanced POCKET statement syntax specifies the geometric elements that define the POCKET boundary and islands. The current cutter and POKMOD parameters are used to determine offset distances and coverage testing. The valid syntax for the Advanced POCKET statement is shown below.

```
POCKET/bottom-plane-or-surface, $  
    [SAME , ] top-plane-or-dist $  
    NORMAL[ , PS, surf]  
    [,direction-mod],perimeter-geo $  
    [,direction-mod],island-geo,... $  
    [,START,start-point][,max-loops] $  
        END,end-point  
            scalar  
            [, PS, THICK, pthk] [, DS, THICK, dth1 [, dth2] ] $  
            [, FINISH, THICK, thk] [, OFF, PART[, dis]] $  
            [, OPEN, ind1[, THRU, ind2]]  
                ALL
```

Where:

bottom-plane or surface:

- A PLANE or SURF that defines the bottom surface of the POCKET. A canted plane may be used as the bottom surface as long as the normal is not perpendicular to the tool axis. Gouge check may be activated for flat tools over wavy surface bottoms. The lowest tool point on the pocket bottom is used with the top plane height to calculate the number of cut layers and their thicknesses.

SAME:

- Specifies a fixed tool axis mode. This is the default.

NORMAL:

- Specifies tool axis mode normal to the bottom of the pocket.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- At a pocket entry the tool axis vector is the same as at the first surface point.
- At an exit move the tool exit is the same as the last surface point.

PS, surf:

- Specifies a secondary part surface to be used as a control surface for the tool axis. The tool axis is normal to this control surface instead of the pocket bottom.
- "surf" is the label of the control surface. It can be a surface type entity or a plane type entity.

top-plane-or-dist:

- A PLANE, a SURFACE of primitive type "PLANAR" or distance from the bottom-plane/surface that defines the top of the POCKET. This should be the level at which material will be cut. This is NOT the clearance plane where positioning moves are made. This top plane or planar surface must be normal to the tool axis if NORMAL is not specified.
- Distance is not recommended if the bottom entity is not a planar type entity, or a planar type entity whose normal is not parallel to the tool axis. **NCL** will determine the lowest point on the bottom entity within the area to be pocketed. An internal top plane would be created at a distance specified by "dist" along the tool axis above this lowest point. This internally created top plane could be below a portion of the bottom entity if the specified distance is not large enough. This will cause **NCL** to generate undesirable pocket motion which could undercut the pocket bottom during entry.

direction-mod:

- The vocabulary word IN, OUT or ON that specifies the cutter's relationship to the perimeter geometry at the finish pass.
- "IN" tells **NCL** to keep the cutter on the inside of the perimeter geometry.
- "OUT" tells **NCL** to produce motion that leaves the cutter moving OUTSIDE the perimeter geometry on the last pass.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- "ON" indicates the final pass should be made with the center of the cutter ON the perimeter geometry. These are similar in concept to the modifiers TO, ON and PAST used in GO . . . type statements. This is optional and if omitted, IN will be used.
- "OFFSET[,thk]" is only valid when the same surface is specified as the bottom surface, perimeter geometry, and optionally the island geometry. When OFFSET is specified, an analysis of the surfaces adjacent to the perimeter surface will be made and the pocket boundary will be recalculated using the following rules.
 - Boundaries of adjacent fillet surfaces that have a radius equal or greater than the corner radius of the cutter will be expanded by the corner radius of the cutter.
 - Boundaries of adjacent surfaces that are at the same level or below the selected surface will be expanded by half the cutter diameter plus the optional thick value (thk) specified with the OFFSET parameter.
 - All other boundaries will remain the same.

perimeter-geo:

- The geometry item that defines the perimeter of the pocket. This may be one of the following:
 - A Composite CURVE
 - A Closed Curve or Spline
 - A PATERN
 - A 360 degree CIRCLE
 - The base name of a subscripted array of POINTS
 - A SURFACE
- If a SURFACE is specified as the perimeter-geo, the outside boundary of this surface will be used. The underlying surface boundary will not be used if the surface specified is of the type "**TRIMMED**", the actual trimmed surface outside boundary will be used instead. This can has the same name as the bottom-surface.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

direction-mod

- The same as the direction-mod for the perimeter geometry except
 - This applies to the cutter's relation to the following island geometry.
 - Inner boundaries whose adjacent surfaces are below the selected surface will be ignored if OFFSET is specified. The outer boundaries *thk* value will be used.
- This is optional and if omitted, OUT will be used.

island-geo:

- The geometry (items) that define any optional islands of a pocket.
- This may be any of the following:
 - A Composite CURVE
 - A Closed Curve or Spline
 - A PATERN
 - A 360 degree CIRCLE
 - The base name of a subscripted array of POINTS
 - A Trimmed Surface
- Islands are areas such as bosses or holes that are to be avoided by the cutter. Each island can optionally be preceded by a direction-mod as defined above. If an island is not preceded by a direction-mod, it will use the value assigned for the preceding island.
- If a "Trimmed Surface" is specified as the island-geo, the inner boundary(ies) for this surface will be used as the islands geometry and no other island geometry will be allowed. This can have the same name as the perimeter-geo or the bottom-surface.
- As many island-geo entities as desired may be specified along with optional preceding direction-mods as long as a "Trimmed Surface" is not specified.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

START,start-point:

- This specifies the optional start-point of LACE or SCRUB style pocket motion. This will not apply to other type of pocket motion. This option cannot be specified with the “END, end-pt-or-scalar” and “max-loops” options.

END, end-point-or scalar:

- The vocabulary word "END" followed by a POINT or an integer scalar which specifies where the POCKET motion will end. If a POINT is given, the motion will end near that POINT. An integer scalar "n" indicates the motion will end near the end point of the "n'th" element of the perimeter-geo.

max-loops:

- Defines the maximum number of loops of motion around the perimeter (excluding the island geometry, however they will be respected for motion generation) of the pocket.
- If not specified, **NCL** will make as many loops as necessary to clean out the entire pocket area.

PS, THICK, pth:

- Specifies an optional part surface thick parameter to the POCKET command. The current part surface thick will be utilized if these parameters are not specified.

DS, THICK, dth1 [, dth2]:

- *dth1* specifies an optional drive surface thick parameter to the POCKET command. The current drive surface thick will be utilized if these parameters are not specified.
- *dth2* specifies the additional open boundary side(s) thick value.

FINISH, THICK, thk:

- Specifies an optional final pass extra thick parameter to the POCKET command.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- If specified, the extra thick is only applied to the pocket bottom for those passes next to the perimeters or islands in the final level. It does not apply to other area of the pocket or other levels. Otherwise specifies THICK/ps,ds before the POCKET command, then “ps” and “ds” will be applied to all areas of the pocket and for all levels.

OFF, PART[, dis]:

- Specifies to enter off the part for open stock pockets if possible.
- The current tool position (just before the POCKET command) is used as the desired start location. The tool is repositioned there at the cutting level before each entry move, and then moved (at the entry feedrate) horizontally to the first cutting contour point.
- If WARN or AVOID gouge checking is specified, the Off Part entry is performed only when the pocket islands are not gouged, otherwise the current POKMOD entry mode is in effect.
- *dis* is an optional parameter specifies the offset distance used when positioning the cutter outside of an open boundary. The cutter will be positioned *dis* away from the perimeter geometry. A value of zero (0) is the default which specifies that the cutter radius should be used.

OPEN,ind1[,THRU,ind2]]:

ALL

- Specifies open sides of the pocket boundary.
- Open sides can be defined for pockets whose perimeter is defined using a composite curve or subscripted point array.
- = There can be multiple open sides for a single perimeter entity. To specify a range of curve components or points the perimeter *THRU* can be used.

The following command will use a composite curve to define the pocket boundary with the boundary being offset along multiple components of CV1

POCKET/(PL/0,0,1,0),0.5,CV1,OPEN,1,3,5,THRU,8

6 AUTOMATIC MOTION ROUTINE STATEMENTS

The following command will use an array of points to define the pocket boundary with two open sides.

```
POCKET/(PL/0,0,1,0),0.5,CV1,OPEN,1,THRU,3,5,THRU,8
```

Note that with an array of points, the parameter *THRU* must be specified in order to define an open side whereas a composite curve can use a single component.

- Entering through an open side will not violate islands or closed boundary sides.
- Ramp entry through an open side will use a single ramp to move down to the pocket level.
- Flat helix entry will end tangent to the first motion segment and in the same direction to eliminate any changes in direction when pocketing starts.
- HELIX entry will not be used when entering through an open side for LACE/SCRUB style motion. Any entry through an open side will use PLUNGE. In addition, any entry through an open side that would be longer than twice the cutter diameter will not be used when LACE/SCRUB style motion is used.
- If no entry can be found that does not violate island boundaries or the closed portion of the pocket boundary, then the original entry method will be used to enter inside the pocket boundary.

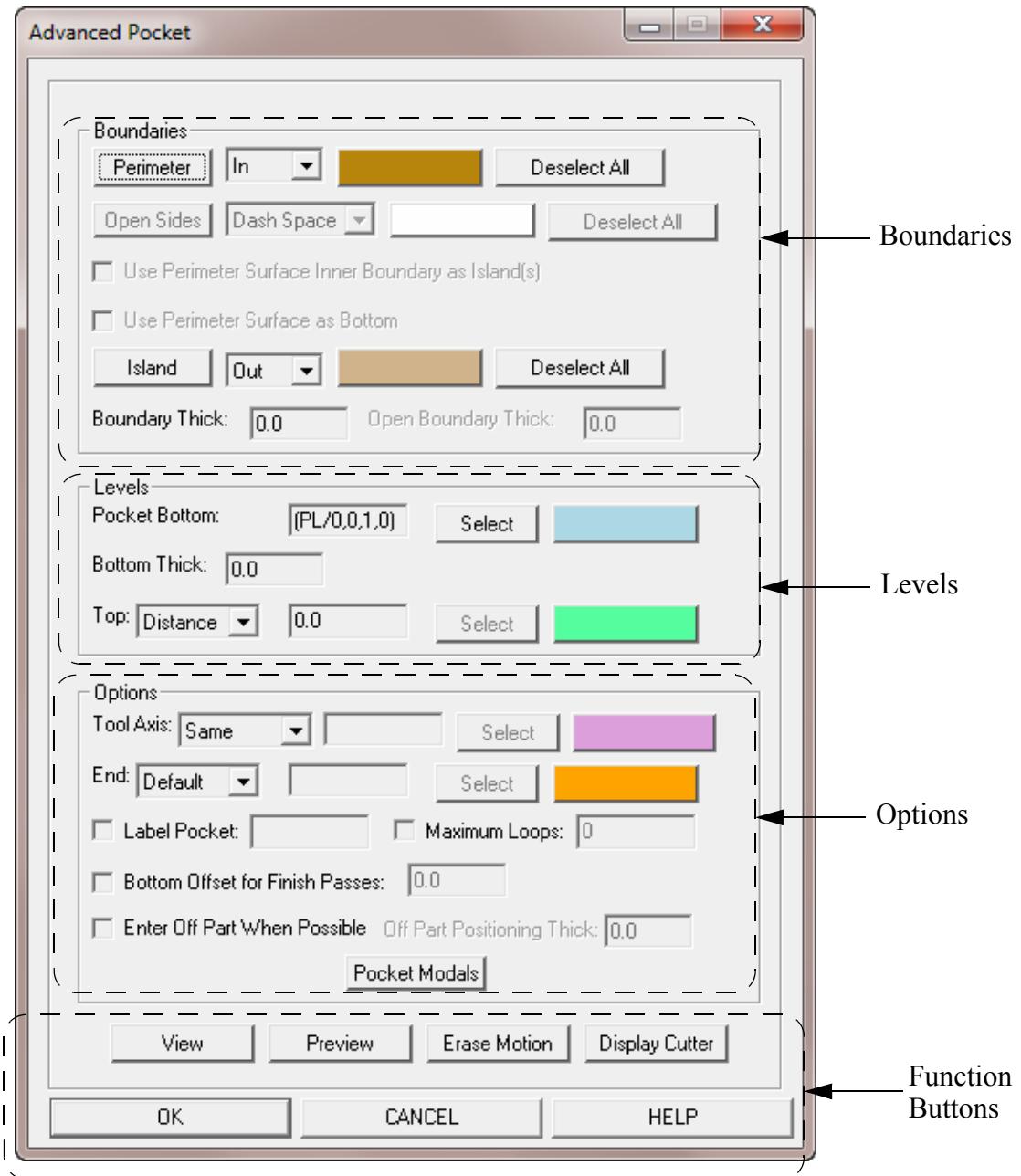
Note: If the optional parameters “**NORMAL [, PS, surf]**” are specified and the boundary are not exactly on the bottom surface, the actual boundary of the bottom surface for the pocket motion will be the projection of the boundary to the bottom surface at a direction normal to the average plane of the specified boundary in 3-D space. This might not be what the user expected. The suggested way is to project the boundary normal to the bottom surface first and use the projected entity as the pocket boundary.

Following pages show the graphical interactive interface for this Advanced **POCKET** routine and a description of how to use this interface. This interface can be activated by using the following on screen menu icon sequences:

6 AUTOMATIC MOTION ROUTINE STATEMENTS



> *Pocket*



This form composed of the following sections: Boundaries, Levels, Options and Function Buttons.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Boundaries:

This section specifies the outside and inside boundaries of pocket(s).

Perimeter:

Takes down the form, brings up the SELECT menu. Allows selecting a list of entities - each perimeter entity accounts for a separate POCKET command. Note that having selected more than one perimeter prevents you from selecting island geometry.

Any curve can be chosen as a perimeter. It should be closed however, otherwise the pocket algorithm will replace the end point to close it.

A surface can be selected as a perimeter. The algorithm will internally use a composite curve made of the surface outer boundary.

Direction-modifier:

Specifies the cutter's relationship to the perimeter geometry at the finish pass. The modifier will be used for all the selected perimeter geometry. Toggle between "In", "On", "Out" or "Offset".

- **In**

Specifies to keep the cutter on the inside of the perimeter geometry. This is the default.

- **On**

Specifies the final pass should be made with the center of the cutter ON the perimeter geometry.

- **Out**

Specifies to produce motion that leaves the cutter moving OUTSIDE the perimeter geometry on the last pass.

- **Offset**

Specifies to do an analysis of the surfaces adjacent to the perimeter surfaces. This option will only be enabled if the selected entity is a surface.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Color:

Chooses a color to highlight the selected perimeter geometry.

Deselect All:

Deselects the currently selected perimeter geometry entities.

Use Perimeter Surface Inner Boundary as Island(s):

Specifies to use the inner boundaries as pocket islands if the perimeter surface is trimmed and has inner boundaries.

Use Perimeter Surface as Bottom:

Specifies to use the perimeter surface as the pocket bottom. If chosen so, the Pocket Bottom fields become disabled.

Island:

Takes down the form, brings up the SELECT menu. Allows selecting a list of entities. Any closed curve can be specified as island geometry. Also, a trimmed surface with inner boundary could be specified, in which case the algorithm will use the inner boundary curves.

Direction-modifier:

Specifies the cutter's relationship to the island geometry at the finish pass. The modifier will be used for all the selected island geometry. Toggle between "In", "On" or "Out".

- In**

Specifies to keep the cutter on the inside of the island geometry.

- On**

Specifies the final pass should be made with the center of the cutter ON the island geometry.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Out**

Specifies to produce motion that leaves the cutter moving OUTSIDE the island geometry on the last pass. This is the default.

- **Offset**

Specifies to do an analysis of the surfaces adjacent to the inner boundary surfaces. This option will only be enabled if the selected perimeter is a surface.

Color

Chooses a color to highlight the selected island geometry.

Deselect All

Deselect the currently selected island geometry entities.

Boundary Thick:

Specify an optional drive surface thick for the pocket motion. This is corresponding to the “DS,THICK,dth1” parameters in the POCKET command.

Open Boundary Thick”

Specify the open boundary side thick. This is corresponding to the dth2 values of the “DS,THICK,dth1,dth2” parameters in the POCKET command.

Levels:**Pocket Bottom:**

Contains the current pocket bottom, unless a pocket perimeter surface is used as pocket bottom (see above).

Select:

Allows picking a plane or surface as pocket bottom.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Color:

Chooses a color to highlight the selected pocket bottom.

Bottom Thick:

Specify an optional part surface thick for the pocket motion. This is corresponding to the “PS,THICK,dth” parameters in the POCKET command.

Pocket Top:

Choice between using a plane or a distance for pocket top. The text field contains the current pocket top, plane or distance.

Select:

Allows picking a plane or planar surface as pocket top - active only when Plane is chosen.

Color:

Chooses a color to highlight the selected pocket top.

Options:

Tool Axis:

Specifies tool axis mode. Toggle between “Same”, Normal”, or “Normal PS”.

- **Same**

Specifies the tool axis is a fixed axis mode and same as before the pocket motion. This is the default.

- **Normal**

Specifies the tool axis normal to the pocket.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Normal,PS**

Specifies the tool axis normal to a secondary Part Surface, i.e. the tool axis control surface.

Text Field:

This contains the tool axis controlling surface. Active only when Normal PS is specified.

Select:

Allows picking a tool axis controlling surface. Active only when Normal PS is specified.

Color:

Chooses a color to highlight the selected secondary Part Surface. Active only when Normal PS is specified.

End:

Choice for the pocket end: not specified (default), use a boundary element number, use a point. The text field contains the current pocket end, a number or a point.

Select:

Allows picking a point as a pocket end - active only when Point is chosen.

Color:

Chooses a color to highlight the selected pocket end.

Label Pocket:

Chooses whether the pocket routine is to be named and saved for later processing, i.e., whether the form outputs a statement like:

POK1=pocket/sf1,pl1,cv1,...

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Text Field:

This contains the name of the pocket routine. Active only if Label Pocket is specified.

Maximum Loops:

Chooses whether to use the optional maximum number of loops. The text field contains the maximum number of loops.

Bottom Offset for Finish Passes:

Enabling this check box allows you to specify an extra thick parameter that will be applied to the cuts at the pocket bottom level and next to the pocket geometry (perimeter and islands).

Enter Off Part When Possible:

When this check box is enabled, **NCL** will enter the pocket from outside the pocket perimeter when it is able to. The current tool position is used as the desired start location. The tool is repositioned to this location at the current cutting level before each entry move and then moved at the entry feed rate horizontally to the first cutting contour point.

Off Part Positioning Thick:

Used for positioning the tool when entering from off the part through an open side. The thick value given will be used in when setting the distance the cutter will start from the boundary of the pocket.

Pocket Modals:

Opens the Pocket Modals form.

Function Buttons:

View:

Takes down the form and enters dynamic viewing.

Preview:

Previews the Pocketing motion. No command is output.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Erase Motion:

Erases all motion displayed in the graphic area.

Display Cutter:

Displays the current cutter position.

OK:

Click this button to accept the setting, close the form, generate the motion and output all the corresponding source codes to the part program.

Cancel:

Click this button to disregard all the changes (if any) and close the form.

Help:

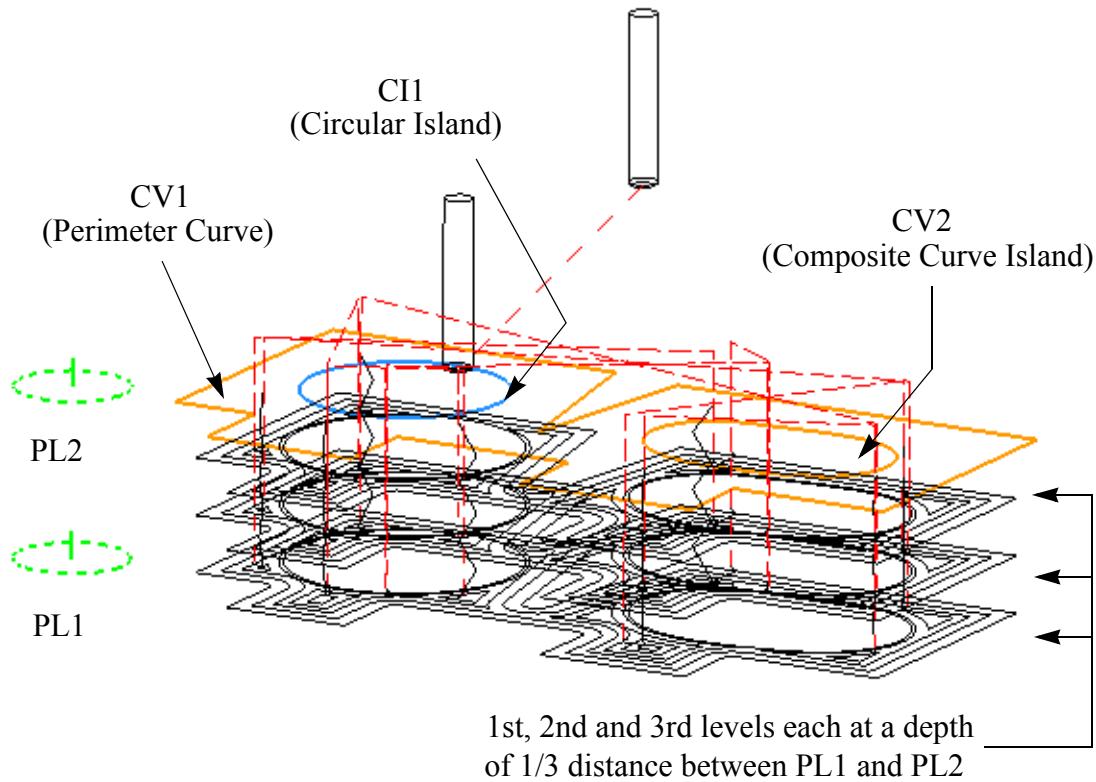
Click this button to open the online help for this Advanced Pocket Form.

Example statement sequence:

```
DIA=.4  
COR=.125  
CUTTER/DIA,COR,1.5  
FD=20  
POKMOD/2,RAMP,WARN,0,1,-3,.25,CCLW,OUT,DOWN, $  
      ARC,0,.1,FD,RAPID,RAPID,-.5,-.5,-1  
POCKET/PL1,PL2,CV1,CI1,CV2
```

The preceding group of statements would machine a pocket with two islands. The bottom of the pocket is at PL1. The top of the pocket is at PL2. The pocket would be cleared at three levels as shown in the following figure.

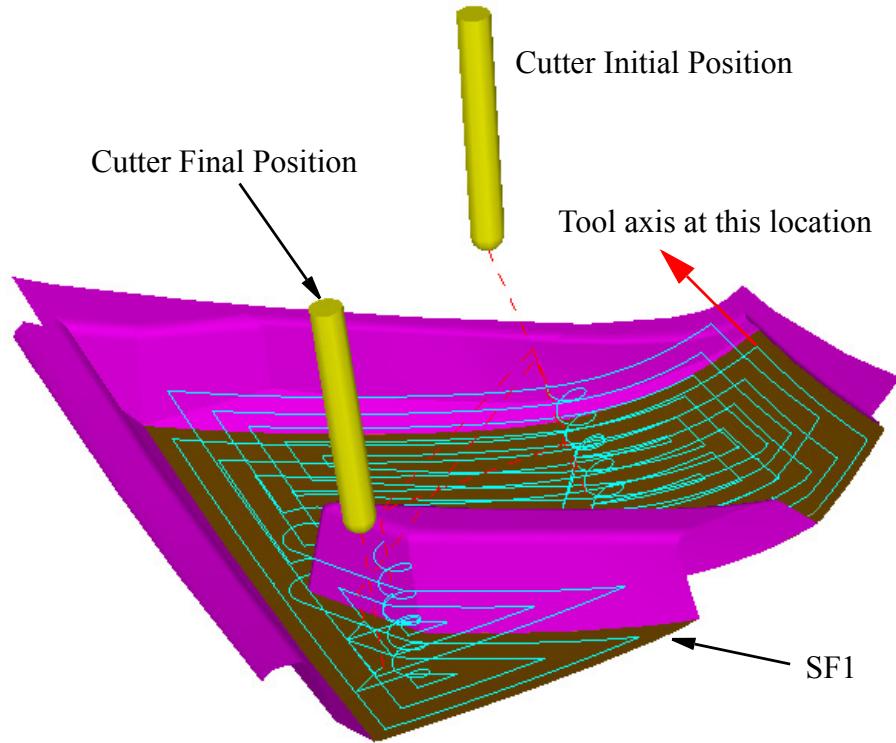
6 AUTOMATIC MOTION ROUTINE STATEMENTS



Example statement sequence with taxis normal to bottom of pocket:

```
DIA=.1  
COR=.05  
CUTTER/DIA,COR,1  
FD=20  
POKMOD/3,HELIX,WARN,0.05,RETRCT,ON,0.2,      $  
    INCR,-2,0.1,CCLW,OUT,UP,SHARP,                $  
    MXSTP,MINSTP,FD,RAPID,RAPID,  
    -.5,-.5,-1  
POCKET/SF1,NORMAL,CLDIS,SF1
```

The preceding group of statements would machine a pocket with taxis normal to bottom of pocket. The bottom of the pocket is at SF1. The top of the pocket is at a distance of CLDIS. The pocket would be cleared at two levels as shown in the following figure.



THICK may be used as with other types of motion where "THICK/ ps" applies to the POCKET floor and "THICK/ ds" applies to the pocket perimeter and islands. "THICK/ cs" has no effect.

6.28.2.3 Save Pocket Routine

A pocket routine may be named and saved for later processing. The following statement:

```
POK1=POCKET/SF1,PL1,CV1,...
```

would create a pocket named POK1 but would not output any points to the CL file.

The pocket entry point-vectors could then be obtained by entering:

```
OBTAINT/POK1,N,P,[,UP ]  
DOWN
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Where N is a scalar variable which will contain the number of entry points and P is a reserved point-vector variable whose elements will contain the pocket entry points and the corresponding tool axis values. If UP is specified, the points-vectors will be located on the top plane of the pocket. If DOWN is specified, the point-vectors will be located on the bottom plane of the pocket. The default value is UP.

The saved pocket may then be processed by entering:

POCKET /POK1

All values such as **CUTTER**, **TOLER**, etc. in effect at the time the pocket was saved will be used except if a scale general feed rate was specified in the **POKMOD** statement (by entering a negative value), the feed rate used for the pocket will be a factor of the feed rate in effect at the time the pocket is executed.

6.29

PROFIL

The PROFIL statement will initialize an automatic routine to generate motion

1. for driving the cutter on a 2D/3D curve with a plane or surface as the part surface, or
2. annotation engraving, i.e. letter engraving.

6.29.1

PROFIL - 2D/3D Curve Cutting

This version of PROFIL statement allows the user to drive the cutter on a 2D/3D curve with a plane or surface as the part surface. The tool axis can either be fixed or normal to a surface. The tool will drive the curve from the starting location to its end or a specified location for a open curve. If the curve is closed, then the tool will drive all the way around the curve or from a specified starting location to a specified location.

The valid syntax construct for this PROFIL statement is shown on next page:

6 AUTOMATIC MOTION ROUTINE STATEMENTS

```
PROFIL/[ON ,]curve [,th1][,AUTO ][,PAST,both      ] $  
    OFF   circle        LEFT       start,end  
          line          RIGHT  
  
[ ,START[,END      ] [,CCLW] $  
    ENDPT      CLW  
    NEARPT,pt1  vel  
          pv1     pv2  
  
[ , OMIT [ ] ] $  
    [ATANGL,ds1,ang1[,h1]] [[,]ARC,r1[,h2]]  
  
[ ,CS,ENDPT      ] [,PS[,pl1][,th2][,SAME      ] ] $  
    NEARPT,pt2          NORMAL[,pl2]  
          pv3           sf2  
  
[ ,CLRSRF,pl3      ] [,RAPTO,pl4      ] $  
    [DIST,]ds2          [DIST,]ds3  
          INCR           INCR  
  
[ ,RETRCT,pl5      ] $  
    ON  
    OFF  
    [DIST,]ds5  
          INCR  
  
[ , OMIT [ ] ] $  
    [ATANGL,ds4,ang2[,h3]] [[,]ARC,r2[,h4]]  
  
[ ,RTRCTO,ds6] [,OFFSET,ds7,stp1      [,DOWN] ] $  
          PASS,n1      UP  
          PASS,n1,stp1  OFF  
  
[ ,STEP,top-plane,stp2      ] [,LEVEL] ] $  
    ds8      PASS,n2      DEEP  
    PASS,n2,stp2  
  
[ ,FEDRAT,gen-fr, pos-fr,ret-fr,ent-fr, $  
    tran-fr,exit-fr] [,FILLET,ang3] $  
  
[ ,CUTCOM,arg1,...,argn[,OFFSET,dis][,ALL  ] $  
          START  
  
[ ,NOMORE] [,MAXDP,maxstp]  
          OFF
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Where:

- “curve”, “circle” or “line” is the wire entity to profile machine. A curve may be of any type (**NCL**, B-spline, Composite, etc.) closed or opened, and planar or non-planar.
- The tool will either drive on this curve or be offset to it. “ON” specifies to drive on the entity and is the default. “OFF” will drive to one side of the entity dependant on the starting point relationship to the entity. The starting point is defined in the “START” parameters.
- The “*thI*” value defines an offset distance that the tool will stay away from the profile on its final pass. When the tool condition is “ON”, then this value will be applied to the tool condition side of the curve without regards to the cutter diameter.
- “AUTO”, “LEFT”, or “RIGHT” specifies the tool condition, when either “OFF” or “*thI*” is specified. This parameter determines the side of the entity to profile. “AUTO” means **NCL** will determine the offset side automatically, “LEFT” will offset the tool to the left of the entity and “RIGHT” will offset the tool to the right of the entity as compared to the initial direction.
- The “PAST” parameters define optional overlap distances that will be applied to the beginning and/or ending of the profile. On a closed profile any extra motion will generate an overlap at the ends of the profile. On an open profile any overlap distance specified will extend the beginning and/or ending of the profile. The single value “*both*” will be applied to both the start and end of the profile. The actual overlap on a closed profile will be twice this value. “*start, end*” specifies separate overlap values for the start and end of the profile.

A negative overlap distance is allowed. When a negative overlap is specified the motion will start at the overlap distance along the curve after the beginning of the curve and end at the overlap distance before the ending of the curve. On a closed profile, the motion will generate a gap at the ends of the profiles.

- The “START” parameters define the startup location, initial direction, and entry method to use. “END” specifies either the beginning point or end point of the curve will be the start location where the profile motion will start. “ENDPT” will use the current tool end point as the starting location. “NEARPT, *ptI*” or “NEARPT, *pvt*” defines an external point or point-vector to use as the starting location. In either case a point on the entity will be calculated from this position and that is where the profile motion will start. In essence, the entity is internally trimmed to this location. In order to start at the beginning of the entity, you can either

6 AUTOMATIC MOTION ROUTINE STATEMENTS

position the tool there prior to profiling it or specify a NEARPT of (POINT/ON,entity,0). The starting location is also used to determine the tool orientation to the entity if “OFF” is in effect.

- The default initial direction for profile machining is the current tool forward direction, which can be changed by specifying either an initial “vector” (dir-vec), “point-vector” or a direction modifier as seen looking down the tool axis. “CLW” specifies that the motion will be generated in a clockwise direction on a closed profile and in the direction that the profile entity was defined in on an open profile. “CCLW” specifies that the motion direction will be in the counter-clockwise direction on a closed profile and opposite the direction that the profile entity was defined in on an open profile.

NOTE: It is strongly recommended to specify the profile direction, otherwise unexpected results might be generated. This is especially true if OFF and either RIGHT or LEFT is specified. The profile motion might be generated on the side opposite from that which you intended to cut. Also motion might not be generated at all for an opened curve if the cutter is located at one end of the curve.

- The entry parameters define how the tool will approach the profile at the beginning of each pass. It can be “OMIT”, “ATANGL”, “ARC” or “ATANGL,...,ARC,...”.
- “OMIT” does not generate any entry motion, the tool will move directly to the start of the profile pass.
- “ATANGL” will generate a linear move of a distance “ dsI ” and at an angle of “ $angI$ ” degrees to the start of the profile on entry.
- “ARC” will generate a 90-degree circular move with a radius of “ rI ” to the start of the profile.
- “ hI ”, “ $h2$ ” specify the rise value or distance above the profile start location for the beginning of the entry move.
- When the entry method is set to “ARC”, then specifying a rise distance other than zero will generate helical entry motion. If the profile condition is set to “ON”, the profile is closed, and the entry method is set to “ATANGL”, then the entry move actually follow the shape of the curve rather than being purely a straight move. In this case the angle is ignored.
- “ATANGL” and “ARC” together specify a combination entry move. “ATANGL” generates a linear move of “ dsI ” and at an angle of “ $angI$ ” to the start of the arc. “ARC” then generates a circular move with a radius of “ rI ” to the start of the profile. The arc will be tangent to the linear entry

6 AUTOMATIC MOTION ROUTINE STATEMENTS

move and the start of the profile. The rises “*h1*” and “*h2*” can be different for the linear and arc moves.

- The “CS” parameters define the ending location of the profile in much the same manner as the starting location. “ENDPT” will use the end point of the entity (or the starting location of the entity if it is closed). “NEARPT, *pt2*” defines an external point to use as the ending location, in which case a point on the entity will be calculated from this position and this is where the profile motion will end. In essence, the entity is internally trimmed to this location.
- The “PS” parameters define the part surface and tool axis mode to use when profiling. The part surface can either be a plane (*pl1*) or surface (*sf1*). If a part surface is not specified, then the profile entity will be machined as it is defined, for example a non-planar entity will be machined in 3-axis, while a planar entity will be machined in 2-axis.
- “*th2*” specifies an offset distance that the tool will stay away from the part surface or offset from the entity along the tool axis if a part surface is not defined.
- “SAME” causes the profile to be machined with a fixed tool axis (as defined by the tool position prior to the start of the profile). “NORMAL” will use a tool axis that is normal to the specified part surface or to a secondary part surface as defined by “*sf2*” or “*pl2*”. If a entity is used as the profile that follows the control of a surface and normal to the part surface tool control is required, then it is recommended that the syntax “PS, NORMAL, *sf2*” be used rather than “PS, *sf*, NORMAL”, as the second syntax required two surface projections per point rather than one and the tool will be projected to the surface rather than projecting onto the surface.
- “CLRSRF” defines the clearance level for Profile Machining and is used to position the tool prior to cutting the profile and can be used as the tool retraction level at the end of the profile motion. It is also used as the clearance level between offset passes of an open profile. “*pl2*” defines a plane to be used as the clearance level. “DIST, *ds2*” specifies a distance above the top plane for the clearance level. “INCR, *ds2*” specifies an incremental distance above the initial profile position when positioning above the entry location. This same distance above the final profile position will be used when retracting to the clearance level at the end of the profile motion and between passes of an open profile. If neither “DIST” nor “INCR” is specified, then “DIST” is assumed. If the “CLRSRF” clause is omitted from the command, then the clearance level will be a plane that goes through the current tool end point and is perpendicular to the current tool axis.
- “RAPTO” defines the location to position the tool using the position feed rate prior to starting the entry move. “*pl4*” defines a plane to be used as the

6 AUTOMATIC MOTION ROUTINE STATEMENTS

positioning level. “DIST,*ds5*” specifies a distance above the top plane for the positioning plane. “INCR,*ds5*” specifies an incremental distance above the initial profile position when positioning above the entry location. If neither DIST nor INCR is specified, then DIST is assumed.

- The “RETRCT” parameters define the tool exit procedure and how the tool will be retracted at the end of the profile motion. “*p15*” defines a plane to retract the tool to at the end of the profile motion. “DIST,*ds5*” specifies a distance above the top plane to retract the tool. ”INCR,*ds5*” specifies an incremental distance above the final profile location to retract the tool to. “ON” retracts the tool to the clearance level, and “OFF” performs no tool retraction at the end of the profile motion, though the tool may still be positioned to the clearance level between passes.
- The exit parameters define how the tool exits each profile pass. It can be “OMIT”, “ATANGL”, “ARC” or “ATANGL,...,ARC,...”.
- “OMIT” does not generate any exit motion, the tool will be positioned at the end of the profile when the retract motion is generated.
- “ATANGL” will generate a linear move of a distance “*ds4*” and at an angle of “*ang2*” degrees to the end of the profile on exit.
- “ARC” will generate a 90-degree circular move with a radius of “*r2*” off of the end of the profile.
- “*h3*”, “*h4*” specify the rise value or distance above the profile end location for the end of the exit move.
- When the exit method is set to “ARC”, then specifying a “*h4*” distance other than zero will generate helical exit motion. If the profile condition is set to “ON”, the profile is closed, and the exit method is set to “ATANGL”, then the exit move will actually follow the shape of the curve rather than being purely a straight move. In this case the angle is ignored.
- “ATANGL” and “ARC” together specify a combination exit move. “ARC” generates a circular move with a radius of “*r2*” to the end of the profile. “ATANGL” then generates a linear move of “*ds4*” and at an angle of “*ang2*” to the end of the arc move. The arc will be tangent to the end of the profile and the start of the linear move. The rises “*h3*” and “*h4*” can be different for the linear and arc moves.
- “RTRCTO, *ds6*” specifies a distance above the ending location of the profile that the tool should retract between depth passes of a closed profile and between offset passes of a closed profile when UP is specified. This value will not be used on open profiles.
- The “OFFSET” parameters controls the output of multiple passes around the periphery of the profile. Multiple passes around the profile will only be generated when the tool condition is “OFF” of the profile.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- “*ds7*” defines a distance from the first pass to the final pass around the profile. “PASS, *n1*” specifies the number of passes to take around the profile. When used with the “*ds7*” parameter, then the distance between passes will be calculated as $ds7/(n1-1)$.
- “*stp1*” defines the maximum step over distance to use when “*ds7*” is specified, the number of passes will be calculated as $ds7/stp1+.9$. When “PASS” is specified, then the distance to the final pass will be calculated as $n1*stp1$.
- “DOWN” causes the transitional moves between the offset passes of a closed profile to be performed with the tool down, in a straight line from the end of one pass to the beginning of the next pass. The ATANGL/ARC exit and entry move specified by the START and RETRCT parameters will only obey by the first entry and last exit move if specified. No ATANGL/ARC move will be generated between loops. However if ARCSLP/FILLET is in effect, the transition moves will be filleted.
- “UP” will exit each of the passes using the exit method specified in the RETRCT clause and enter the next pass using the entry method specified in the START clause. The tool will also be retracted by the distance specified in the RTRCTO clause, if specified.
- If the distance specified in the RTRCTO clause has a value of zero and “OFF” is specified,
- The “STEP” parameters control the output of multiple depth levels around the profile. Multiple depth levels can be defined when the tool is “ON” or “OFF” the profile.
- “*top-plane*” defines a top plane to use as the top of the part. “*ds8*” defines a distance above the lowest level of the profile to use as the top of the part. When “*ds8*” is specified and the tool axis mode is set to “NORMAL”, then the top of the part will be defined as a profile offset from the input entity, instead of as a flat plane.
- “PASS, *n2*” specifies the number of passes to take between the top and the bottom of the part. When used with the “*top-plane*” or “*ds8*” parameter, then the distance between passes will be calculated as $thick/(n-1)$. “*thick*” is the distance between the entity and “*top-plane*” or a distance of “*ds8*” along the initial tool axis. “*stp2*” defines the maximum step down distance to use when “*top-plane*” or “*ds8*” is specified, the number of passes will be calculated as $thick/stp2+.9$. When “PASS” is specified, then the distance to the final depth will be calculated as $n2*stp2$.
- “LEVEL” specifies to machine the “OFFSET” passes at a single depth prior to moving to the next depth. “DEEP” will cause each profile loop to be cut to depth prior to advancing to the next profile offset pass.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- The “FEDRAT” values define the feed rates to use for the various motion types of Profile Machining. “*gen-fr*” defines the feed rate to use during basic motion along the profile. “*pos-fr*” defines the feed rate to use when positioning the tool to and from the clearance and rapto planes. “*ret-fr*” defines the feed rate to use when the tool is retracted at the end of profiling motion and between passes. “*ent-fr*” defines the feed rate to use during the entry move to the profile. “*tran-fr*” defines the feed rate to use when transitioning between loops offset from the profile curve. “*exit-fr*” defines the feed rate to use during the exit move from the profile. A value of zero specified for any of the feed rate settings will use the previously programmed feed rate. “RAPID” will cause the corresponding motion type to be output using rapid. A negative value will be used as a percentage of the programmed feed rate, for example, -0.85 specifies 85% of the programmed feed rate.
- “FILLET, *ang3*” specifies the maximum angle between profile motions allowed before breaking the move up into separate CLfile records. This feature is designed to be used with the [ARCSLP/FILLET](#) command so that fillets can be generated while profiling a sharp cornered curve (typically a composite curve). Any move that is greater or equal to the angle specified will be broken up into two separate CLfile records. It is recommended that you give some play in this value, especially when the curve is being offset and/or projected to a surface. For example, specify 29 if you want corners with a 30 degrees angle to be considered by the ARCSLP/FILLET command.
- The optional “CUTCOM” parameters cause **NCL** to output a “CUTCOM/*arg1,...,argn*” postprocessor statement to the cl file just before the tool making the pass around the profile. “*arg1,...,argn*” are parameters (separated by comma) used to specify the CUTCOM condition which may compose of valid postprocessor vocabulary words, scalers or scaler variables. A “CUTCOM/OFF” postprocessor statement will immediately output to the cl file to reset the CUTCOM condition after the profile motion and prior to the exit move.
- “OFFSET,*dis*” specifies the positional distance that is used to calculate a new entry point after which the CUTCOM commands will be output. With no entry method has been specified, the new entry point will be calculated using “*dis*” on a line perpendicular to the first move. When the “angle”, “arc” or “BOTH” entry methods are used an entry point will be calculated using “*dis*” on a line perpendicular to the entry arc or angle.
- “ALL” will output CUTCOM on all entry moves when machining a closed profile with looping, and “START” will output CUTCOM on just the first entry move.
- “NOMORE” denotes end of CUTCOM parameters and is optional.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

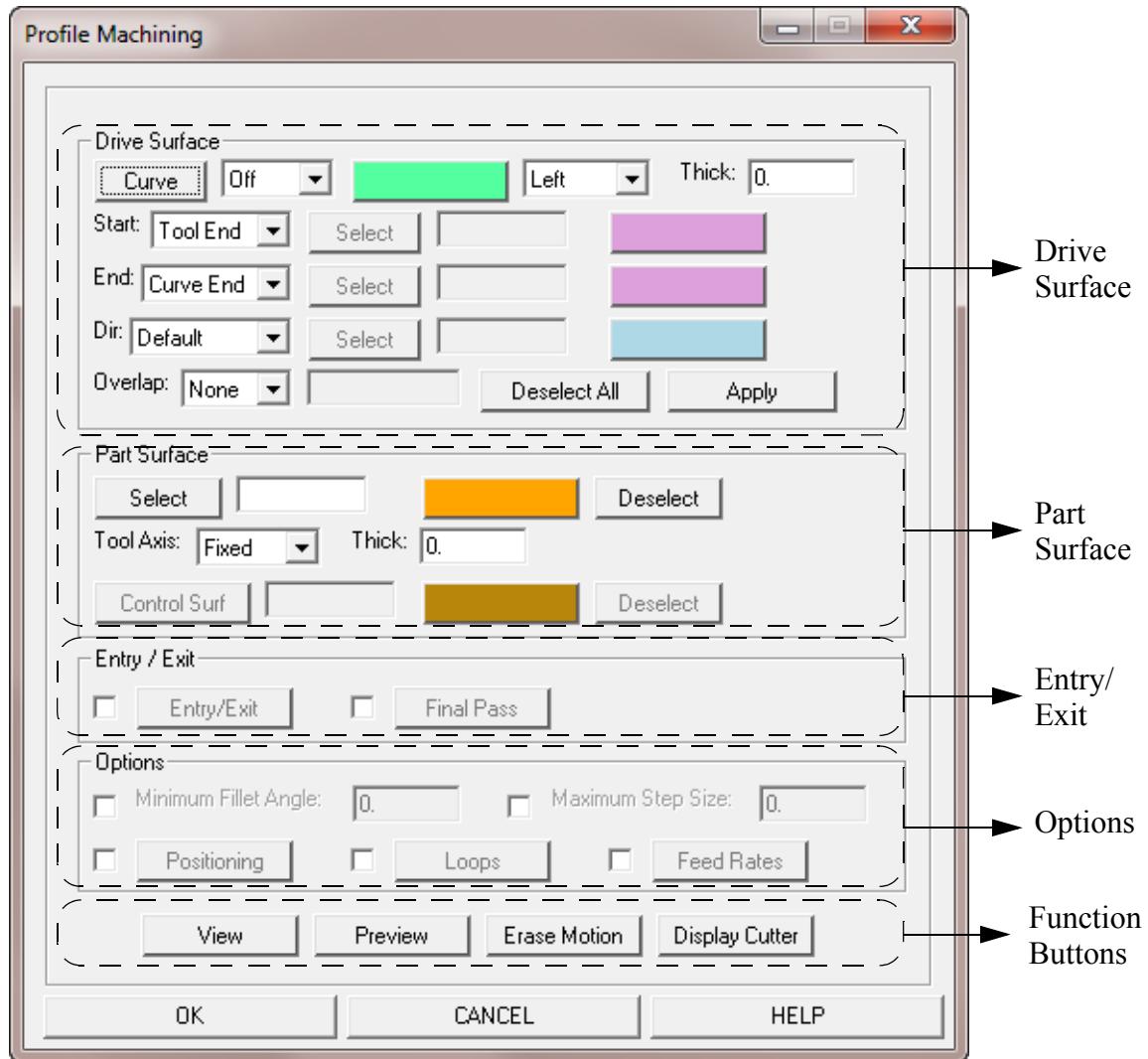
- “MAXDP” allows the user to specify a maximum step size. “maxstp” specifies the maximum distance the tool can move in a single step. This is similar to the MAXDP/STEP,ON setting. “OFF” disables this feature, the motion steps will be spaced to maintain tolerance with the curve. You can also disable the MAXDP feature by specifying a value of zero for “maxstp”.
- The default Profile settings if no parameters are given are as follows:

```
PROFIL/ON,curve,0,AUTO,PAST,0,START,ENDPT,      $  
          OMIT,CS,ENDPT,PS,0,SAME,RAPTO,0,           $  
          RETRCT,OFF,OMIT,RTRCTO,0,                  $  
          OFFSET,PASS,1,DOWN,STEP,PASS,1,LEVEL,       $  
          FEDRAT,0,RAPID,RAPID,0,0,0
```

Following pages show the graphical interactive interface for this **PROFIL** routine and a description of how to use this interface. This interface can be activated by using the following on screen menu icon sequence:



6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form is composed of the following sections: Drive Surface, Part Surface, Entry/Exit, Options and Function Buttons.

Drive Surface

The Drive Surface section controls the settings used for defining the profile to machine.

Curve

Pressing the Curve button allows the user to select the wire entity to profile machine. The entity may be a B-spline, **NCL** Curve, Composite Curve,

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Line, Circle, closed or opened, and planar or non-planar. More than one entities can be picked in one time.

“**On**” specifies that the tool will follow exactly the profile without any offset passes allowed. Multiple depth passes may still be used. Example geometry that will be machined in an “**On**” condition include vacuum lines and engraved letters. “**Off**” offsets the tool from the profile, allowing for multiple offset and depth passes. Profiles that define the edge of the part will be machined in an “**Off**” condition.

The color choice field specifies which color to use to highlight the profile curve.

“**Auto**” specifies that the tool condition, when either “**Off**” or a “**Thick**” value is specified, will determine the side of the entity to profile. “**Left**” will offset the tool to the left of the entity and “**Right**” will offset the tool to the right of the entity as compared to the initial direction.

Thick:

The “**Thick**” value specifies an offset distance that the tool will stay away from the profile. When the tool condition is “**On**”, then this value will be applied to the tool condition side of the entity without regards to the cutter diameter.

Start:

The starting location of the profile can either be specified as the “**Tool End**” point or as a near “**Point**” to the profile curve, or as “**End**” point of the profile curve. In the first two cases a point on the curve will be calculated from this position and that is where the profile motion will start. In essence, the curve is internally trimmed to this location. In order to start at the beginning of the curve, you can either position the tool there prior to profiling it or specify a “**Point**” of “**(PT/ON,entity,0.)**”. In the “**End**” point case, either the beginning point or end point of the curve will be the start location where the profile motion will start.

Select

This button is only active when the **Start** position is a user defined “**Point**”.

Pressing this button allows you to select a point to use as the starting location of the profile.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

The **Color** choice field specifies which color to use to highlight the starting location.

End:

The ending location of the profile can either be specified as the “**Curve End**” point or as a near “**Point**” to the profile entity. When a near “**Point**” is used, a point on the entity will be calculated from this position and that is where the profile motion will end. In essence, the entity is internally trimmed to this location.

If “**Curve End**” is specified and the profile is closed, then the entire profile will be machined, no matter where the starting location is.

Select

This button is only active when the **End** position is a user defined “**Point**”.

Pressing this button allows you to select a point to use as the ending location of the profile.

The **Color** choice field specifies which color to use to highlight the ending location.

Direction

The initial direction of motion around the profile is specified using this field. “**Default**” will use the current forward tool vector. “**Vector**” allows you to specify the name of an existing vector or the i,j,k components of a vector. “**CLW**” specifies that the motion will be generated in a clockwise direction on a closed profile and in the direction that the profile entity was defined in on an open profile. “**CCLW**” specifies that the motion direction will be in the counter-clockwise direction on a closed profile and opposite the direction that the profile entity was defined in on an open profile.

Overlap:

The **Overlap** setting specifies whether any extra motion will be generated at the beginning and/or ending of the profile. On a closed profile, any extra motion will generate an overlap at the ends of the profile. On an open profile, any overlap distance specified will extend the beginning and/or ending of the profile.

“**None**” disables any overlap motion from being generated. “**Both**” specifies that the value entered in the field will be used as the overlap

6 AUTOMATIC MOTION ROUTINE STATEMENTS

distance for both the start and end of the profile. The actual overlap on closed profiles will be twice this value. “**Start**” specifies that the overlap distance should only be applied to the start of the profile. “**End**” applies the overlap distance to the end of the profile.

Entering two values in this field, for example “.1,.05” will cause the first value to be used as the starting overlap distance and the second value to be used as the ending overlap distance, no matter what Overlap setting is specified.

A negative overlap distance is allowed. When a negative overlap is specified the motion will start at the overlap distance along the curve after the beginning of the curve and end at the overlap distance before the ending of the curve. On a closed profile, the motion will generate a gap at the ends of the profiles.

Deselect All

Click this button to deselect all previously selected entities.

Apply

Pick this button to create the PROFIL commands output for all the selected entities without taking down the form so that other PROFIL commands output can be created.

Part Surface

The Part Surface section controls the settings used for defining the part surface and tool axis mode to use when profiling.

Select

Pressing the **Select** button allows the user to select a surface or plane to use as the part surface when profiling. If a part surface is not specified, then the profile entity will be machined as it is defined, for example a non-planar entity will be machined in 3-axis, while a planar entity will be machined in 2-axis.

The **Color** choice field specifies which color to use to highlight the part surface.

Deselect

Pressing the **Deselect** button will deselect the part surface and default back to no part surface being defined.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Tool Axis:

The profile can either be machined with a '**Fixed**' tool axis (as defined by the tool position prior to the start of the profile) or with a tool axis that is '**Normal**' to the part surface or a secondary part surface.

Thick:

The **Thick** value specifies an offset distance that the tool will stay away from the part surface.

Control Surf

Pressing the "**Control Surf**" button allows the user to select a secondary surface that will be used to control the tool axis when "**Normal**" has been specified as the tool axis mode. If a controlling surface (secondary part surface) is not specified, then the part surface entity will be used to control the tool axis when "**Normal**" is specified.

The **Color** choice field specifies which color to use to highlight the tool axis control surface.

Deselect

Pressing the **Deselect** button will deselect the tool axis control surface and default back to the part surface being used to control the tool axis.

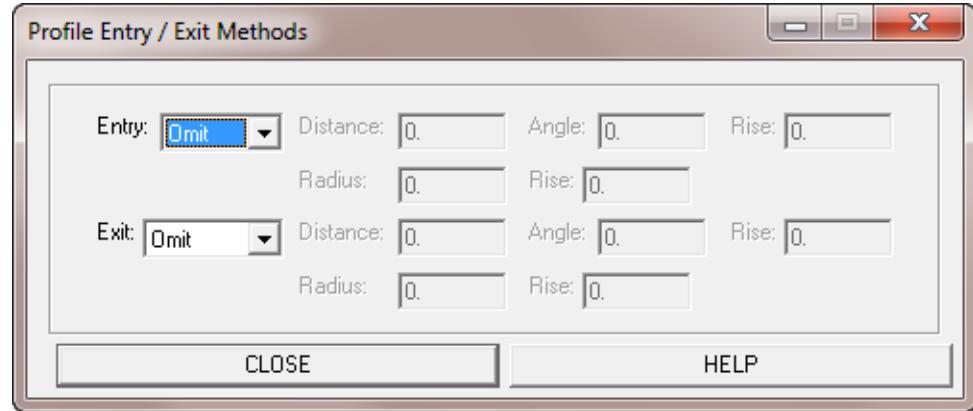
Entry / Exit

The **Entry/Exit** section controls the settings used for entering onto and exiting off of the profile and the final pass parameters for Profile Machining.

Entry/Exit:

Check this box if you would like to specify Entry/Exit methods for the Profile Machining. Pressing the **Entry/Exit** button will display the **Entry/Exit** form as shown on next page

6 AUTOMATIC MOTION ROUTINE STATEMENTS



Entry:

The **Entry** method specifies how the tool enters each profile pass. “**Omit**” does not generate any entry motion, the tool will move directly to the start of the profile pass. “**Angle**” will generate a linear move at an angle to the start of the profile on entry. “**Arc**” will generate a 90-degree circular move to the start of the profile. “**Both**” will generate a combination linear and circular move onto the profile. The linear move will occur first and will always be tangent to the arc move, which transitions between the linear move and the profile. The circular move will always be tangent to the profile.

Distance:

This field is only enabled when the **Entry** method is set to “**Angle**” and specifies the length of the linear entry move.

Radius:

This field is only enabled when the **Entry** method is set to “**Arc**” and specifies the radius of the circular entry arc.

Angle:

This field is only enabled when the **Entry** method is set to “**Angle**” and specifies the angular offset to the initial direction of the profile to use as the entry direction.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Rise:

This field specifies the rise value or distance above the profile start location for the beginning of the entry move. When the **Entry** method is set to “**Arc**”, then specifying a **Rise** distance other than zero will generate helical entry motion.

Exit:

The **Exit** method specifies how the tool exits each profile pass. “**Omit**” does not generate any exit motion, the tool will be positioned at the end of the profile when the retract motion is generated. “**Angle**” will generate a linear move at an angle to the end of the profile on exit. “**Arc**” will generate a 90-degree circular move off of the end of the profile. “**Both**” will generate a combination circular and linear move off of the profile. The circular move will occur first and will always be tangent to the profile and transitions between the profile and the linear move. The linear move will always be tangent to the circular move.

Distance:

This field is only enabled when the **Exit** method is set to “**Angle**” and specifies the length of the linear exit move.

Radius:

This field is only enabled when the **Exit** method is set to “**Arc**” and specifies the radius of the circular exit arc.

Angle:

This field is only enabled when the **Exit** method is set to “**Angle**” and specifies the angular offset to the final direction of the profile to use as the exit direction.

Rise:

This field specifies the rise value or distance above the profile end location for the ending of the exit move. When the **Exit** method is set to “**Arc**”, then specifying a **Rise** distance other than zero will generate helical exit motion.

One special note to entry and exit moves:

6 AUTOMATIC MOTION ROUTINE STATEMENTS

If the profile condition is set to **On**, the profile is closed, and the **Entry/Exit** method is set to **Angle**, then the entry move will actually follow the shape of the entity rather than being purely a straight move. In this case the angle is ignored.

Close

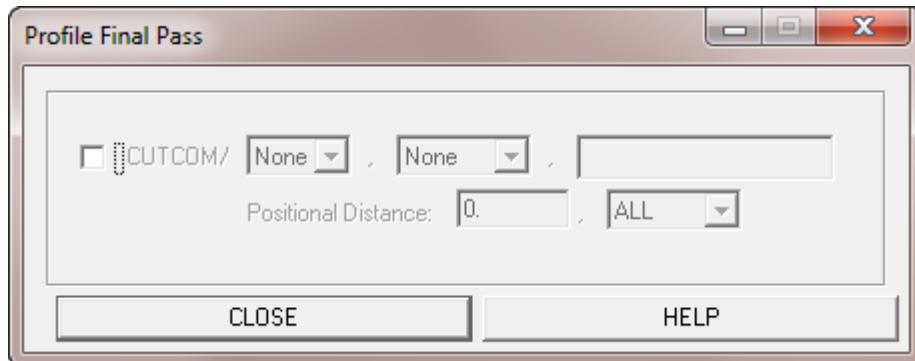
Click this button to accept the changes (if any) and close the form.

Help

Click this button to open the online help for the Profile Entry/Exit Form.

Final Pass

Check this box if you would like to specify the final pass for the Profile Machining. Pressing the **Entry/Exit** button will display the *Profile Final Pass* form as shown below.



CUTCOM

If this box is checked, then the CUTCOM command defined by the CUTCOM parameters will be output to the clfile just prior to making the final pass around the profile. A CUTCOM/OFF post-processor statement will be output to the clfile to reset the CUTCOM condition after the profile motion and prior to the exit move.

Positional Distance

The positional distance is used to calculate a new entry point after which the CUTCOM command will be output.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

ALL/START

All will output CUTCOM on all entry moves when machining a closed profile with looping and START will output CUTCOM on just the first entry move.

Close

Click this button to accept the changes (if any) and close the form.

Help

Click this button to open the online help for the Profile Final Pass Form.

Options

The **Options** section controls miscellaneous options for Profile Machining and provides access to other Profile forms.

Minimum Fillet Angle:

Check this box if you would like to have the profile motion broken up into multiple clfile records so that the [ARCSLP/FILLET](#) command can be used to generate fillets in sharp corners along the profile. Any move that is greater than or equal to the angle specified will be broken up into two separate clfile records. It is recommended that you give some play in this value, especially when the entity is being offset and/or projected to a surface. For example, specify 29 if you want corners with a 30 degree angle to be considered by the ARCSLP/FILLET command.

Maximum Step Size:

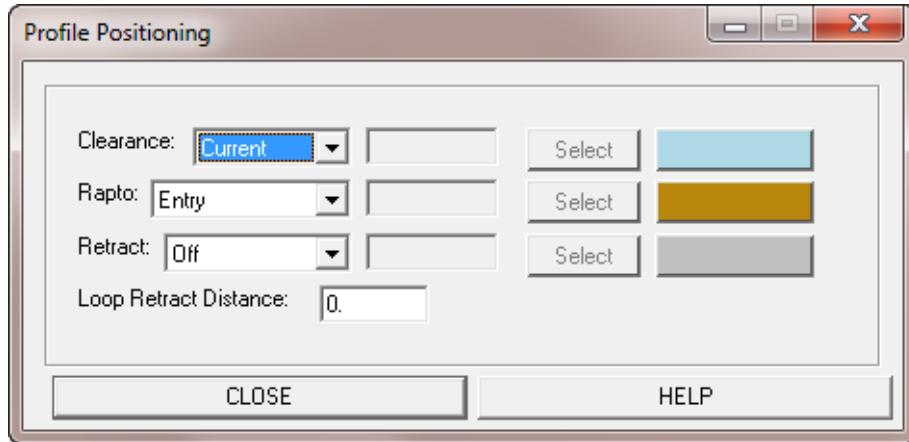
Check this box to specify the maximum distance the tool can move in a single step. Motion steps generated by Profile Machining that exceed this distance will have multiple points output so that no motion step exceeds this value. Disabling this box causes the Profile Machining motion steps to be calculated and output using the machining tolerance.

Positioning

Check this box if you would like to specify positioning options for the Profile Machining. Pressing the **Positioning** button will display the *Profile Positioning form* as shown on next page, which is used to define the

6 AUTOMATIC MOTION ROUTINE STATEMENTS

positioning settings and values. Unchecking this box will not utilize any of the positioning settings no matter how they are set.



Clearance

The clearance level of Profile Machining is used to position the tool prior to cutting the profile and can be used as the tool retraction level at the end of the end of the profile motion. It is also used as the clearance level between offset passes of an open profile.

"Current" uses the current tool position to specify the clearance level. A clearance plane will be created that goes through the tool end point and is perpendicular to the tool axis.

"Plane" allows the user to specify/select a predefined plane.

"Distance" specifies a distance above the top of the part level for the clearance plane.

"Incremental" specifies an incremental distance above the entry point when positioning above the entry location.

This same distance above the exit point will be used when retracting to the clearance level at the end of the profile motion and between loops of an open profile.

Select

This button is only active when the Clearance type is set to **"Plane"**. Pressing this button allows you to select the plane or planar surface to use as the clearance plane.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Color

The color choice field specifies which color to use to highlight the clearance plane.

Rapto

This specifies the distance above the entry location to position the tool at the Positioning feed rate. The remainder of the entry move will be made at the Entry feed rate.

'Entry' will position to the start of the entry move using the Positioning feedrate. If an entry move is not programmed, then the tool will move to the start of the profile using the General feed rate.

'Plane' allows the user to specify/select a predefined plane.

'Distance' specifies a distance above the top of the part level for the rapto plane.

'Incremental' specifies an incremental distance above the first point of the profile to use as the rapto plane.

Select

This button is only active when the Rapto type is set to **"Plane"**. Pressing this button allows you to select the plane or planar surface to use as the rapto plane.

Color

The color choice field specifies which color to use to highlight the rapto plane.

Retract

This specifies the retract logic to use at the end of the profile motion, between depth passes, and between offset passes of an open profile.

"Off" performs no tool retraction at the end of the profile motion, though the tool may still be positioned to the clearance plane between passes.

"On" retracts the tool to the Clearance level at the end of the profile motion.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

“**Plan**” allows the user to specify/select a predefined plane.

“**Distance**” specifies a distance above the top of the part level to retract the tool.

“**Incremental**” specifies an incremental distance above the last point of the profile to use as the retract plane.

Select

This button is only active when the Retract type is set to “**Plane**”. Pressing this button allows you to select the plane or planar surface to use as the retract plane.

Color

The color choice field specifies which color to use to highlight the retract plane.

Loop Retract Distance

The Loop Retract Distance specifies a distance above the ending location of the profile that the tool should retract between depth passes of a closed profile.

This value will be ignored on open profiles and between offset passes.

Final Retract

This field allows you to specify the retract logic to use at the end of the profile motion, between depth passes, and between offset passes of an open profile.

“**Off**” performs no tool retraction at the end of the profile motion, though the tool may still be positioned to the clearance plane between passes.

“**On**” retracts the tool to the Clearance level at the end of the profile motion.

“**Plane**” allows the user to specify/select a predefined plane.

“**Distance**” specifies a distance above the exit location to retract the tool.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Select

This button is only active when the Final Retract type is set to “**Plane**”.

Pressing this button allows you to select the plane or planar surface to use as the retract plane.

Color

The color choice field specifies which color to use to highlight the retract plane.

Close

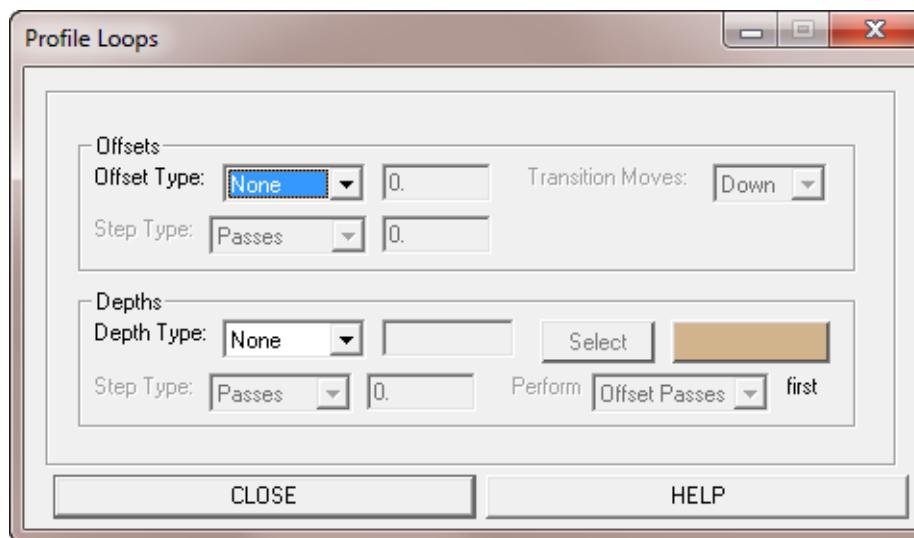
Click this button to accept all the changes (if any) and close the form.

Help

Click this button to open the online help for the Profile Positioning Form.

Loops

Check this box if you would like to use offset and/or depth passes with Profile Machining. Pressing the **Loops** button will display the *Profile Loops form* as shown below, which is used to define the offset and depth pass settings and values. Unchecking this box will not generate multiple offset or depth passes no matter how they are defined in the *Profile Loops form*.



6 AUTOMATIC MOTION ROUTINE STATEMENTS

This form is composed of two sections: **Offsets** and **Depths**.

Offsets

The **Offsets** section controls the output of multiple passes around the profile. Multiple passes around the profile can only be specified when the tool is “**Off**” the profile.

Offset Type

Controls the *type of offsets* to apply to the profile motion. “**None**” will take a single pass around the profile, “**Thickness**” defines the distance from the *first pass* to the *final pass* around the profile, and “**Passes**” defines the number of passes to take around the profile.

A value must be entered when either “**Thickness**” or “**Passes**” is selected.

Transition Moves

The transition moves between passes of a closed curve can be made with the tool moving directly to the next pass or by exiting and re-entering the part using the programmed exit and entry methods.

Select “**Down**” if the tool should move directly between the passes. The [ARCSLP/FILLET](#) command can be used in this case to create an S-shaped transitional move.

Select “**Up**” to have the tool exit the current pass and enter the next pass using the programmed exit and entry methods.

The [Loop Retract Distance](#) will be used to retract the tool after exiting the part between passes.

Select “**Off**” to disable the automatic retract if the next tool entry level is higher than the retract level when the loop Retract distance is set to 0.

Step Type

This field will only be enabled when the **Offset Type** is not set to “**None**”.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

“**Passes**” will use the number of passes entered in this field to calculate the step over distance and can only be specified when the **Offset Type** is set to “**Thickness**”. “**Max-step**” specifies the (maximum) step of each offset pass.

When the **Offset Type** is set to “**Passes**”, this value is the actual step over distance used.

Depths

The **Depths** section controls the output of multiple depth passes around the profile. Multiple depth passes around the profile can be defined when the tool is “**On**” or “**Off**” the profile.

Depth Type

Controls the *type of depths* to apply to the profile motion. “**None**” will perform at a single depth around the profile, “**Top Plane**” defines a plane to use as the top of the part. The first pass will be the calculated depth of cut beneath this plane. “**Distance**” defines the distance from the calculated bottom of the profile (lowest point on the profile) to the top of the part. “**Passes**” defines the number of depth passes to take around the profile.

Select

This button is only active when the **Depth Type** is set to “**Top Plane**”. Pressing this button allows you to select the plane or planar surface to use as the top of the part.

Color

The color choice field specifies which color to use to highlight the top of the part.

Step Type

This field will only be enabled when the **Depth Type** is not set to “**None**”.

“**Passes**” will use the number of passes entered in this field to calculate the depth distance and can only be specified when the **Depth Type** is set to “**Top Plane**” or “**Distance**”. “**Max-step**”

6 AUTOMATIC MOTION ROUTINE STATEMENTS

specifies the (maximum) step of each depth pass. When the **Depth Type** is set to “**Passe**”, this value is the actual step down distance used.

Perform --- first

This field specifies whether the “**Offset Passes**” or “**Depth Passes**” should be machined first. “**Offset Passes**” causes the tool to move in towards the profile prior to moving down in depth. “**Depth Passes**” causes the tool to cut each profile loop to depth prior to advancing to the next profile offset pass.

Close

Click this button to accept all the changes (if any) and close the form.

Help

Click this button to open the online help for the Profile Loops Form.

Feed Rates

Check this box if you would like to specify feed rates for the various motion types of Profile Machining. Pressing the **Feed Rates** button will display the *Profile Feed Rates form* as shown below, which is used to define the feed rates used with Profile Machining. Unchecking this box will use the default feed rate values in effect at this time.



Each feed rate has the following modes that can be programmed.

Current = Uses the currently programmed feed rate.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Value = Allows you to specify an absolute value as the feed rate.

Rapid = Uses RAPID.

Factor = Enter a percentage of the programmed feed rate to use, for example:

.85 specifies 85% of the programmed feed rate.

General

Defines the feed rate to use during the basic motion along the profile.

Position

Defines the feed rate to use when positioning the tool to and from the clearance plane and rapto plane.

Retract

Defines the feed rate to use when the tool is retracted at the end of the profiling motion and between passes.

Entry

Defines the feed rate to use during the entry move to the profile.

Transition

Defines the feed rate to use when transitioning between "loops" offset from the profile curve.

Exit

Defines the feed rate to use during the exit move from the profile.

Close

Click this button to accept all the changes (if any) and close the form.

Help

Click this button to open the online help for the Profile Feed Rate Form.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Function Buttons

View

Takes down the form(s) and enters dynamic viewing.

Preview

Previews the Profile Machining motion. No command is output.

Erase Motion

Erases all motion.

Display Cutter

Displays the current cutter position.

OK:

Click this button to accept the setting, close the form, generate the motion and output all the corresponding source codes to the part program.

Cancel:

Click this button to disregard all the changes (if any) and close the form.

Help:

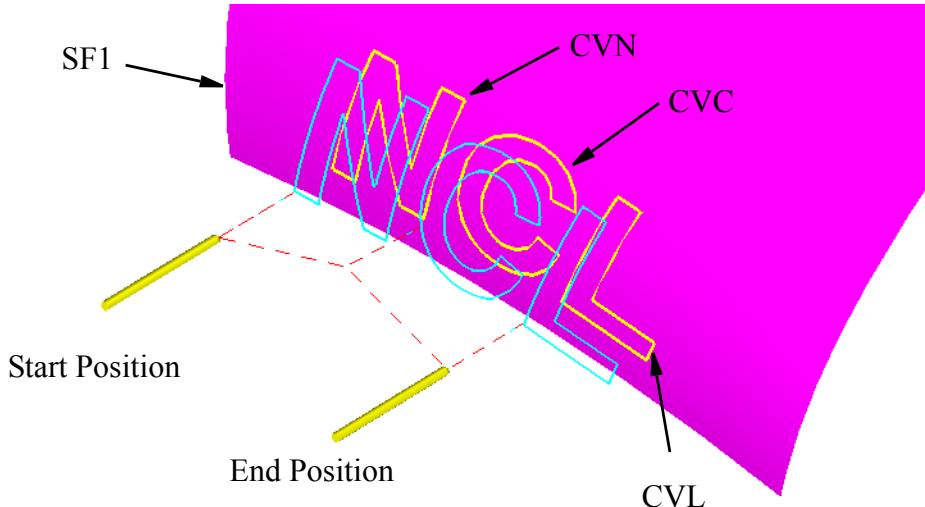
Click this button to open the online help for the Profile Machining Form.

Example PROFIL command sequence:

```
MULTAX/ON
CUTTER/0.05,0.025,0.75
DNTCUT
GOTO / ( PNTVEC/ ( POINT/ON, CVN, 0 ), SF1 )
GODLTA/0.75
RAPID
CUT
PROFIL/ON, CVN, PS, 0.25, NORMAL, SF1, RAPTO, 0.1,           $
      RETRCT, ON
PROFIL/ON, CVC, PS, 0.25, NORMAL, SF1, RAPTO, 0.1,           $
      RETRCT, ON
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

```
PROFIL/ON,CVL,PS,0.25,NORMAL,SF1,RAPTO,0.1,$  
RETRCT,ON
```



PROFIL motion is normal to SF1 and at a distance of 0.25 above SF1.

6.29.2 PROFIL - Annotation Engraving

This version of PROFIL statement is used to engrave annotation (lettering). The valid syntax for this statement is:

```
PROFIL/anote[,PS,depth][,CLRSRF,plane1]      $  
[DIST,]dis1  
INCR  
  
[,RAPTO,plane2]      $  
[DIST,]dis2  
INCR  
  
[,RETRCT,plane3]      [,RTRCTO,dis4]]      $  
ON  
OFF  
[DIST,]dis3  
INCR  
  
[,STEP,top-plane,maxstep]  
dis5      PASS,n  
PASS,n[,maxstep]  
  
[,FEDRAT,gen-fr, pos-fr, ret-fr, ent-fr]
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Where:

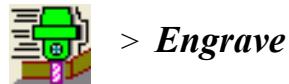
- “anote” is the label of the annotation to be engraved.
- The tool axis will always be the same as the plane vector of the annotation for 2-D annotation or normal to the surface that the annotation was projected.
- “PS,depth” defines the distance below the defined annotation to machine.
- “CLRSRF” defines the clearance level for Annotation Engraving and is used to position the tool prior to cutting the annotation and can be used as the tool retraction level at the end of the engraving motion. “*plane1*” defines a plane to be used as the clearance level. “DIST, *ds1*” specifies a distance above the top plane for the clearance level. “INCR, *ds1*” specifies an incremental distance above the initial engraving position when positioning above the entry location. This same distance above the final engraving position will be used when retracting to the clearance level at the end of the engraving motion and between passes. If neither “DIST” nor “INCR” is specified, then “DIST” is assumed. If the “CLRSRF” clause is omitted from the command, then the clearance level will be a plane that goes through the current tool end point and is perpendicular to the current tool axis.
- “RAPTO” defines the location to position the tool using the position feed rate prior to starting the entry move. “*plane2*” defines a plane to be used as the positioning level. “DIST,*ds2*” specifies a distance above the top plane for the positioning plane. “INCR,*ds2*” specifies an incremental distance above the initial engraving position when positioning above the entry location. If neither DIST nor INCR is specified, then DIST is assumed.
- The “RETRCT” parameters define the tool exit procedure and how the tool will be retracted at the end of the engraving motion. “*plane3*” defines a plane to retract the tool to at the end of the engraving motion. “DIST,*ds3*” specifies a distance above the top plane to retract the tool. ”INCR,*ds3*” specifies an incremental distance above the final engraving location to retract the tool to, “ON” retracts the tool to the clearance level, and “OFF” performs no tool retraction at the end of the engraving motion, though the tool may still be positioned to the clearance level between passes.
- “RTRCTO,*dis4*” specifies the distance above the top of the part to retract the tool to when moving between the polylines of an annotation. If this parameter is not specified, then the standard RETRCT parameter settings will be used when moving between polylines.
- The “STEP” parameters control the output of multiple depth levels around the engraving.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

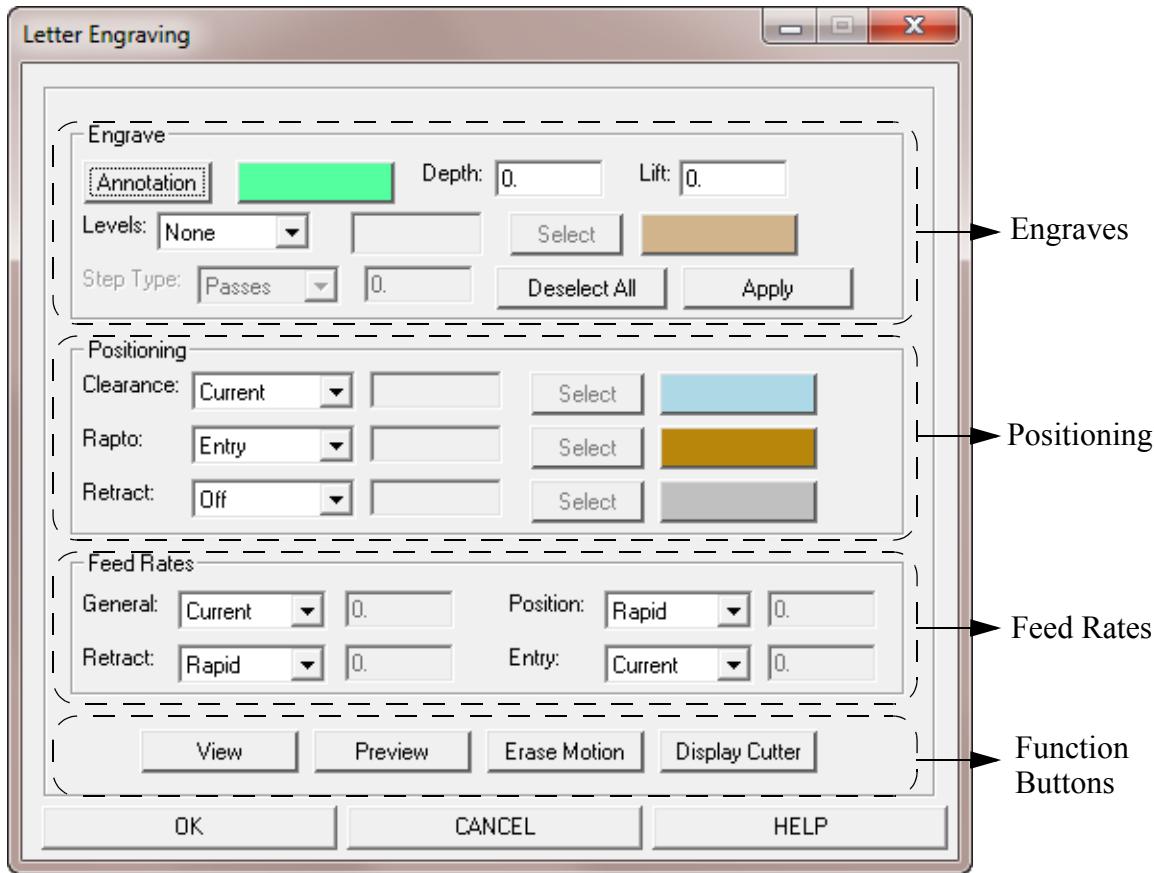
- “*top-plane*” defines a top plane to use as the top of the part. “*ds5*” defines a distance above the lowest level of the engraving to use as the top of the part. When “*ds5*” is specified then the top of the part will be defined as an offset from the input annotation, instead of as a flat plane.
- “PASS, *n*” specifies the number of passes to take between the top and the bottom of the part. When used with the “*toppl*” or “*ds5*” parameter, then the distance between passes will be calculated as $\text{thick}/(n-1)$. “*thick*” is the distance between the entity and “*toppl*” or a distance of “*ds5*” along the initial tool axis. “*maxstep*” defines the maximum step down distance to use when “*toppl*” or “*ds5*” is specified, the number of passes will be calculated as $\text{thick}/\text{maxstep}+.9$. When “PASS” is specified, then the distance to the final depth will be calculated as $n*\text{maxstep}$.
- The “FEDRAT” values define the feed rates to use for the various motion types of Profile Machining. “*gen-fr*” defines the feed rate to use during basic motion along the profile. “*pos-fr*” defines the feed rate to use when positioning the tool to and from the clearance and rapto planes. “*ret-fr*” defines the feed rate to use when the tool is retracted at the end of profiling motion and between passes. “*ent-fr*” defines the feed rate to use during the entry move to the profile. “*tran-fr*” defines the feed rate to use when transitioning between loops offset from the profile curve. “*exit-fr*” defines the feed rate to use during the exit move from the profile. A value of zero specified for any of the feed rate settings will use the previously programmed feed rate. “RAPID” will cause the corresponding motion type to be output using rapid. A negative value will be used as a percentage of the programmed feed rate, for example, -0.85 specifies 85% of the programmed feed rate.

Note: The THICK command will not obey by this PROFIL command.

Following pages show the graphical interactive interface for this **PROFIL** routine and a description of how to use this interface. This interface can be activated by using the following on screen menu icon sequence:



6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form is composed of the following sections: Engrave, Positioning, Feed Rates and Function Buttons.

Engrave

The Engrave section controls the settings used for the actual engraving of the annotation.

Annotation

Pressing the Annotation button allows the user to select the annotation to engrave. Multiple annotations may be selected. A separate PROFIL command will be output for each annotation.

Annotation created in **NCL V9.6** or earlier is not labeled and cannot be selected for engraving. If multiple entities are selected for engraving and at least one is a labeled annotation, then no error message will be output but

6 AUTOMATIC MOTION ROUTINE STATEMENTS

any unlabeled annotation will be ignored. If all entities selected are unlabeled annotations, then an error message will be displayed stating that no valid entities have been selected.

Unlabeled annotation that is renamed with a valid label can be engraved.

The **Color** choice field specifies which color to use to highlight the annotation.

Depth

The Depth value specifies the depth below the top level of the annotation to machine to. The top level defaults to the defined level of the annotation or can be changed using the Levels field.

Lift

The Lift value specifies the distance above the top level of the annotation to position the tool to when moving between the line segments of the annotation.

Levels

Controls the top level of the annotation.

“**None**” will engrave the annotation at the level that the annotation is defined (minus the depth value).

“**Top Plane**” defines the top of the part and is used instead of the annotation level.

“**Distance**” defines the distance from the calculated bottom of the annotation (lowest point) to the top of the part.

“**Passes**” uses the top of the annotation as the top machining level and defines the number of passes to take to machine to the specified depth level.

Select

This button is only active when the Levels is set to “**Top Plane**”. Pressing this button allows you to select the plane or planar surface to use as the top of the part.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

The **Color** choice field specifies which color to use to highlight the top of the part.

Step Type

This field will only be enabled when the Levels field is not set to “**None**”

“**Passes**” will use the number of passes entered in this field to calculate the depth distance and can only be specified when the Levels field is set to “**Top Plane**” or “**Distance**”. “**Max-step**” specifies the (maximum) step of each depth pass. When the Levels field is set to ”**Passes**”, this value is the actual step down distance used.

Deselect All

Deselects all previously selected annotation.

Apply

Creates the annotation without taking down the form so that other annotations can be created.

Positioning

The Positioning section controls the positioning and tool retraction features of Letter Engraving.

Clearance:

The clearance level of Letter Engraving is used to position the tool prior to engraving the annotation and can be used as the tool retraction level at the end of the engraving motion.

“**Current**” uses the current tool position to specify the clearance level. A clearance plane will be created that goes through the tool end point and is perpendicular to the tool axis. “**Plane**” allows the user to specify/select a predefined plane. “**Distance**” specifies a distance above the top of the part level for the clearance plane. “**Incremental**” specifies an incremental distance above the entry point when positioning above the entry location.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Select

This button is only active when the Clearance type is set to “IB”. Pressing this button allows you to select the plane or planar surface to use as the clearance plane.

The **Color** choice field specifies which color to use to highlight the clearance plane.

Rapto

The Rapto field specifies the distance above the entry location to position the tool at the Positioning feed rate. The remainder of the entry move will be made at the Entry feed rate.

“**Entry**” will position to the start of the entry move using the **Positioning** feed rate. “**Plane**” allows the user to specify/select a predefined plane. “**Distance**” specifies a distance above the top of the part level for the rapto plane. “**Incremental**” specifies an incremental distance above the first point of the annotation to use as the rapto plane.

Select

This button is only active when the Rapto type is set to “**Plane**”. Pressing this button allows you to select the plane or planar surface to use as the rapto plane.

The **Color** choice field specifies which color to use to highlight the rapto plane.

Retract

This field allows you to specify the retract logic to use at the end of the letter engraving.

“**Off**” performs no tool retraction at the end of the engraving motion. “**On**” retracts the tool to the Clearance level at the end of the engraving motion.

“**Plane**” allows the user to specify/select a predefined plane. “**Distance**” specifies a distance above the top of the part level to retract the tool. “**Incremental**” specifies an incremental distance above the last point of the annotation to use as the retract plane.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Select

This button is only active when the Final Retract type is set to “*Plane*”.

Pressing this button allows you to select the plane or planar surface to use as the retract plane.

The **Color** choice field specifies which color to use to highlight the retract plane.

Feed Rates

The Feed Rates section defines the feed rates to use when Engraving Letters. Each feed rate has the following modes that can be programmed.

Current = Uses the currently programmed feed rate.

Value = Allows you to specify an absolute value as the feed rate.

Rapid = Uses RAPID.

Factor = Enter a percentage of the programmed feed rate to use, for example .85 specifies 85% of the programmed feed rate.

General:

Defines the feed rate to use during the basic engraving motion.

Position

Defines the feed rate to use when positioning the tool to and from the clearance plane and rapto plane.

Retract

Defines the feed rate to use when the tool is retracted at the end of the engraving motion and between line segments.

Entry

Defines the feed rate to use during the entry move.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Function Buttons

View

Takes down the form(s) and enters dynamic viewing.

Preview

Previews the Profile Machining motion. No command is output.

Erase Motion

Erases all motion.

Display Cutter

Displays the current cutter position.

OK:

Click this button to accept the setting, close the form, generate the motion and output all the corresponding source codes to the part program.

Cancel:

Click this button to disregard all the changes (if any) and close the form.

Help:

Click this button to open the online help for the Profile Machining Form.

6.30

RMILL

The RMILL statement generates back and forth type tool motion to machine a region of a part (Region MILL). It is based on the concept of machining a region of a SURF or a PLANE that is bounded by two ends and two sides. RMILL generates either a LACE or SCRUB type series of motions. The following example explains the difference.

```
RMILL/SF1,ON,CV1,ON,CV2,ON,PL1,ON,PL2,1,  
CPL,.1,2,.125,30,RAPID,10 $
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

This statement specifies a region of SF1 to be milled from CV1 to CV2 along a series of planes from PL1 to PL2. It specifies the following LACE type cutting motion to be generated. The cutter will start at the intersection of CV1 and PL1, with SF1 as the part surface, and will move along PL1 until the cutter is ON CV2. The cutter will then retract up to the clearance plane CPL at a RAPID feed rate. It will then move back along PL1 until it is ON CV1.

The cutter will then RAPID down until it is within .1 (plunge distance) of SF1. Changing the feed rate to 10 IPM (plunge feed rate), the cutter will continue down until contacting SF1 again. The next cutting plane (drive surface) is calculated by **NCL** such that a .125 scallop height at the CV1 check surface end will be left. The cutter then steps over to that internally generated drive plane at 30 IPM (general feed rate) and another lace motion is performed.

Since the cutter retracted to a clearance plane at the end of each pass and went back to CV1 before dropping back to SF1, all the cutting passes would be either climb cutting or conventional cutting, depending upon the situation.

The other motion type of tool path (SCRUB) would drive the tool from CV1 to CV2 along PL1. It would then drive along CV2 towards PL2 until it was ON the internally generated step-over plane, then drive back along that plane until it was ON CV1. At that point, it would drive along CV1 towards PL2 until it was on the next internally generated step-over plane and start the pattern over again until PL2 was eventually reached. During this motion, the tool would never leave SF1. The motion generated would climb cut in one direction and conventional cut in the other direction.

The valid syntax construct for the RMILL statement is shown on next page:

6 AUTOMATIC MOTION ROUTINE STATEMENTS

RMILL/part-surf	\$
[, check-1-modifier], check-sf-1	\$
[, check-2-modifier], check-sf-2	\$
[, drive-1-modifier], drive-sf-1	\$
[, drive-2-modifier], drive-sf-2	\$
, motion-type	\$
[, clearance-plane-or-dist, plunge-dist]	\$
, stepover-type, stepover-dist	\$
, general-fedrat, positioning-fedrat	\$
, plunge-fedrat	\$
[, retract-plane-point-or-dist]	\$
[, ROUGH, rcsth1[, rcsth2[, rdsth1[, rdsth2]]]]	\$
[, FINISH, fcsth1[, fcsth2[, fdsth1[, fdsth2]]]]	\$

Where:

part-surf:

- The SURF or PLANE which controls the bottom of the cutter.

check-1-modifier:

- The vocabulary word TO, ON or PAST. This controls the relationship of the cutter to the first check surface geometry. When the cutter approaches the check surface geometry, it will either stop as it touches it (TO), is directly on it (ON), or after it is past it (PAST).

check-sf-1:

- The geometry that will limit the cutter motion on the region and can be thought of as a check surface for the back and forth motion that is generated. This can be any of the following types of geometry:

LINE, CIRCLE, PLANE, CURVE or SURF

6 AUTOMATIC MOTION ROUTINE STATEMENTS

check-2-modifier:

- The vocabulary word TO, ON or PAST. This is identical to the check-1-modifier except that it applies to the check-sf-2 geometry.

check-sf-2:

- The other geometry that will limit the cutter motion along with check-sf-1. This can be any of the following types of geometry:

LINE, CIRCLE, PLANE, CURVE or SURF

drive-1-modifier:

- The vocabulary word TO, ON or PAST. This is similar to the check-1-modifier and specifies the tool relationship to the drive-sf-1 geometry during motion generation.

drive-sf-1:

- The geometry that is the limit of one side of the region to be milled and can be thought of as the drive surface for the back and forth motion that is to be generated. This can be either a PLANE or a LINE. It does not need to be parallel to the drive-sf-2 geometry. Cutter motion begins at the corner where drive-sf-1 and check-sf-1 intersect.

drive-2-modifier:

- The vocabulary word TO, ON or PAST. This is identical to the drive-1-modifier except that it applies to the drive-sf-2 geometry.

drive-sf-2:

- The geometry that will limit the other side of the region along with drive-sf-1. This can be either a PLANE or a LINE. It does not need to be parallel to the drive-sf-1 geometry. Cutter motion ends at the corner where drive-sf-2 and check-sf-2 intersect.

motion-type:

- Specifies the type of motion to be generated. Valid values are:

6 AUTOMATIC MOTION ROUTINE STATEMENTS

1 = LACE cutting	-1 = LACE profile
2 = SCRUB cutting	-2 = SCRUB profile

- See the foregoing explanation for the difference between the two motion styles.
- The profile options add a finish profile cut to the two check surfaces immediately following the LACE or SCRUB cut.

clearance-plane-or-dist:

- The plane the cutter will move to at the end of each pass across the region during a LACE cutting operation. The cutter is then moved back to the proper relationship with the check-sf-1 geometry at the height of the plane.
- A surface of primitive type "**PLANAR**" can be used as the clearance plane.
- If a distance value is given instead of a plane, the cutter will be moved delta up the tool axis that distance then to a point that is delta up the tool axis the distance from where it will contact the part surface for the next cutting pass. This parameter may and must only be specified for a LACE motion type RMILL statement.
- A distance value of "0" must be specified if a "plunge-dist" is specified for a SCRUB motion type RMILL statement.

plunge-dist:

- The distance away from the part surface the cutter will be moved to as it moves down the tool axis from the clearance plane or distance at the positioning feed rate. The cutter then moves from this point to contact with the part surface at the plunge-feedrate.
- The "*clearance-plane-or-dist*" parameter must have a value of "0" if this "*plunge-dist*" parameter is specified for a SCRUB motion type RMILL statement.

stepover-type:

- The type of stepover calculations to be made. This is the distance the cutter is moved at the end of each cutter pass. Valid values are:

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- 1 = Set the stepover to be a fixed distance given by the stepover-dist value.
- 2 = Set the stepover based on the stepover-dist value being the scallop height between passes at the check-sf-1 end of the region.

stepover-dist:

- The value used to calculate the distance between each pass. See the explanation for stepover-type above for more details.

general-fedrat:

- The feed rate used for all motion other than during positioning and plunging.

positioning-fedrat:

- The feed rate used during initial positioning, LACE cut moves back to the check-sf-1 geometry and the optional final retraction move.

plunge-fedrat:

- The feed rate used for moves down to the part surface from the plunge-dist distance above the surface.

retract-plane-point-or-dist:

- Specifies the place where the cutter will be retracted to at the completion of all motion across the region.
- This may be a PLANE, POINT, SURFACE of primitive type "PLANAR" or a distance to move up the tool axis from the ending position.

ROUGH, rcsth1 [, rcsth2 [, rdsth1 [, rdsth2]]]:

- Specifies the optional over-riding check/drive surfaces thickness values within RMILL motion. Once the RMILL motion finished, the original **THICK** parameters will be restored for subsequent motion generation.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- ROUGH is required for the syntax.
- Where:

rcsth1 : First Check Surface THICK value.
rcsth2 : Second **Check Surface** THICK value.
rdsth1 : First **Drive Surface** THICK value.
rdsth2 : Second Drive Surface THICK value.

- If only rcsth1 is specified, just the first check surface thick value will be overridden. The rest will use the drive surface thick specified in the regular THICK statement.
- If only rcsth1 and rcsth2 are specified, just the first and the second check surfaces thick values will be overridden. The two drive surfaces will use the drive surface thick value specified in the regular THICK command.
- If only rcsth1, rcsth2 and rdsth1 are specified, just the first, second check surfaces and the first drive surface thick values will be overridden. The second drive surfaces will use the drive surface thick value specified in the regular THICK command.
- The ROUGH parameters will be applied to the final profile motion within RMILL for the motion type lace-profile or scrub-profile if "FINISH" is not specified.

FINISH, fcsth1 [, fcsth2 [, fdsth1 [, fdsth2]]]:

- Specifies the optional over-riding check/drive surfaces thickness values of the final profile move for either the motion type lace-profile or scrub-profile within RMILL motion. This has no effect if the motion type is not lace-profile or scrub-profile. Once the RMILL motion finished, the original **THICK** parameters will be restored for subsequent motion generation.
- FINISH is required for the syntax.
- Where:

fcsth1 : First Check Surface THICK value.
fcsth2 : Second Check Surface THICK value.
fdsth1 : First Drive Surface THICK value.
fdsth2 : Second Drive Surface THICK value.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- If only fcsth1 is specified, just the first check surface thick value will be overridden. The rest will use the drive surface thick specified in the regular THICK statement.
- If only fsth1 and fcsth2 are specified, just the first and the second check surfaces thick values will be overridden. The two drive surfaces will use the drive surface thick value specified in the regular THICK command.
- If only fcsth1, fsth2 and fdsth1 are specified, just the first, second check surfaces and the first drive surface thick values will be overridden. The second drive surfaces will use the drive surface thick value specified in the regular THICK command.
- If "FINISH" is not specified and the motion type is either lace-profile or scrub-profile, the override values for ROUGH will be applied to the final profile motion within RMILL.

Any valid cutter type and all forms of **TLAXIS** may be used for regional milling. However, it is recommended that only "**TLAXIS/ TT, DS, height, FAN**" and "**TLAXIS/ NORMAL, PS**" be used.

The current THICK parameters for part surface and drive surface are honored by RMILL. The check surface thick parameters will not be honored by RMILL, instead the current drive surface thick parameter is used for both the check and drive surfaces within RMILL motion.

The secondary feed rates are ignored within RMILL motion.

Example RMILL Statements:

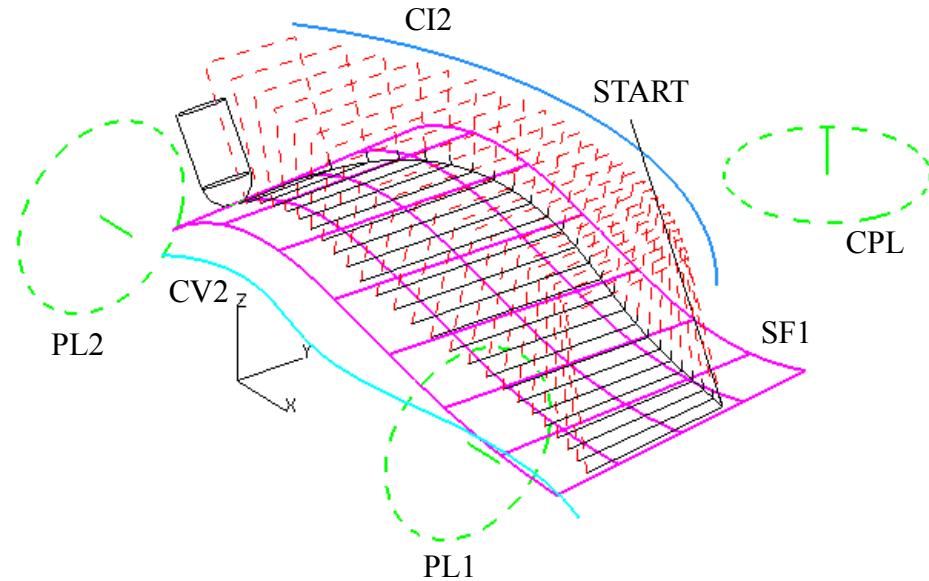
```
RMILL/ON, CV3, ON, CV31, ON, PLX11, ON, PLX12, 1, CPL,      $  
.1, 2, .2, 35, RAPID, 10
```

It should be noted that in RMILL statements, the default modifier is **ON** instead of **TO** as in other **NCL** motion statements.

The figure on next page shows the results of the following RMILL statement:

```
TLAXIS/NORMAL, PS  
RMILL /SF1, TO, CI2, TO, CV2, ON, PL1, ON, PL2, 1, CPL,      $  
.1, 2, .030, 30, RAPID, 10
```

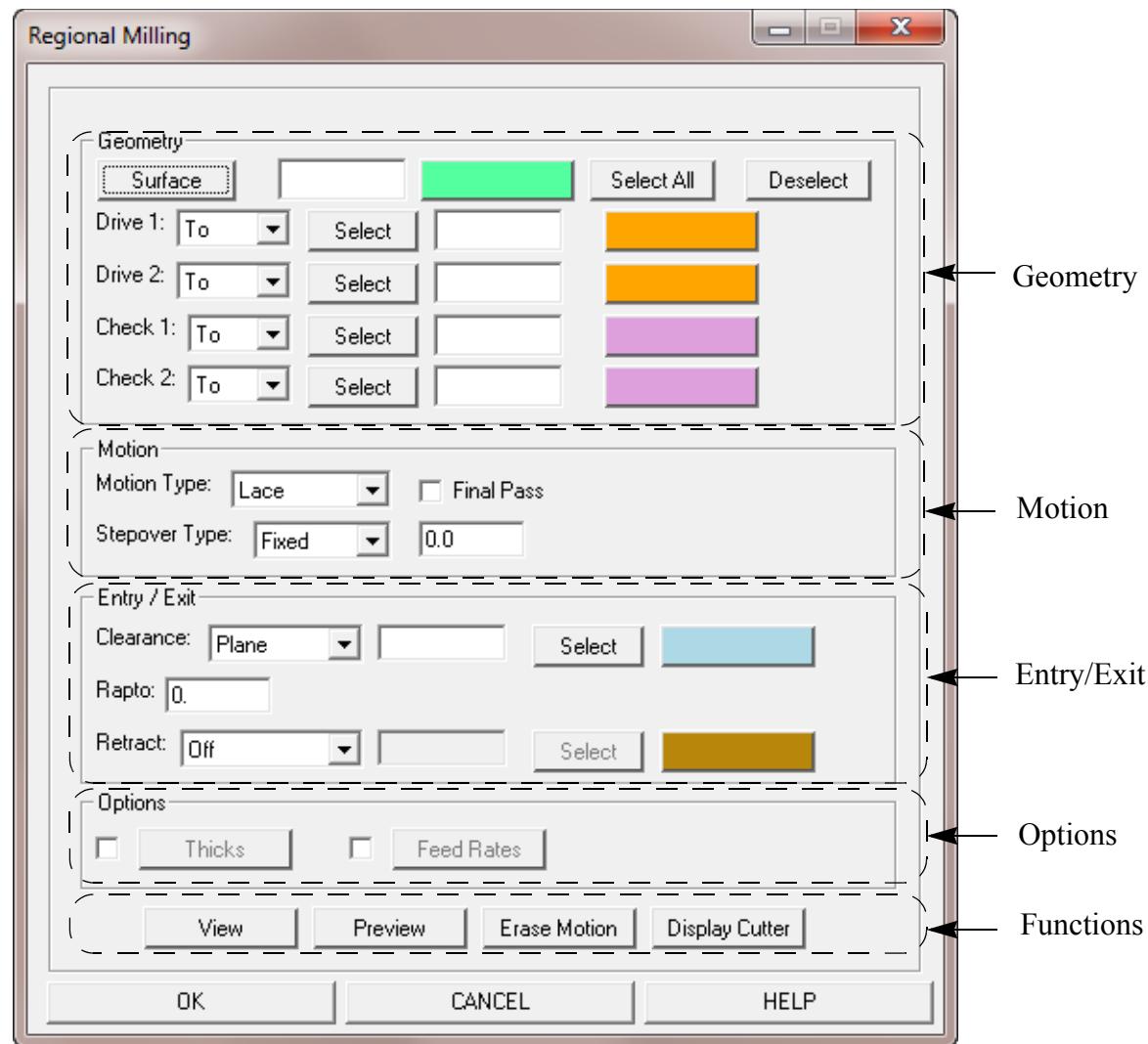
6 AUTOMATIC MOTION ROUTINE STATEMENTS



Following pages show the graphical interactive interface for this RMILL routine and a description of how to use this interface. This interface can be activated by using the following on screen menu icon sequence:



6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form is composed of the following sections: Geometry, Motion, Entry/Exit, Options and Function Buttons.

Geometry

The Geometry section selects the controlling geometry for the regional milling operation.

Surface

Pressing the Surface button allows the user to select the surface or plane to mill.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

The **Color** choice field specifies which color to use to highlight the surface.

Select All

Pressing the Select All button will allow you to select the part surface, drive planes, and the check curves/planes used for the regional milling operation in sequence without redisplaying the form.

It is similar to pressing the Surface, Drive1/Select, Drive 2/Select, Check 1/Select, and Check 2/Select buttons individually.

Deselect

The Deselect button will deselect all entities selected in the Geometry section.

Drive1

This geometry that is the limit of one side of the region to be milled and can be thought of as the drive surface for the motion that is to be generated. It can be either a plane, a planar surface or a line.

‘To’, ‘On’, and ‘Past’ controls the relationship of the cutter to the first drive surface geometry. When the cutter approaches the drive surface geometry it will either stop as it touches it (To), is directly on it (On), or after it is past it (Past).

Select

Press the Select button to select the Drive 1 geometry.

The **Color** choice field specifies which color to use to highlight the Drive 1 geometry.

Drive 2

The geometry that will limit the other side of the region along with the Drive 1 entity. It can be either a plane, a planar surface or a line.

‘To’, ‘On’, and ‘Past’ controls the relationship of the cutter to the second drive surface geometry. When the cutter approaches the drive surface geometry it will either stop as it touches it (To), is directly on it (On), or after it is past it (Past).

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Select

Press the Select button to select the Drive 2 geometry.

The **Color** choice field specifies which color to use to highlight the Drive 2 geometry.

Check 1

The geometry that will limit the cutter motion in the region to be milled and can be thought of as the check surface for the motion that is to be generated. It can be a line, circle, plane, curve, or surface.

‘To’, ‘On’, and ‘Past’ controls the relationship of the cutter to the first check surface geometry. When the cutter approaches the check surface geometry it will either stop as it touches it (To), is directly on it (On), or after it is past it (Past).

Select

Press the Select button to select the Check 1 geometry.

The **Color** choice field specifies which color to use to highlight the Check 1 geometry.

Check 2

The geometry that will limit the other side of the region along with the Check 1 entity. It can be a line, circle, plane, curve, or surface.

‘To’, ‘On’, and ‘Past’ controls the relationship of the cutter to the first check surface geometry. When the cutter approaches the check surface geometry it will either stop as it touches it (To), is directly on it (On), or after it is past it (Past).

Select

Press the Select button to select the Check 2 geometry.

The **Color** choice field specifies which color to use to highlight the Check 2 geometry.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Motion

The Motion section controls the settings used for creating the regional milling motion.

Motion Type

Specifies the type of motion to be generated.

‘Lace’ motion will retract the cutter at the end of each pass and then reposition it at the start of the next pass. All cutting motion will be in a single direction.

‘Scrub’ will create scrub style motion and will not retract the cutter at the end of each pass. Cutting motion will be generated in both directions without retracting the cutter between passes.

Final Pass:

Check this box if a final pass should be taken around the drive and check surfaces after the lace or scrub motion is completed.

Stepover Type

The stepover distance between passes can either be a fixed distance (Fixed) or defined as being the maximum scallop height allowed between passes (Scallop). This value cannot be zero.

Entry/Exit

The Entry/Exit section controls the settings used for entering onto and exiting off of the regional mill surface.

Clearance:

The Clearance fields are only active when the motion type is set to Lace. They define the clearance level to retract the tool to at the end of each pass. The clearance level can either be a plane or a distance.

Select

Press the Select button to select the clearance plane. This button is only active with the Clearance type is set to Plane.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

The color choice field specifies which color to use to highlight the Clearance plane.

Rapto:

The Rapto distance specifies the distance above the regional mill surface to position at when moving down the tool axis from the clearance plane during Lace style motion when starting the next pass. This field is only active when the motion type is set to Lace.

Retract:

The Retract fields define the level to retract the cutter to after the regional milling motion is complete. You can specify a Plane or a Point (Entity), a distance (Distance), or for the cutter not to be retracted at the end of the motion (Off).

If a plane is selected, then the cutter will retract along the tool axis to the level of the plane. If a point is selected, then the cutter will move to that point at the end of the regional milling motion.

Select

Press the Select button to select the retract geometry. This button is only active with the Retract type is set to Entity. A plane or a point can be selected.

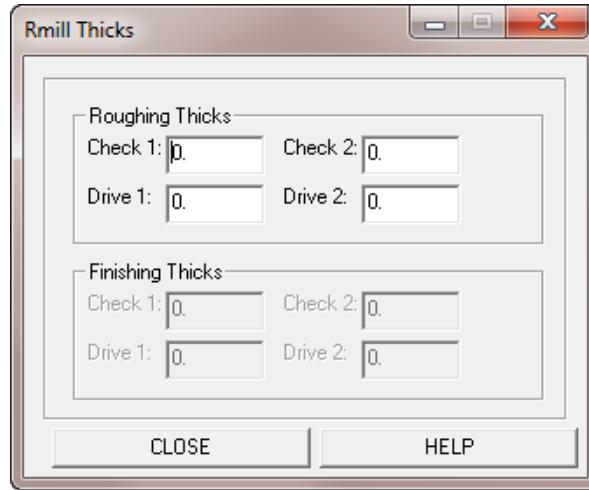
The **Color** choice field specifies which color to use to highlight the Retract entity.

Options

The Options section provides access to the thick and feed rate forms for Regional Milling.

Thicks

Check this box if you would like to specify thick values that will be applied to the Regional Milling drive and check surfaces. Pressing the Thicks button will display the Rmill Thicks form as shown on next page. Unchecking this box will not utilize any of the thick values no matter how they are set.



Rmill Thicks

This form defines the thick values to apply to the drive and check surfaces used in Regional Milling.

Roughing Thicks

Specifies the thick values to apply to the back and forth motion of Regional Milling. A separate value can be set for each drive and check entity.

Finishing Thicks

Specifies the thick values to apply to the finish pass motion of Regional Milling. A separate value can be set for each drive and check entity. These fields will only be active if the [Final Pass](#) box is checked in the main form.

Close

Click this button to accept all the changes (if any) and close the form.

Help

Click this button to open the online help for the Rmill Thicks Form.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Feed Rates

Check this box if you would like to specify feed rates for the various motion types of Regional Milling. Pressing the Feed Rates button will display the Rmill Feed Rates form as shown below. Unchecking this box will use the default feed rate values in effect at this time.



Rmill Feed Rates

This form defines the feed rates to use for Regional Milling. Each feed rate has the following modes that can be programmed.

Current = Uses the currently programmed feed rate.

Value = Allows you to specify an absolute value as the feed rate.

Rapid = Uses RAPID. (Only valid for the Position field).

General:

Defines the feed rate to use during the basic motion of regional milling.

Position:

Defines the feed rate to use when positioning the tool to and along the clearance plane and to the retract plane.

Plunge:

Defines the feed rate to use when the tool is positioned to the rapto distance above the regional milling part surface.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Close

Click this button to accept all the changes (if any) and close the form.

Help

Click this button to open the online help for the Rmill Feed Rates Form.

Action Buttons

View

Takes down the form(s) and enters dynamic viewing.

Preview

Previews the Regional Milling motion. No command is output.

Erase Motion

Erases all motion.

Display Cutter

Displays the current cutter position.

OK:

Click this button to accept the setting, close the form, generate the motion and output all the corresponding source codes to the part program.

Cancel:

Click this button to disregard all the changes (if any) and close the form.

Help:

Click this button to open the online help for the Regional Milling Form.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.31 SMILL

The SMILL statement is used to machine multiple surfaces.

The valid syntax construct for the SMILL statement is shown below.

```
SMILL/sf-list [,TO      ],ds1,ds2 [,BOUND,PART]           $
               PAST    ve1          cv1
               ON
               CONTCT

[,PASS,num-passes           ] [,START,pt1]           $
[HEIGHT,sclop-hgt] [[,]STEP,stp]

[,SCRUB                  ] [,TOLER,tol]           $
COMBIN
LACE   [,ent1[,fr1  ]]
       dis1 RAPID

[,RAPTO,ent2[,fr2  ]] [,RETRCT,ent3[,fr3  ]]           $
       dis2 RAPID          dis3 RAPID

[, OMIT, IN      ]
       NOMORE
```

Where:

- “sf-list” is the standard list of surfaces to machine (sf,sf,..., LAYER,n, sfx(m,thru,n), netsf, etc.).
- “TO” is the default motion edge condition, it keeps the tool within the boundaries of the surfaces. “PAST” causes the tool to go past the boundaries of the surfaces, “ON” positions the tool end point on the surface boundaries, and “CONTCT” causes the contact point of the tool to be positioned on the surface boundaries.
- “ds1,ds2” are the drive surface planes to machine between. They must be parallel planes. If drive surface planes are not specified, then the boundary curve defined by the “BOUND” parameter will be used to automatically calculate the drive surface planes. “ve1” defines the initial direction of the motion. The start point will be used to determine the direction towards the subsequent passes of the motion.
- “BOUND” specifies a closed boundary that encompasses the area to be machined. “PART” specifies that the contour of the surfaces to be

6 AUTOMATIC MOTION ROUTINE STATEMENTS

machined will be used to define the closed boundary and is the default setting. Optionally a user specified closed curve “cv1” can be defined.

Note: When “BOUND” is specified, only the motion for the first boundary encountered will be calculated if there are multiple boundaries for the surface(s) selected. The rest of the boundaries will be ignored.

- “PASS” defines a fixed number of passes to take. “HEIGHT” specifies a scallop height that will control the number of passes and distances between the passes. The default is the current machining tolerance. “STEP” defines a fixed step-over amount between each pass.

Note: For bull nose cutters a combination of “HEIGHT” and “STEP” can be used, where the “STEP” value is used when machining flat surfaces and “HEIGHT” is used when machining sloped surfaces. For bull nose cutters with “HEIGHT” specified an effective tool radius is used to calculate the distance between the passes. When no “STEP” is specified, the bull nose cutter flat diameter will be used as the step-over distance when machining flat surface.

- “START,pt1” specifies a point near where the motion should start. The initial point will be on the drive surface (ds1 or ds2) and at the boundary curve (cv1 or surface boundary) that is closest to this start point. The default start point is the current tool position. The initial direction is determined by what drive surface the motion starts at.
- “SCRUB” performs scrub type motion (back and forth) and is the default setting. “COMBIN” performs scrub type motion, except it skips a pass on the back motion and then cuts the pass that was skipped on the forth motion so that the majority of the material will be cut in one direction, with the opposite direction performing a cleanup type cut. “LACE” performs lace type motion (retracts the tool at the end of each pass and repositions it to the starting side so that all cutting will be done in one direction). An optional retract plane/surface entity (ent1) or distance above the entry point (dis1) and feed rate (fr1 or RAPID) for the retract motion at the end of each pass can be specified.
- “TOLER” defines the machining tolerance to maintain along passes. The active machining tolerance is the default.
- “RAPTO” specifies the initial move into the part at the start of the motion and when re-entering the part for each “LACE” pass. “ent2” specifies a plane or surface to position at prior to entering the part, “dis2” specifies the distance above the part to position to. “fr2” specifies the feed rate to enter the part at or “RAPID” can be used to specify a rapid rate.
- “RETRCT” specifies the position at which to retract to at the end of all motion. “ent3” specifies a plane or surface to retract the tool to, “dis3”

6 AUTOMATIC MOTION ROUTINE STATEMENTS

specifies the distance above the part to retract to. “fr3” specifies the feed rate to retract at or “RAPID” can be used to specify a rapid rate.

- “OMT” allows the inner boundaries of trimmed surfaces to be ignored during motion generation. “IN” specifies that all inner boundaries will be ignored and so any holes in a surface will be considered filled during motion generation. “NOMORE” is the default setting and causes all holes in the surfaces to be respected during motion generation.

NOTE:

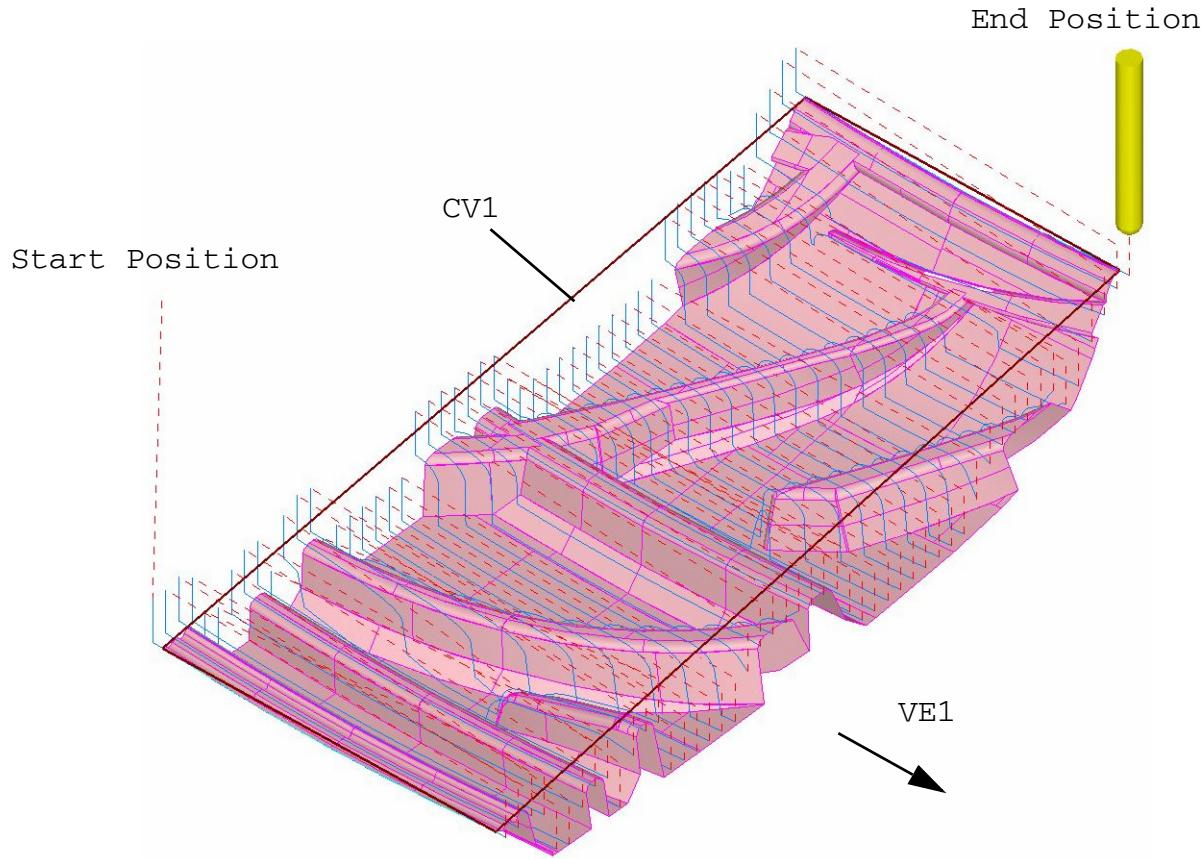
- Not all types of cutter and taxis modes can be used with the SMILL statement. Only a ball-end cutter (i.e. the corner radius is equal to half the diameter) or a flat-nosed cutter (i.e. the corner radius is less than half the diameter) can be specified with the SMILL command. The tool axis must remain constant.
- The part surface thick from the THICK/ps command will be applied to all surfaces so that the tool remains this distance away from each surface.

Example:

```
CUTTER/0.625,0.03125,1  
TLAXIS/0,0,1  
DRAFT/MODIFY=surface_list,LAYER=200  
SMILL /LAYER=200,PAST,VE1,BOUND,CV1,PASS,50, $  
      LACE,0.25,RAPID,RAPTO,0.25,RAPID,      $  
      RETRCT,0.25,RAPID
```

The figure on next page shows the results of the above SMILL statement:

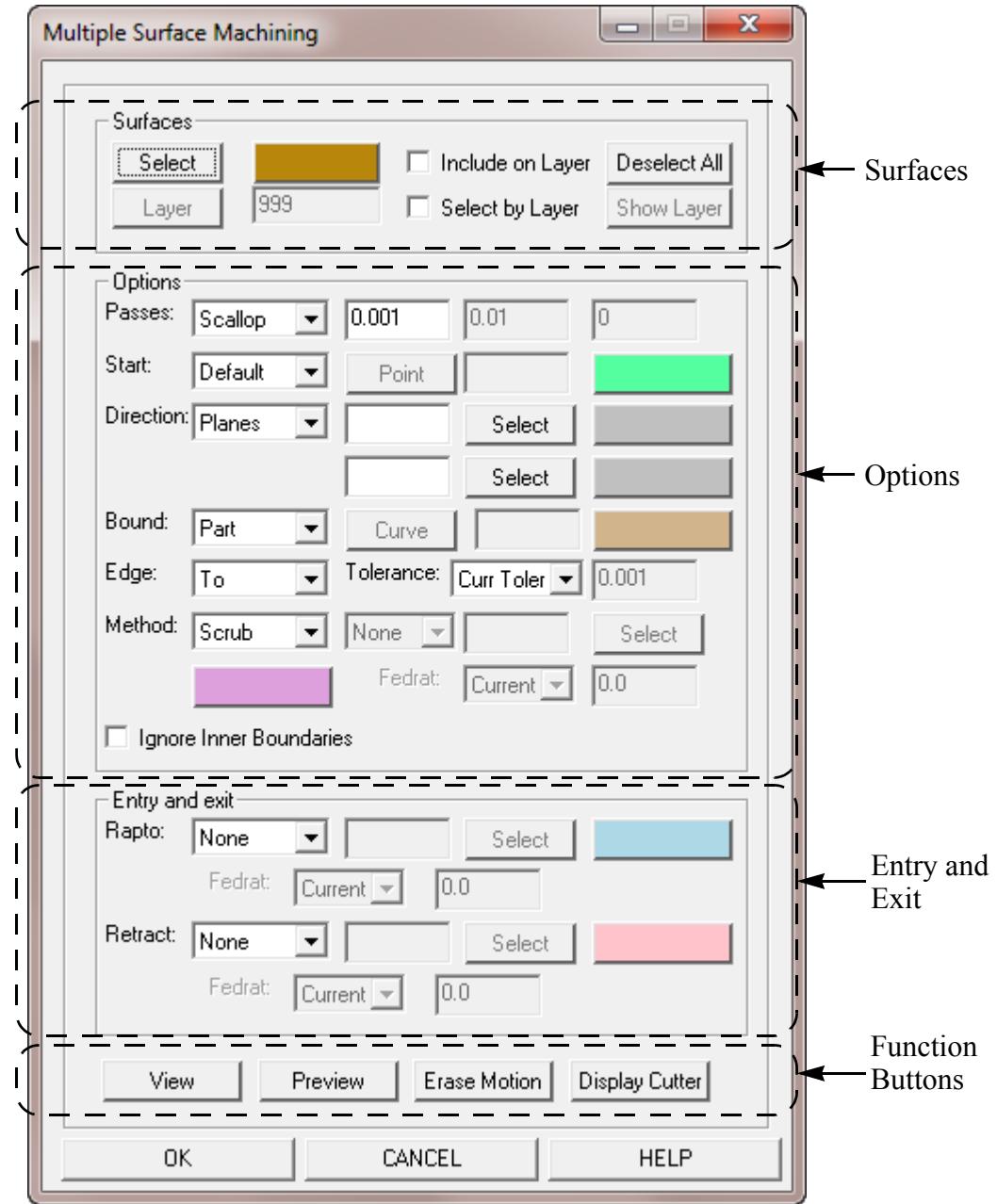
6 AUTOMATIC MOTION ROUTINE STATEMENTS



Following pages show the graphical interactive interface for this SMILL routine and a description of how to use this interface. This interface can be activated by using the following on screen menu icon sequence:



6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form composed of the following sections: Surfaces, Options, Entry and Exit, and Function Buttons.

Surfaces:

This section selects the surface entities which will be considered for the SMILL routine.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Select:

Used to select the surfaces to machine or put on the layer.

Color:

Open the Color Form to choose a color to highlight the selected surfaces or layer surfaces.

Include on Layer:

If checked, the selected surfaces will be put on the specified layer by issuing the appropriate "DRAFT/MODIFY=sf2,sf3,LAYER=4" command. This command will be created prior to the SMILL command. The SMILL command will then reference the layer number rather than the actual surfaces selected.

Deselect All

Deselects all of the currently selected surfaces.

Layer

Used to select a layer number from a list of existing layers. This field is active only when the Select by Layer box is checked. The text field contains the layer number from which all of the surfaces on the layer will be projected, including any surfaces selected in this form.

Select by Layer

If checked, the surfaces on the specified layer will be included in the SMILL command.

Show Layer

Highlights the surfaces currently residing on the selected layer.

Options:

This section composed of six items: Passes, Start, Direction, Bound, Edge and Method.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Passes:

Toggle between Scallop, Stepover, Both and Passes.

- “Scallop” causes **NCL** to calculate the number of passes based on the scallop height specified in the next field. Each pass can be at a varying distance from the previous pass based on the curvature of the surface(s).
- “Stepover” specifies the pass stepover distance. The number of passes will be based on this stepover distance.
- “Both” specifies that the stepover distance is used for the flat area passes and scallop height is used for slope area passes when a bull nose tool is defined.
- “Passes” specifies the number of passes to generate across the surface(s).

Start:

Toggle between Default and Point.

- “Default” causes the motion start nearest to the last tool end point.
- “Point” will start the motion nearest to the selected point.

Point

Used to select the point to start the motion at.

Color

Open the Color Form to choose a color to highlight the selected point.

Direction:

Toggle between Planes and Vector

- “Planes” is used to machine the surfaces between the two specified drive planes. The two drive planes must be parallel.
- “Vector” defines the initial direction of the motion.

Select

The Select buttons allow you to pick the drive surface planes or initial direction vector based on “Direction” field.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Color

Open the Color Form to choose the color(s) to highlight the driver surface planes or initial vector.

Bound:

The motion can be bounded by the outer boundaries of the surfaces to be machined (Part) or a closed curve can be selected to bound the motion (Curve).

Curve

Press this button to select the closed boundary curve to use to bound the motion.

Color

Open the Color Form to choose the color to highlight the closed boundary curve.

Edge:

The edge modifier specifies how the tool is applied to the boundaries of the surfaces or user specified boundary curve.

- “To” stays within the boundary curve.
- “Past” positions the tool past the boundary curve.
- “On” positions the tool end point on the boundary curve.
- “Contact” will position tool so that its surface contact point is on the boundary curve.

Tolerance:

- “Curr Toler” uses the default machining tolerance as specified by the TOLER command for surface machining.
- “Tolerance” allows you to enter a tolerance value to use to calculate the motion for this surface machining.

Method:

The cutting method can be specified as “Scrub”, “Combin”, or “Lace”.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- “Scrub” performs scrub type motion (back and forth) across the surfaces.
- “Combin” also performs scrub type motion, except it skips a pass on the back motion and then cuts the pass that was skipped on the forth motion so that the majority of the material will be cut in one direction, with the opposite direction performing a cleanup type cut.
- “Lace” performs lace type motion (retracts the tool at the end of each pass and repositions it to the starting side so that all cutting will be done in one direction. An optional retract plane/surface and feed rate for the retract motion at the end of each pass can be specified. “None” specifies that the tool will not retract between passes, “Dist” specifies a fixed distance to retract the tool, “Entity” specifies a plane or surface to retract the tool to.

Select

The Select button is used to select the retract plane or surface entity.

Color

Open the Color Form to choose the color to highlight the retract geometry.

Fedrat:

This is only active if LACE with either Dist or Entity is specified. The choices are Current, Rapid and Value.

Ignore Inner Boundaries:

When checked, the inner boundaries of trimmed surfaces will be ignored when the motion is calculated and so all holes will be considered filled during Motion generation.

Entry and Exit:

This section composed of four items: Rapto, Retract, Exit Feedrate. They are described as follows.

Rapto:

- ‘None’ specifies that no special entry move is made prior to the first motion and for each lace pass.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- “Distance” specifies a fixed distance to plunge the tool at each entry,
- ‘Entity’ specifies a plane or surface to move to prior to entering the part.

Select

The Select button is used to select the entry plane or surface entity.

Color

Open the Color Form to choose the color to highlight the entry geometry.

Fedrat:

Specifies the entry move feed rate.

- “Current” uses the currently programmed feed rate.
- “Rapid” makes the move in rapid mode.
- “Value” allows you to enter an entry move feed rate.

Retract

- “None” specifies that the tool will not retract after the last pass.
- “Distance” specifies a fixed distance to retract the tool.
- “Entity” specifies a plane or surface to retract to.

Select

The Select button is used to select the retract plane or surface entity.

Color

Open the Color Form to choose the color to highlight the retract geometry.

Fedrat:

Specifies the retract move feed rate.

- “Current” uses the currently programmed feed rate.
- “Rapid” makes the move in rapid mode.
- “Value” allows you to enter a retract move feed rate.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Function Buttons:

This section composed of four items: View, Preview, Erase Motion and Display Cutter. They are described as follows:

View:

Click to activate dynamic viewing.

Preview:

Click to preview the motion without output the SMILL command or data.

Erase Motion:

Click to erase the displayed motion.

Display Cutter:

Click to display the current cutter.

OK:

Click this button to accept the entries, generate the motion, output the SMILL command and close the Multiple Surface Machining form.

CANCEL:

Click this button to cancel all the changes and close the Multiple Surface Machining form.

HELP:

Click this button to display a brief description of the Multiple Surface Machining form.

6.32

Waterline Roughing

Waterline Roughing is an automatic routine used for roughing a part in multiple levels.

The part to be machined is defined by a list of surfaces in a user specified layer. The surfaces specified do not necessarily to be trimmed surfaces. The only

6 AUTOMATIC MOTION ROUTINE STATEMENTS

requirements are they must share a common edge with the adjacent surfaces and they are not "QUILT" surfaces. The top and bottom of this roughing routine is defined by the planes or revolved surfaces passing the highest and lowest points of the part or defined by the user. The stock can be either a box or a contour (whose shape is determined by the part's projection), possibly expanded by an offset parameter. The stock is interpreted as an extra ribbon surface around the part.

For this routine, just as for the [Advanced Pocket routine](#) or the [VoluMill routine](#), the "UP" direction is defined as the current tool axis.

The algorithm defines the cutting levels (as horizontal planes or revolved surfaces), and the pocket geometry for each level.

The height of each level is defined the same way for the Advanced and VoluMill Pocket routines - using the specified Bottom and Top planes or revolved surfaces together with the Pocket Modal step down parameter (revolved surface is not allowed for VoluMill).

At every level the surfaces (including the Stock) are intersected by the level plane or the revolved surface. The resulting curves are connected into contours, which are used as the pocket geometry for the current level. At each lower level the pocket is compared with the pocket at the immediate level above, and adjusted if necessary, so that pockets at lower levels are completely inside those above.

The [Part Surface Thick](#) Value is not applied to the top and bottom level. The Part Surface Thick Value will only apply to the actual surfaces of the part. However setting the [version flag](#) to version 9.7 or earlier will cause **NCL** to apply the Part Surface Thick Value to the top and bottom level for the Advanced Pocket logic (not VoluMill Pocket logic).

The order of cutting can be specified by:

1. Cutting each individual pockets to depth first before advancing to the next pocket or cutting all the pockets to the same level first within a zone before advancing to the next level.
2. Cutting all the pockets which can be partially, mainly or totally enclosed in a zone first before advancing to the next zone.
3. Cutting or not cutting all the pockets excluded by the zone specification after all the zones are cut.
4. Cutting all the pockets within a zone either randomly, or from largest pocket to smallest pocket, etc.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

The valid syntax of the statement is:

```
POCKET/LAYER=n, [ POCKET, ] [ PLANE , ] $  
    VMPOCK   REVOLV, surf  
    VMP3AX  
  
    STOCK, [offset,] STOCK[,offset] $  
    plane1,[offset,] plane2[,offset]  
    DEPTH,dep,      OFFSET,dist  
    AT,z-level,     AT,z-level  
  
    [, FINISH, LEVEL[ ,ADJUST, UP, dup, DOWN, ddn] ] $  
  
    [, OMIT          ] [, AVOID,sflist2] $  
        IN  
        PART[,expand]  
        FIT[,expand]  
        BOUND,sflist1[,exp1]  
  
    [, ERROR,max-gap] [, START,RANDOM ] $  
        TOLER      NEGX  
                    POSX  
                    NEGY  
                    POSZ  
                    NEGZ  
                    POSZ  
        IN  
        OUT  
        SMALL  
        LARGE  
        NEARPT,point  
  
    [,max-loops] $  
  
    [, ZONE,cv-list, [STOP,] ALL ] [, LEVEL] $  
        LAST   MAIN   DEEP  
        PART  
  
    [, OFF,PART[,offset]] [,STEP,UP,stp]
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Where:

LAYER=n:

- Specifies the name of a layer which holds all the surfaces used to define the part. The surfaces specified in this layer do not necessarily have to be trimmed surface. The only requirements are they must share a common edge with the adjacent surfaces and they are not "QUILT" surfaces.

POCKET:

- Specifies that the [Advanced Pocket](#) logic should be used for the Waterline Roughing. This is the default condition.

VMPOCK/VMP3AX

- VMPOCK specifies the [VoluMill](#) logic be used for the Waterline Roughing.
- VMP3AX specifies to use 3-Axis VoluMill style motion for Waterline Roughing.
- This is optional and requires a separate license.

PLANE:

- Specifies that the Waterline Roughing is processed in the standard mode, i.e. by horizontal level cuts. This is the default condition.

REVOLV, surf:

- Specifies the cutting levels are calculated as offsets of a specified Revolved Surface "surf", and level pockets are cut normal to the Revolved surface. This option is not available for VoluMill logic.
- The direction of the Revolved Surface normal is set along the current tool axis vector.

STOCK:

- Specifies that the plane at the part's lowest level is used as the pocket bottom, optionally offset by "offset".
- If "REVOLV" is specified, this means the lowest revolved surface offset that intersects the layer surfaces is used as the pocket bottom, optionally offset by "offset".

6 AUTOMATIC MOTION ROUTINE STATEMENTS

plane1:

- Specifies the label of a plane or planar surface used as the pocket bottom, optionally offset by "offset".
- This option is not available if "REVOLV" is specified.

DEPTH:

- Specifies that the bottom is defined as the plane offset by "dep" from the top.
- If "REVOLV" is specified, this means the bottom is defined as a revolved surface offset by "dep" from the top revolved surface.

AT:

- Specifies that the bottom is defined as the horizontal plane at "z-level".
- If "REVOLV" is specified, "AT, z0" means the z0-offset of the revolved surface "surf" (with a positive offset along the revolved surface normal).

STOCK:

- Specifies that the plane at the part's highest level is used as the pocket top, optionally offset by "offset".
- If "REVOLV" is specified, this means the highest revolved surface offset that intersects the layer surfaces is used as the pocket top, optionally offset by "offset".

plane2:

- Specifies the label of a plane or planar surface used as the pocket top, optionally offset by "offset".
- This option is not available if "REVOLV" is specified.

OFFSET:

- Specifies that the top is defined as the plane offset by "dist" from the bottom.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- If “REVOLV” is specified, this means the top is defined as a revolved surface offset by “dist” from the top revolved surface.

AT:

- Specifies that the top is defined as the horizontal plane at “z-level”.
- If “REVOLV” is specified, “AT, z1” means the z1-offset of the revolved surface “surf” (with a positive offset along the revolved surface normal).

FINISH, LEVEL:

- Specifies intermediate machining levels will be internally created to finish horizontal planar surfaces.
- This only applies to planar type waterline roughing motion, not to revolved surface type waterline motion.

ADJUST, UP, dup, DOWN,ddn:

- “UP, dup” specifies when an intermediate machining level (a horizontal surface) is found within the “dup” value above a regular cutting level, the regular level will be moved upward to the intermediate level.
- “DOWN, ddn” specifies when an intermediate machining level (a horizontal surface) is found within “ddn” value below a regular cutting level, the regular level will be moved downward to the intermediate level.

OMIT:

- Means no extra stock surfaces will be used to pocket the part, it is the default.

IN:

- Means no extra stock surfaces will be used to pocket the part, plus the outermost contours will be ignored when creating the pocket geometry at each level.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

PART:

- Means the stock will have the shape of the part's vertical projection, optionally expanded by "expand".
- If "REVOLV" is specified, the external contour is calculated by projecting the layer surfaces onto the Revolved Surface and then the smallest UV-contour is used as the stock contour, optionally expanded by "expand".

FIT:

- Means the stock is a rectangular box around the part, optionally expanded by "expand".
- If "REVOLV" is specified, the external contour is calculated by projecting the layer surfaces onto the Revolved Surface and then the smallest UV-box is used as the stock box, optionally expanded by "offset".

BOUND, slist1[,exp1]:

- Means the surfaces in the list "slist1" will be used to create the perimeter contour at each cutting level. The surfaces in the list will not be considered in the overhanging logic, i.e. no effort will be made to avoid them when cutting is done.
- "slist1" can be specified as:
 - A standard sequence of surface labels, e.g. SF1, SFF(1).
 - A sequence indicated by a THRU clause, e.g. "SF2, THRU, SF5," or "2, THRU, 6".
 - A net surface - it is treated internally as a collection of individual surfaces.
 - A layer number, e.g. LAYER=5.
 - Any combination of above, separated by commas.
- "exp1" specifies the optional expansion distance applied to the stock profile created by the bounding surfaces.

AVOID, slist2:

- Means the surfaces in the list "slist2" will be used exclusively in the overhanging logic. The surfaces in the list will not be considered for pocket geometry calculation.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- “slist2” can be specified as:
 - A standard sequence of surface labels, e.g. SF1, SFF(1).
 - A sequence indicated by a THRU clause, e.g. “SF2, THRU, SF5,” or “2, THRU, 6”.
 - A net surface - it is treated internally as a collection of individual surfaces.
 - A layer number, e.g. LAYER=5.
 - Any combination of above, separated by commas.

ERROR:

- Parameter used by the algorithm when it connects separate surface/level plane intersections into a composite curve used as pocket geometry. The word "TOLER" may be specified instead of a scalar parameter. If not specified, the default value is four times the current tolerance.

START:

- Defines the pocketing order within each zone for Waterline Roughing. This defines the secondary order while “ZONE” defines the primary order.

RANDOM:

- Means no special order and this is the default.

POSX:

- Means start from the negative X to the positive X direction.

NEGX:

- Means start from the positive X to the negative X direction.

POSY:

- Means start from the negative Y to the positive Y direction.

NEYG:

- Means start from the positive Y to the negative Y direction.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

POSZ:

- Means start from the negative Z to the positive Z direction.

NEGZ:

- Means start from the positive Z to the negative Z direction.

IN:

- Means start from the pocket nearest to the center.

OUT:

- Means start from the pocket furthest away from the center.

SMALL:

- Means start from the smallest to the largest.

LARGE:

- Means start from the largest to the smallest.

NEARPT,point:

- Means start from the pocket closest to the specified "point".

max-loops:

- Used exactly as in the [Advanced Pocket](#) command, specifies the maximum number of loops of the pocketing motion for each level. If not specified, the algorithm will use as many loops as necessary.

ZONE:

- Define the pocketing order for Waterline Roughing. This defines the primary order while "[START](#)" defines the secondary order.

cv-list:

- An ordered list of any 3-D closed curves to define the zone cutting priority for Waterline Roughing. Each effective zone is

6 AUTOMATIC MOTION ROUTINE STATEMENTS

defined by projecting the corresponding curve normal to the cutting plane. The priority in the list is from left to right, i.e. the zone specified by the first curve will be cut first and the last one on the list will be cut last.

STOP:

- Specifies any pockets not in any of the listed zone will not be cut. This is the default.

LAST:

- Specifies any pockets not in any of the listed zone will be cut after the last zone is done and the order of cutting is defined by “[START](#)”.

ALL:

- Specifies the pocket perimeter must be completely inside the zone curve.

MAIN:

- Specifies the pocket perimeter must be mainly inside the zone curve, i.e. most of the pocket area must be inside the zone.

PART:

- Specifies the pocket perimeter must be partially inside the zone curve, i.e. some of the pocket area must be inside the zone.

LEVEL / DEEP:

- “LEVEL” specifies all the pockets in the specified zone will be cut to the same level first before advancing to the next level and is the default setting. “DEEP” specifies each individual pocket will cut to depth prior to advancing to the next pocket. If the active pocket branches out into multiple pockets, then these pockets will be cut individually to depth prior to advancing.

OFF, PART[,offset]:

- This applies only when the perimeter is specified by either “PART” or “FIT”, i.e. calculated by **NCL**. If specified, for each entry move the algorithm tries to find a near point on the stock from which the

6 AUTOMATIC MOTION ROUTINE STATEMENTS

tool can enter without gouging the pocket islands. If such stock point is found, the tool is repositioned there at the cutting level, and then moved (at the entry feedrate) horizontally to the first cutting contour point. If such point cannot be found, the tool enters by current POKMOD entry mode.

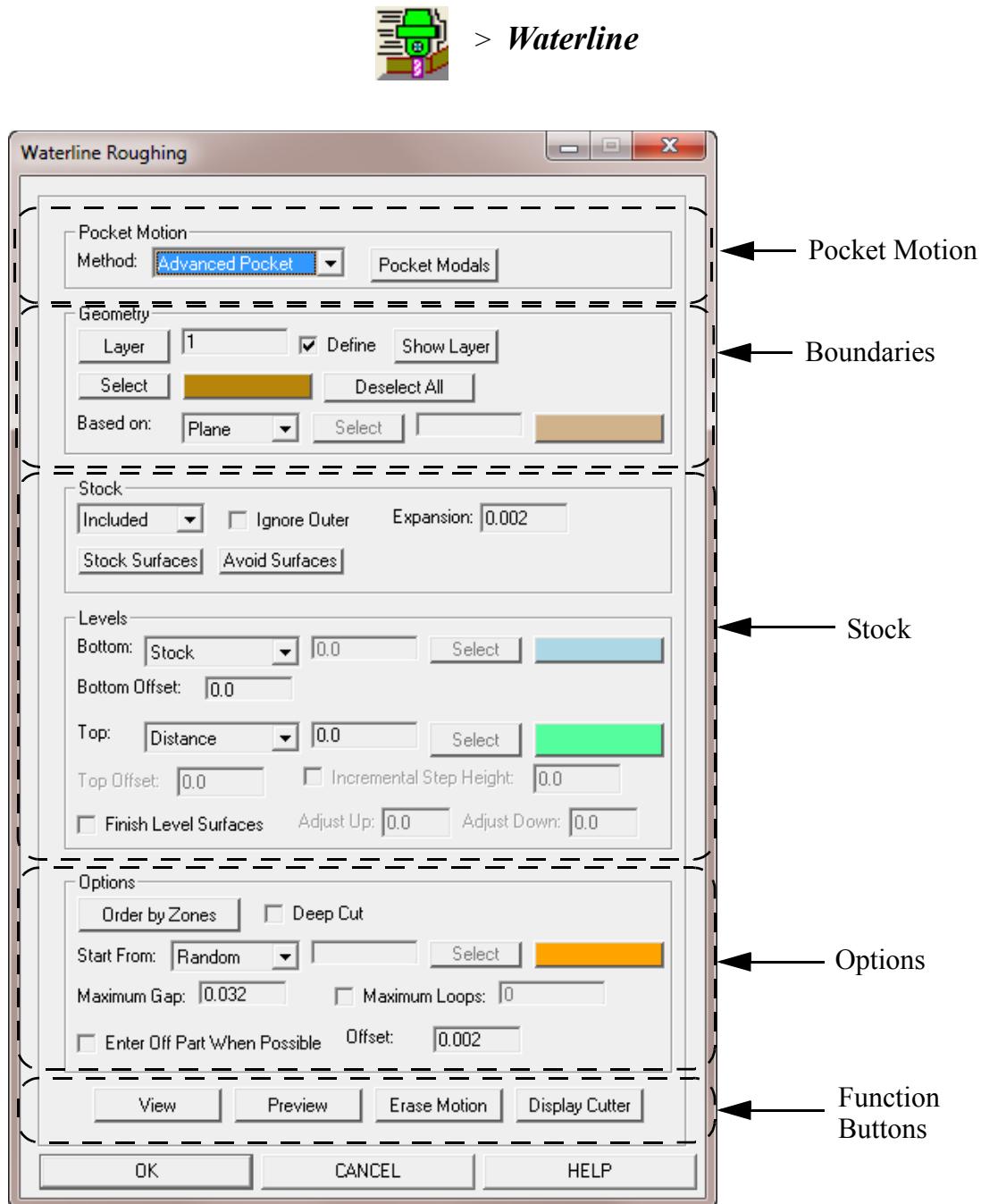
- Specifies the cutter edge is positioned outside of the stock profile by this optional offset distance.
- This option is not available for **VoluMill** logic pocketing motion.

STEP, UP, stp:

- These parameters are only valid with the VMP3AX pocket mode.
- stp specifies the step up height for additional cutting levels made above the current cutting level after the current cutting level depth is finished.

Following pages show the graphical interactive interface for this **POCKET (Waterline Roughing)** routine and a description of how to use this interface. This interface can be activated by using the following on screen menu icon sequence:

6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form composed of the following sections: Boundaries, Stock, Options, and Function Buttons.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Pocket Motion:

This section selects the pocketing motion method for waterline roughing.

Method:

Selects the pocketing algorithm to use to create the pocketing motion. Either [Advanced Pocket](#), [VoluMill 2-Axis](#) or VoluMill 3-Axis can be chosen. This field is only active when VoluMill is licensed.

Pocket Modals

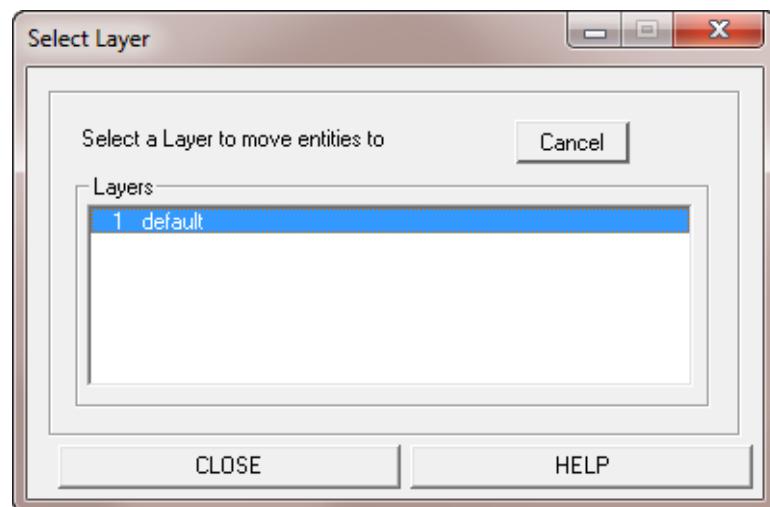
Opens either the [Advanced Pocket Modals form](#) or the [VoluMill Pocket Modals form](#) depending on the pocket method chosen.

Boundaries:

This section selects the surface entities which will be considered for the Waterline Roughing Routine.

Layer

Specifies the number of the layer that contains surfaces defining the part to pocket. All surfaces that reside on this layer will be used in the pocketing algorithm. This can also be done by clicking the LAYER button to open the following form.



6 AUTOMATIC MOTION ROUTINE STATEMENTS

This form displays a list of all defined layers that can be selected for the Waterline Routine. Highlight the appropriate layer and click CLOSE to accept the selection.

- Click CLOSE to accept the selection and close the form.
- Click HELP to open up the online help window for this form.

Define

Determines whether to add surfaces to the selected layer.

Show Layer

Highlights the surfaces residing on the selected layer (including manually selected surfaces).

Select

Used to add surfaces to the selected layer. This button will only be enabled when the Define box is checked. All surfaces selected via this form will be added to any existing entities already residing on the selected layer, and all of these surfaces will be used for pocketing.

Color

Chooses a color to highlight the selected surfaces or the layer surfaces.

Deselect All

Deselects the currently selected surfaces.

Note: The following command is output if surfaces are added to the specified layer

```
DRAFT/MODIFY=surface_list,LAYER=layer_number
```

Base On

Specifies the type of cutting levels. Toggle between “Plane” or “Surface”.

- **Plane** - Default method which defines levels as horizontal planes.
- **Surface** - Defines levels as offsets of a selected surface of revolution. This option is not available for VoluMill logic.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Select

Allows picking a surface of revolution as the Revolved Surface.

Text Field

Contains the current Revolved Surface.

Color

Chooses a color to highlight the Revolved Surface.

Stock:

This section defines the stock boundary for the Waterline Roughing Routine and composed of two items: Stock and Levels.

Stock:

This section specifies the stock side wall boundary.

Included or Calculate

- Included means no extra Stock surfaces will be used to machine the part.
- Calculate will cause the pocketing algorithm to automatically calculate a stock based on the outer surfaces of the part.

Ignore Outer

A choice possible only if the Stock is Included meaning the outermost contours will be ignored when creating the pocket geometry.

Contour or Box

- A choice possible only if the Stock is Calculated.
- Contour means the stock has the shape of the part's outer surfaces vertical projection.
- Box means the Stock is a rectangular box around the part.

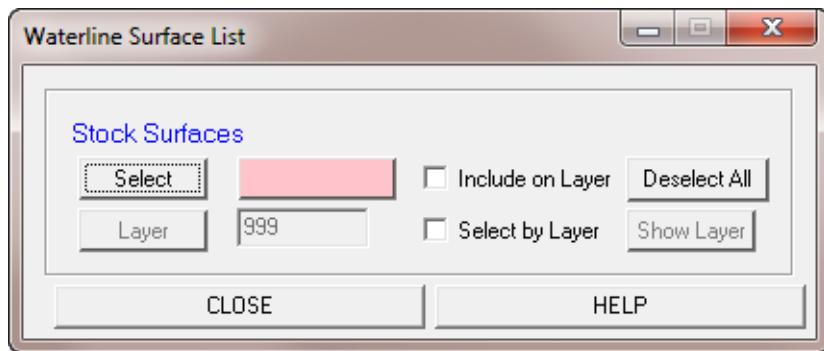
Expansion

The offset parameter for the stock when it is Calculated.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Stock Surfaces

Specifies Stock surfaces which will be used only to create the perimeter contour at each cutting level. Click this button to open the Waterline Surface List form as shown below.



Select

Click this button to pick the surfaces.

Color

Chooses a color to highlight the selected surfaces or layer surfaces.

Include on Layer

If checked, the selected surfaces will be put on the specified layer by issuing the appropriate DRAFT command as shown below:

```
DRAFT/MODIFY=surf-list, LAYER=n
```

This command will be created prior to the POCKET command. The POCKET command will then reference the layer number rather than the actual surfaces selected.

Deselect All

Deselects all of the currently selected surfaces.

Layer

Used to select a layer number from a list of existing layers. This field is active only when the **Select by Layer box** is checked. The

6 AUTOMATIC MOTION ROUTINE STATEMENTS

text field contains the layer number from which all of the surfaces on the layer will be intersected, including any surfaces selected in this form.

Select by Layer

If checked, the specified layer will be included in the surface list.

Show Layer

Highlights the surfaces currently residing on the selected layer.

Close

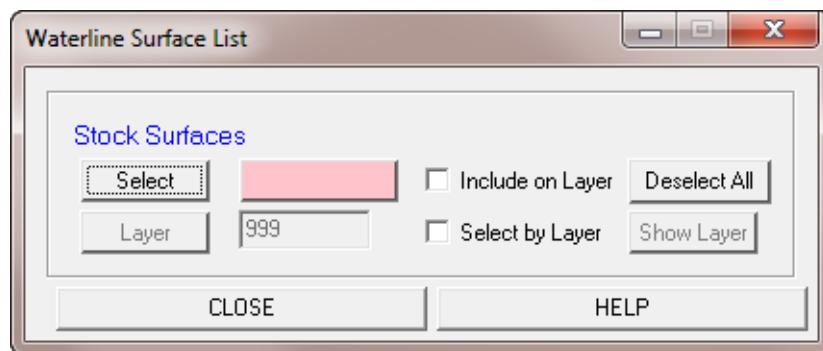
Click this button to accept the selection and close the form.

Help

Click this button to open the on line help for this Waterline Surface List form.

Avoid Surfaces

Specifies the surfaces which will be used only in the overhanging logic, to avoid at each cutting level. Click this button to open the Waterline Surface List form as shown below.



Select

Click this button to pick the surfaces.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Color

Chooses a color to highlight the selected surfaces or layer surfaces.

Include on Layer

If checked, the selected surfaces will be put on the specified layer by issuing the appropriate DRAFT command as shown below:

```
DRAFT/MODIFY=surf-list, LAYER=n
```

This command will be created prior to the POCKET command. The POCKET command will then reference the layer number rather than the actual surfaces selected.

Deselect All

Deselects all of the currently selected surfaces.

Layer

Used to select a layer number from a list of existing layers. This field is active only when the **Select by Layer box** is checked. The text field contains the layer number from which all of the surfaces on the layer will be intersected, including any surfaces selected in this form.

Select by Layer

If checked, the specified layer will be included in the surface list.

Show Layer

Highlights the surfaces currently residing on the selected layer.

Close

Click this button to accept the selection and close the form.

Help

Click this button to open the on line help for this Waterline Surface List form.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Levels:

This section specifies the stock top and bottom boundaries.

Bottom

Toggle between Stock, Plane, Depth from Top, or Z-level. Defines the type of the Pocket Bottom as follows:

- **Stock** - a plane at the part's lowest level.
- **Plane** - an existing plane or planar surface.
- **Depth from Top** - the Bottom is defined relative to the Top.
- **Z-level** - the horizontal plane at a specified height.

Text Field

Contains the current Pocket Bottom: the label if Plane, the height parameter if Depth from Top, or Z-level.

Select

Allows picking a plane or planar surface as Bottom.

Color

Chooses a color to highlight the selected Bottom geometry.

Bottom Offset

Used only with Stock, or Plane Bottom types. The Bottom is defined as the ZLARGE offset of the selected plane by the specified parameter.

Top

Toggle between Stock, Plane, Distance, or Z-level. Defines the type of the Pocket Top as follows:

- **Stock** - a plane at the part's highest level.
- **Plane** - an existing plane or planar surface.
- **Distance** - the Top is defined relative to the Bottom.
- **Z-level** - the horizontal plane at a specified height.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Text Field

Contains the current Pocket Top: the label if Plane, the height parameter if Distance, or Z-level.

Select

Allows picking a plane or planar surface as Top.

Color

Chooses a color to highlight the selected Top geometry.

Top Offset

Used only with Stock, or Plane Top types. The Top is defined as the offset of the selected plane by the specified parameter.

Incremental Step Height

When checked, intermediate machining levels will be internally created by moving the cutting level up by the given height. The boundaries from the previous cutting level will be treated as islands to prevent unnecessary motion. The number of incremental levels will be the maximum number so the incremental cutting level does not pass a previous standard cutting level. This is only valid for VoluMill motion.

Finish Level Surfaces

When checked, intermediate machining levels will be internally created to finish horizontal planar surfaces if such surfaces are found between regular cutting levels.

Adjust Up

If an intermediate machining level is found within this value above the current regular cutting level, the regular level will be moved upward to the intermediate level.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Adjust Down

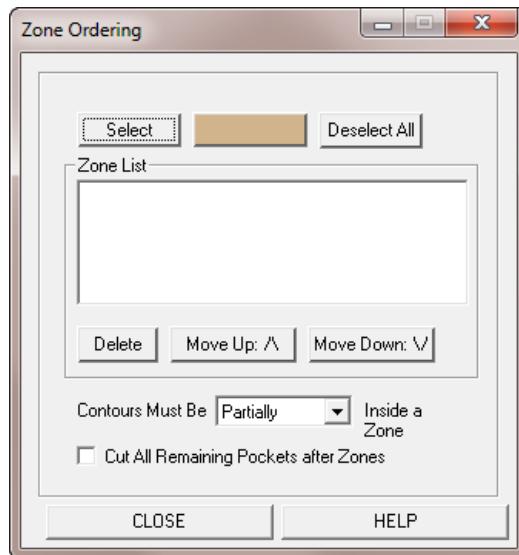
If an intermediate machining level is found within this value below the current regular cutting level, the regular level will be moved downward to the intermediate level.

Options:

To set Various Waterline Roughing Routine options.

Order by Zones

This sets the Zone Ordering of the Waterline Roughing Routine by defining a list of zones and the rule that decides when a pocket belongs to a zone. A valid zone is the horizontal projection of a closed curve or circle. It defines a region on the pocket cutting plane. Click this button to open the “Zone Ordering” form as shown below.



Select

Used to add curves to the list of zones. If a curve already in the list is selected, it will be removed from the list.

Color

Chooses a color to highlight the selected curves.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Deselect All

Deselects the currently selected curves. The list is emptied as a result.

Zone List

This list box contains the current curve list, in order. The list can be altered by selecting lines and using the buttons 'Delete', 'Move Up', and 'Move Down'.

Delete

Removes the highlighted curve from the list.

Move Up

Moves the highlighted curve one position up in the zone list.

Move Down

Moves the highlighted curve one position down in the zone list.

Contours Must Be Partially, (Completely, Mostly) Inside a Zone

This choice box defines the rule by which the pocket-in-a-zone decisions are made.

- **Partially** - some part of the perimeter contour must be inside the zone curve.
- **Completely** - the perimeter must be completely inside the curve.
- **Mostly** - most of the perimeter area must be inside the curve.

Cut All Remaining Pockets After Zones

If checked, the pockets not in any of the listed zones will be cut after the last zone is done. If not checked, the pockets not in zones will not be cut.

Close

Click this button to accept the selection and close the form.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Help

Click this button to open the on line help for this Zone Ordering form.

Deep Cut

If selected, the algorithm will cut each pocket as deep down as possible before cutting the next pocket at the same level.

Start From

Defines the order of the pocketing as follows:

- **Random** - No particular order
- **Neg X** - Order the pockets along the X-axis
- **Pos X** - Reverse order along the X-axis
- **Neg Y** - Order the pockets along the Y-axis
- **Pos Y** - Reverse order along the Y-axis
- **Neg Z** - Order the pockets along the Z-axis
- **Pos Z** - Reverse order along the Z-axis
- **Nearpt** - Do the pockets closer to the near point first
- **Inside** - Do the pockets closer to the center first
- **Outside** - Do the pockets farther from the center first
- **Smallest** - Do the smallest pockets first
- **Largest** - Do the largest pockets first

Text Field

Contains the current near point, ignored if the "Start From" choice is not "Nearpt".

Select

Allows picking of a near point.

Color

Chooses a color to highlight the selected near point.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Maximum Gap

The parameter used by the algorithm when it connects separate surface/level plane intersections into a composite curve used as pocket geometry.

Maximum Loop

Chooses whether to use the optional maximum number of loops.

Text Field

Contains the current maximum number of loops - active only if the option above is chosen.

Enter Off Part When Possible

This applies only to the pockets that have the calculated stock as the perimeter. If selected, for each entry move the algorithm tries to find a near point on the stock from which the tool can enter without gouging the pocket islands. If such stock point is found, the tool is repositioned there at the cutting level, and then moved (at the entry feedrate) horizontally to the first cutting contour point. If such point cannot be found, the tool enters by current POKMOD entry mode. This option is not available for [VoluMill](#) logic.

Expansion

This specifies the offset distance at which the edge of the cutter will be positioned outside of the part.

Function Buttons:

This section composed of three items: View, Preview, Erase Motion and Display Cutter. They are described as follows:

View:

Takes down the form and enters dynamic viewing.

Preview:

Previews the Waterline Roughing motion. No command is output.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Erase Motion:

Erases all motion.

Display Cutter:

Displays the current cutter position.

OK:

Click this button to accept the setting, close the form, generate the motion and output all the corresponding source codes to the part program.

Cancel:

Click this button to disregard all the changes (if any) and close the form.

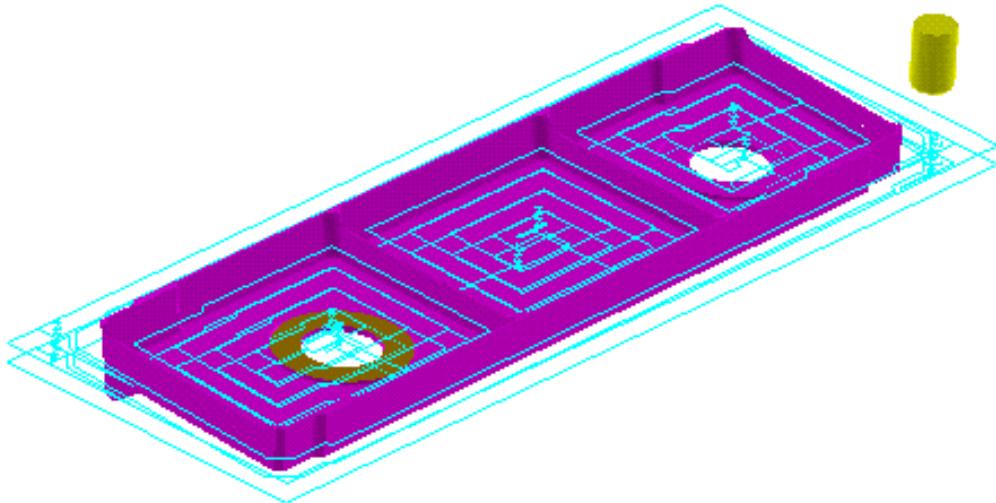
Help:

Click this button to open the online help for this Waterline Roughing Routine Form.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.32.1 Waterline Roughing Examples

6.32.1.1 Example Using The FIT (Calculated Box) Stock Specification

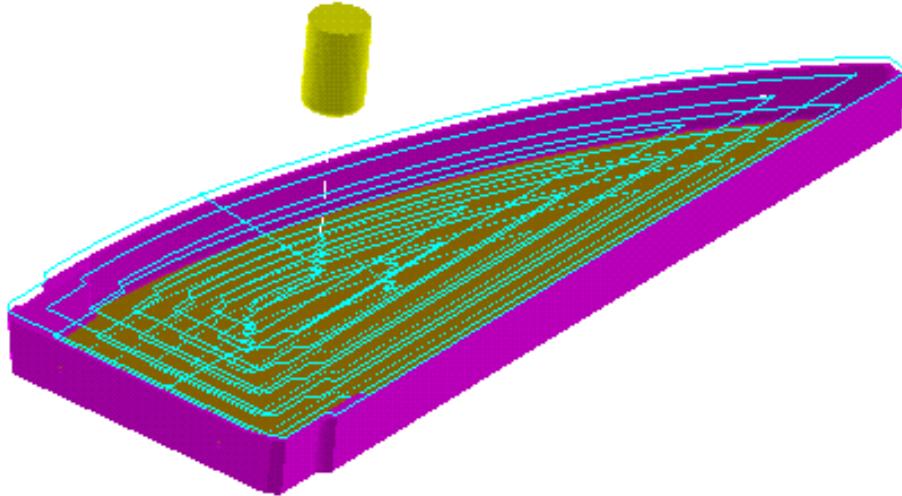


```
CUTTER/1,.12,1.5
OBTAIN/CU,DIA
DRAFT /MODIFY=SF,LAYER=1
POKMOD/-20,RAMP,AVOID,DIA*.25,RETRCT,ON,1,.5,      $
      DEPTH,.1,CCLW,OUT,UP,SHARP,0,0,FD,                 $
      RAPID,RAPID,-.5,-.5,-1
POCKET/LAYER=1,SF89,0,STOCK,0,FIT,1.5,ERROR,        $
      .032,START,IN
```

- The DRAFT command puts all surfaces on layer 1.
- The step down distance is .5.
- The bottom surface is SF89 (colored brown).
- The top surface, specified by STOCK, is at the highest point found on layer 1.
- FIT specifies to center the part into a box that is offset from the part outer extremes by 1.5. Using an offset value that is smaller than the tool diameter would prevent the tool from cutting between the box stock and the outside profile of the part.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.32.1.2 Example Using The PART (Calculated Contour) Stock Specification

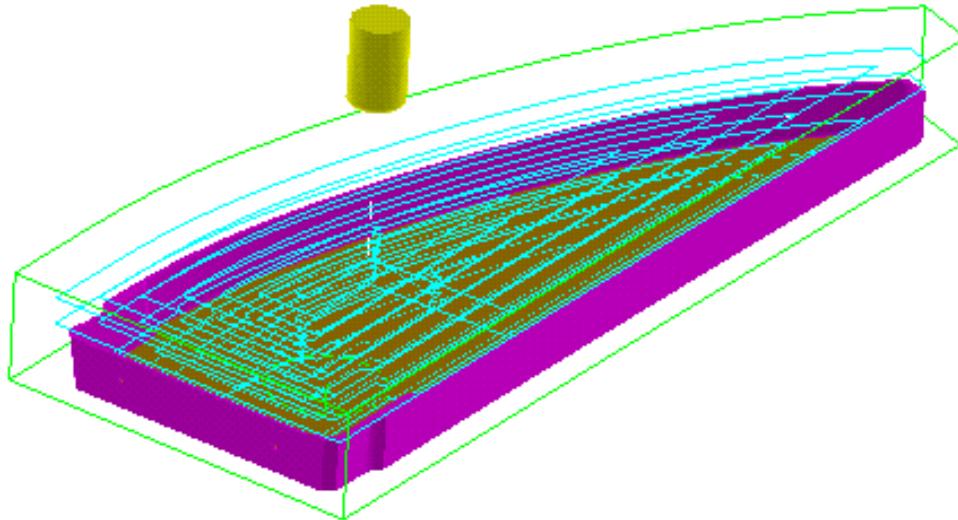


```
CUTTER/1,.12,1.5
OBTAIN/CU,DIA
DRAFT /MODIFY=SF,LAYER=1
POKMOD/-20,RAMP,AVOID,DIA*.25,RETRCT,ON,1,.5,      $
      DEPTH,.1,CCLW,OUT,UP,SHARP,0,0,FD,                 $
      RAPID,RAPID,-.5,-.5,-1
POCKET/LAYER=1,SF7,0,STOCK,.55,PART,DIA/2,          $
      ERROR,.032
```

- The DRAFT command puts all surfaces on layer 1.
- The step down distance is .5.
- The bottom surface is SF7.
- The top surface, specified by STOCK, is offset from the highest point found on layer 1 by .55. This ensures that a pocketing level will made at .05 from the highest flange top.
- PART instructs the system to build stock surfaces that follow the contour of the part, as viewed down the current tool axis. The stock surfaces are offset from the part by DIA/2, allowing the cutter to overhang the outer contour by half the cutter diameter.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.32.1.3 Example Using The OMIT (Included) Stock Specification

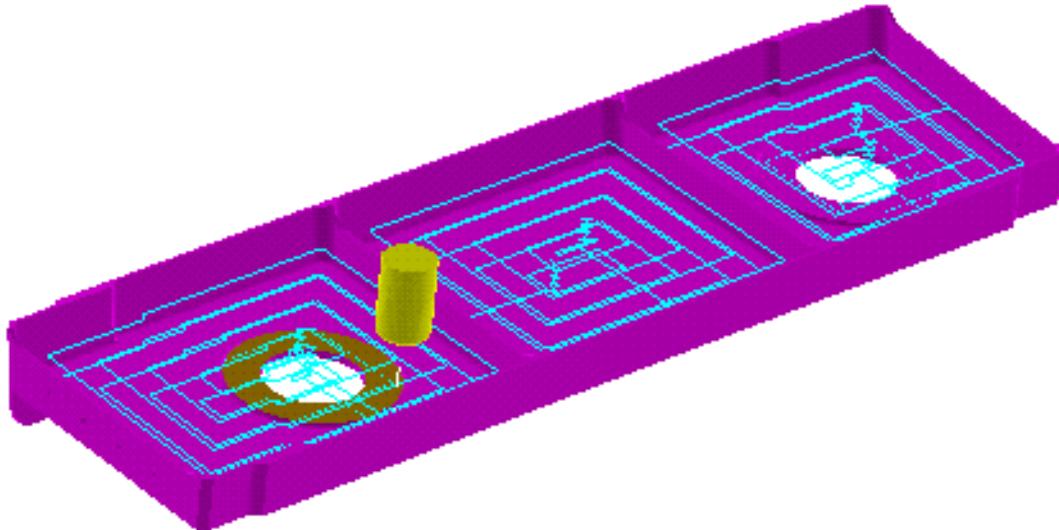


```
CUTTER/1,.12,1.5
OBTAIN/CU,DIA
DRAFT /MODIFY=SF,LAYER=1
POKMOD/-20,RAMP,AVOID,DIA*.25,RETRCT,ON,1,.5,    $
      DEPTH,.1,CCLW,OUT,UP,SHARP,0,0,FD,           $
      RAPID,RAPID,-.5,-.5,-1
POCKET/LAYER=1,SF7,0.0,STOCK,0,OMIT,ERROR,.0322
```

- The DRAFT command puts all surfaces on layer 1.
- The step down distance is .5.
- The bottom surface is SF7 (colored brown).
- The top surface, specified by STOCK is at the highest point found on layer 1.
- OMIT specifies that the outermost contours be considered the stock geometry. This method results in faster calculation times than PART (Calculated Contour) and gives you more control of the shape and location of the outer contours. The stock surfaces are usually built by defining edge curves on the part profile, offsetting the curves by the desired amount, projecting the curves to the desired bottom and top location, and creating ruled surfaces between the projected curves.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.32.1.4 Example Using The IN (Included, Ignore Outer) Stock Specification

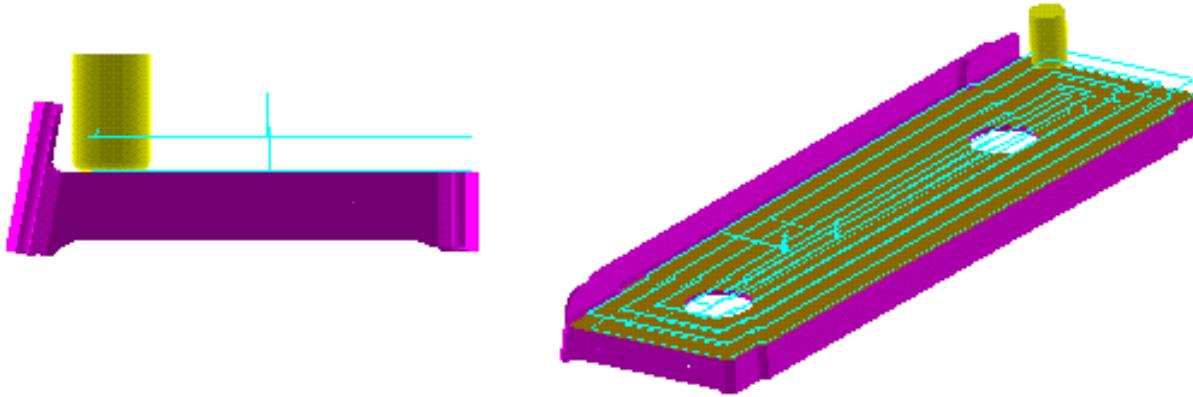


```
CUTTER/1,.12,1.5
OBTAIN/CU,DIA
DRAFT /MODIFY=SF,LAYER=1
POKMOD/-20,RAMP,AVOID,DIA*.25,RETRCT,ON,1,.5,      $
      DEPTH,.1,CCLW,OUT,UP,SHARP,0,0,FD,                 $
      RAPID,RAPID,-.5,-.5,-1
POCKET/LAYER=1,SF89,0,STOCK,0,IN,ERROR,               $
      .032,START,IN
```

- The DRAFT command is used to put all surfaces on layer 1.
- The step down distance is .5.
- The bottom surface is SF89 (colored brown).
- The top surface, specified by STOCK, is at the highest point found on layer 1.
- IN specifies that only the inner contours are to be considered, the outer contours are ignored. This is useful when you want the pocketing motion to stay within the pockets and not machine the flange top surfaces.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.32.1.5 Example Machining Overhanging Surfaces



```
CUTTER/1,.12,1.5
OBTAIN/CU,DIA
DRAFT /MODIFY=SF,LAYER=1
POKMOD/-20,RAMP,AVOID,DIA*.25,RETRCT,ON,1,.5,    $
      DEPTH,.1,CCLW,OUT,UP,SHARP,0,0,FD,           $
      RAPID,RAPID,-.5,-.5,-1
THICK /0,.1,0
POCKET/LAYER=1,SF85,STOCK,PART,DIA/2,ERROR,0.032
```

- The DRAFT command is used to put all surfaces on layer 1.
- The step down distance is .5.
- The bottom surface is SF89 (colored brown).
- The top surface, specified by STOCK, is at the highest point found on layer 1.
- PART instructs the system to build stock surfaces that follow the contour of the part, as viewed down the current tool axis.
- The THICK command causes the cutter to be offset from the part and stock contours by .1
- Notice that the overhanging surfaces are respected.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

6.33 VoluMill Pocketing Motion

This feature generates a set of NC motion instructions to remove material using the VoluMill high performance pocketing algorithm.

The following is the syntax and functionality of the VoluMill Pocket feature.

The VoluMill routine uses two types of **NCL** statements. The VMPMOD statement sets modal values that apply all subsequent VoluMill pocket type statements. The VMPOCK statement causes motion to be generated to clear out material in the pocket.

Note: This feature requires a separate VoluMill license.

6.33.1 VMPMOD

This sets values used by the VoluMill pocket motion generating statement. The valid syntax of the VMPMOD command is shown below.

```
VMPMOD/ [RAMP ,ang      ] [,CLW ] [,CLRSRF,cpln]      $  
          HELIX,ang,rad   CCLW           cdis  
  
          [,RAPTO,rapd] [,RTRCTO,ret1,ret2]      $  
  
          [,DIST ,dep1] [,STEP,stp1][RADIUS,rad1]      $  
              DEPTH  
  
          [,ROUGH          ] [,SIDE          ]      $  
              SMOOTH,sang,srad   SLOT,sdep,sstp  
  
          [,FLUTES,n[,flen[,tlen]]][,LEVEL]  
              DEEP  
  
          [,FEDRAT,f1,f2,f3,f4[,IN  ]]  
              OUT           $  
  
          [,SPINDL,s1,s2,s3][,DWELL,d] [,SMALL,ON ]      $  
              OFF  
  
          [,MINFED,fedmin][,RAPID,xyrap[,zrap]]  
              OFF
```

Where:

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- “**RAMP**” specifies that the entry type should utilize ramp motion. “**HELIX**” specifies that the entry type will be helical motion. “**ang**” specifies the angle of descent for the entry motion. “**rad**” defines the radius of the helix.
- “**CLW**” specifies that the pocket motion will be in a clockwise direction. “**CCLW**” specifies a counter-clockwise direction.
- “**CLRSRF**” defines a clearance plane where the tool can move freely in a horizontal direction at the rapid rate. “**cpln**” defines a physical plane, while “**cdis**” defines a distance above the pocket top plane for the clearance level.
- “**RAPTO,rapd**” defines the distance above the pocket top plane to move at a rapid rate prior to entering the part at the programmed feed rate.
- “**RTRCTO,ret1**” defines the distance above the floor to retract the tool when repositioning within a pocket area. “**ret2**” specifies the distance above the floor to retract to when positioning between pocket areas.
- “**DIST**” specifies that “**dep1**” defines the approximate depth of cut to take between pocket levels. The number of pocket levels will be calculated using this depth value and then a new uniform depth value will be calculated using the number of pocket levels. Each level will cut at this depth, which may be less than or greater than the input depth value.
- “**DEPTH**” specifies that “**dep1**” defines the maximum depth of cut to take between pocket levels. The number of pocket levels will be calculated using this depth value and then a new uniform depth value will be calculated using the number of pocket levels. Each level will cut at this depth, which may be equal to or less than the input depth value. The number of pocket levels machined using the *DEPTH* parameter can be equal to or one greater than the levels calculated using the *DIST* parameter.
- The number of passes to make rather than the depth of cut can be defined by specifying a negative value for “**dep1**”. In this case, it does not matter if *DIST* or *DEPTH* is used. A value of “0” for “**dep1**” will cause half the defined cutter diameter to be used as the depth of cut.
- “**STEP,stp1**” specifies the step over distance to use between successive tool paths. If a value of “0” is specified, then the calculated tool flat diameter will be used.
- “**RADIUS,rad1**” defines the minimum pocket boundary radius that can be machined. If a pocket has a maximum inscribed circle radius less than the tool radius plus this value, it will not be machined. “**rad1**” must be between “0” and “0.9 * *tool radius*” and, if unspecified, the smoothing radius is used.
- “**ROUGH**” specifies that the perimeter and island boundaries will not be attempted to be smoothed. “**SMOOTH**” will attempt to smooth the pocket

6 AUTOMATIC MOTION ROUTINE STATEMENTS

boundaries. “**sang**” specifies the limit at which corner angles at less than this angle will be kept as corners when boundary smoothing is enabled. “**srad**” is the minimum allowable radius in the final tool path when boundary smoothing is enabled. Smaller radii will go deeper into corners and cut narrower channels, but will result in a longer tool path.

- “**SIDE**” causes the motion style to use the side of the tool only. This keeps the tool from making a full cut through the material. “**SLOT**” allows the motion to make a full cut (slot) through the material in order to traverse to separate pocket areas. “**sdep**” specifies the maximum depth of cut the tool is allowed to take when cutting a slot through the material when transitioning between pocket areas. “**sstp**” is the stepover distance to use when opening up a new pocket area by cutting a slot through the material.
- “**FLUTES,n[flen[,tlen]]**” where *n* specifies the number of flutes on the cutting tool, *flen* specifies the length of the cutting section of the cutter and *tlen* specifies the overall length of the cutter.
- “**LEVEL**” causes the pocketing motion to make all required cuts at the same level prior to moving to the next level. “**DEEP**” will cut each individual pocket area to the final level prior to moving to the next pocket area.
- “**FEDRAT**” defines the feed rates to use for the pocketing motion. A value of 0 specifies that the active feed rate should be used. A negative value specifies that a percentage of the active feed rate should be used. These rules apply to all feed rate parameters.
 - “**f1**” specifies the general feed rate for pocketing motion, “**f2**” specifies the feed rate to use when positioning the tool when it is not engaged in the material, “**f3**” specifies the feed rate when entering the part in a ramping motion, and “**f4**” specifies the feed rate to use when cutting a slot in the material.
 - **IN/OUT** specifies whether to reduce the feed rate for entry moves. *IN* specifies the entry feed rate will be output as programmed, meaning that the feed rate will be applied to the center of the tool. *OUT* specifies the entry feed rate will be reduced so the periphery of the tool moves at the entry feed rate instead of the center of the tool. *OUT* is the default condition.
- “**SPINDL**” defines the spindle speeds to use for the pocketing motion. A value of 0 specifies that the active spindle speed should be used. A negative value specifies that a percentage of the active spindle speed should be used. These rules apply to all spindle speed parameters.
 - “**s1**” specifies the general spindle speed for pocketing motion, “**s2**” specifies the spindle speed when entering the part in a ramping motion,

6 AUTOMATIC MOTION ROUTINE STATEMENTS

and “**s3**” specifies the spindle speed to use when cutting a slot in the material.

- “**DWELL,d**” specifies the time in seconds to dwell when the spindle speed is changed.
- “**SMALL,ON/OFF**” signifies that **NCL** will use entry type motion to cut regions too small to pocket using standard motion when *ON* is specified. Regions too small will be omitted if *OFF* is specified. *ON* is the default.
- “**MINFED,fedmin/OFF**” specifies the minimum allowed feed rate when *fedmin* is given. When *OFF* is specified, the minimum feed rate will be ignored.
- “**RAPID,xyrap[,zrap]**” where *xyrap* specifies the rate the tool will move at for moves parallel to the XY-plane. *zrap* specifies the rate the tool will move at for moves parallel to the Z-axis. If *zrap* is not specified, the value specified by *xyrap* will be used for it.

Wherever a "scalar" is allowed in these parameters, a variable scalar name may be given. If it is, the variable will be evaluated at the time the VMPMOD statement is processed and its value used as the modal setting. The name of the variable is NOT passed or used when processing the VoluMill pocketing routine, so if a variable's value is changed between the VMPMOD and the VoluMill pocketing routine, the change will not be reflected in the processing of the VoluMill pocketing routine.

Following shows the graphical interactive interface for this **VMPMOD** routine and a description of how to use this interface. This interface can be activated by using either one of the following on screen menu icon sequences:



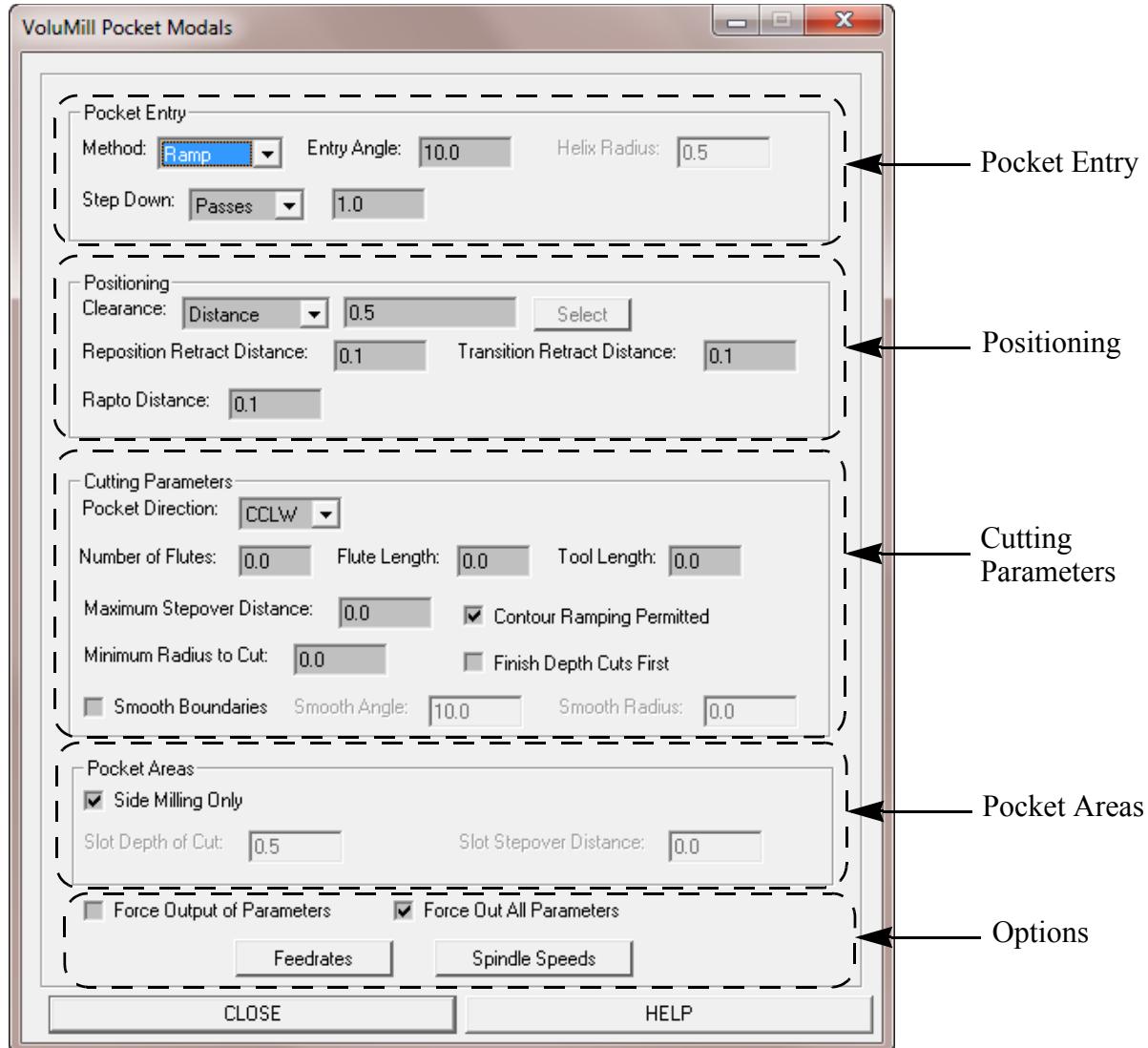
> **VoluMill > VoluMill Modals**

or



> **Waterline > Pocket Modals**

6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form composed of the following sections: Pocket Entry, Positioning, Cutting Parameters, Pocket Areas and Options.

Pocket Entry:

This section specifies how the tool enter the pocket.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Method:

This defines the tool entry method used. Toggle between “Ramp” and “Helix”.

- **Ramp**

The tool will make a series of ramping motions to enter the pocket.

- **Helix**

The tool will make a spiraling move until the current pocketing level is reached.

- **Entry Angle**

Specifies the angle to enter the pocket during the ramp/helix entry.

- **Helix Radius**

Active only with the Helix entry method, defines the radius of the helix.

- **Step Down**

Specifies the method to be used to determine the number of pocketing levels (passes) to machine in a pocket.

Passes

Specifies the actual number of passes to take. The depth of cut will be an equal distance between each pass.

Distance

Specifies an approximate distance between each level. The number of passes will be determined by the total distance divided by this distance. The actual depth of cut may be less than or greater than the input value.

Depth

Specifies the maximum depth of cut for each level. The number of passes is calculated so that an equal depth of cut will be taken for

6 AUTOMATIC MOTION ROUTINE STATEMENTS

each level of the pocketing motion. The depth of cut will actually be this value or smaller. One more pass could be taken when using the Depth parameter as compared to the Distance parameter.

Positioning:

This section specifies the positioning of the cutter before entry of the pocket between levels and exit of the pocket.

Clearance:

Specifies the clearance level when exiting a pocket section and positing to a new section. Toggle between “Distance” and “Plane”.

- Distance**

The distance is based on the top plane of the pocket.

- Plane**

When Plane is specified, then the Select button can be used to interactively pick the plane.

Reposition Retract Distance:

The distance above the pocket floor that the tool will retract to when repositioning within the same pocket.

Transition Retract Distance:

Enter the distance above the pocket floor that the tool will retract to when repositioning between pockets.

Rapto Distance:

The tool will move down the tool axis at the retract feed rate until it is this distance above the top of the pocket or the top plane of the last machined level.

Cutting Parameters:

This section specifies how the pocket motion should be generated.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Pocket Direction:

“CLW” specifies that the pocket motion will be in a clockwise direction.
“CCLW” specifies a counter-clockwise direction.

Number of Flutes:

Enter the number of flutes on the cutting tool.

Flute Length:

Enter the length of the cutting portion of the cutter.

Tool Length:

Enter the total length of the cutter.

Maximum Stepover Distance:

Specifies the maximum stepover amount from one concentric pass to the next. If set to 0, the current effective cutter diameter will be used (dia-cr).

Control Ramping Permitted

When enabled and the area to machine is too small for VoluMill to machine normally, VoluMill will machine the pocket using RAMP/HELIX entry style motion to machine the pocket when possible. The Entry Angle defined above will be used to determine the depth of the cut for each RAMP/HELIX move.

Minimum Radius to Cut:

Enter the minimum pocket boundary radius that can be machined. If a pocket has a maximum inscribed circle radius less than the tool radius plus this value, it will not be machined. A value of between “0.” and “.9 * tool” radius can be entered.

Finish Depth Cuts First:

Check this box if each individual pocket area should be cut to final depth prior to moving to the next pocket area. Leaving this box unchecked will cause all required cuts to be performed at the same level prior to moving to the next depth level.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Smooth Boundaries:

Check this box if an attempt to smooth the perimeter and island boundaries should be made.

Smooth Angle:

Enter the limit at which corner angles at less than this angle will be kept as corners when boundary smoothing is enabled.

Smooth Radius:

Enter the minimum allowable radius in the final toolpath when boundary smoothing is enabled. Smaller radii will go deeper into corners and cut narrower channels, but will result in a longer tool path.

Pocket Areas:

This section specifies how the tool traverse to separate pocket areas.

Side Milling Only:

Check this box if you want the motion style to use the side of the tool only. This keeps the tool from making a full cut through the material. Unchecking this box allows the motion to make a full cut (slot) through the material in order to traverse to separate pocket areas.

Slot Depth of Cut:

Specifies the maximum depth of cut the tool is allowed to take when cutting a slot through the material when transitioning between pocket areas.

Slot Stepover Distance:

Enter the stepover distance to use when opening up a new pocket area by cutting a slot through the material.

Options:

This section composed of four items: Force Output of Modals, Force Out All Parameters, Feedrates and Spindle Speeds. They are described as follows:

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Force Output of Modals:

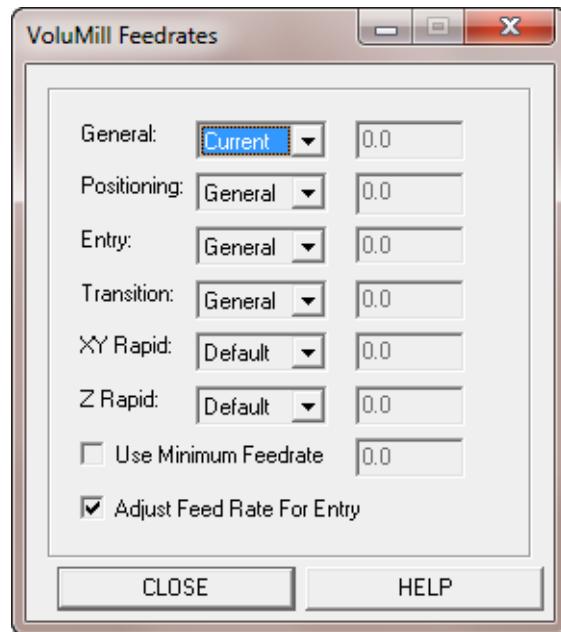
If this box is checked, the VMPMOD statement is output unconditionally. If not checked, the VMPMOD statement is output only if the VoluMill Pocket Modals have been changed and a valid VMPOCK statement is also output by the main form.

Force Out All Parameters:

Check this box if all parameters should be output with the VMPMOD command. If this box is unchecked, then only the parameters that have been changed will be output with the VMPMOD command.

Feedrates:

Specifies the feedrate parameters for the VoluMill pocketing motion. Click this button to display the VoluMill Feedrates form as shown below.



General:

Specifies the general pocketing feedrate. Toggle between “Current” and “Feed”.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Current**

Specifies to use the current primary feedrate as the general pocketing feedrate. This is the default.

- **Feed**

Specifies the general feedrate value.

Positioning:

Specifies the feedrate used for positioning the tool when it is not engaged in the material.

- **General**

Specifies the tool will travel in the general feedrate.

- **Feed**

Specifies the positioning feedrate value.

- **Factor**

Specifies the positioning feedrate as a factor (0 to 1) of the general feedrate. A value of 0 means general feedrate.

Entry:

Specifies the feed rate to use when entering the part in a ramping motion.

- **General**

Specifies the tool will enter the part in the general feedrate.

- **Feed:**

Specifies the entry feedrate value.

- **Factor**

Specifies the entry feedrate as a factor (0 to 1) of the general feedrate. A value of 0 means general feedrate.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Transition:

Specifies the feed rate to use when cutting a slot to enter a new pocket area.

- General**

Specifies the general feedrate for the slot cutting.

- Feed:**

Specifies the feedrate value for slot cutting.

- Factor**

Specifies the transition feedrate as a factor (0 to 1) of the general feedrate. A value of 0 means general feedrate.

XY Rapid:

Specifies the rate at which the machine moves at rapid in the XY plane.

- Default:**

Set XY Rapid rate to the Transition rate.

- Feed:**

Specifies an XY Rapid rate.

Z Rapid:

Specifies the rate at which the machine moves at along the Z-Axis.

- Default:**

Sets the Z Rapid rate to the Transition rate.

- Feed:**

Specifies a Z Rapid rate.

Use Minimum Feedrate:

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Specifies the minimum feed rate allowed.

Adjust Feed Rate For Entry

When enabled the feed rate at the tool center is reduced during material entry so that the periphery of the tool moves at the Plunge Feedrate.

CLOSE:

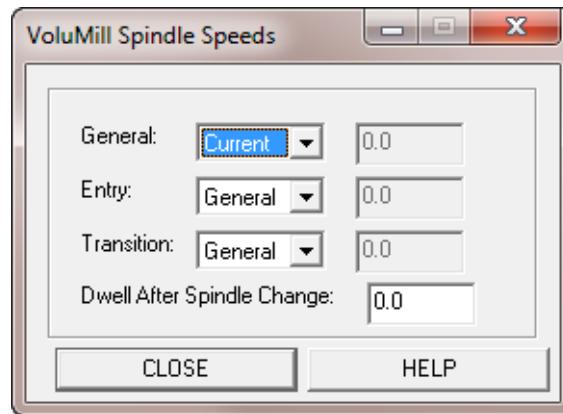
Click this button to accept the changes and close the VoluMill Feedrate form.

HELP:

Click this to bring up the online help of the VoluMill Feedrate form.

Spindle Speeds:

Specifies the spindle speeds for the VoluMill pocketing motion. Click this button to display the VoluMill Spindle Speeds form as shown below.



General:

Specifies the general pocketing spindle speed. Toggle between “Current” and “Feed”.

- Current**

Specifies to use the current spindle speed as the general pocketing spindle speed. This is the default.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

- **Speed**

Specifies the general spindle speed value.

Entry:

Specifies the spindle speed to use when entering the part in a ramping motion.

- **General**

Specifies the tool will enter the part in the general spindle speed.

- **Speed:**

Specifies the entry spindle speed value.

- **Factor**

Specifies the entry spindle speed as a factor (0 to 1) of the general spindle speed. A value of 0 means general spindle speed.

Transition:

Specifies the spindle speed to use when cutting a slot to enter a new pocket area.

- **General**

Specifies the general spindle speed for the slot cutting.

- **Speed:**

Specifies the spindle speed value for slot cutting.

- **Factor**

Specifies the slot cutting feedrate as a factor (0 to 1) of the general spindle speed. A value of 0 means general spindle speed.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Dwell After Spindle Change:

Specifies the amount of time in seconds to dwell after changing the spindle speed.

CLOSE:

Click this button to accept the changes and close the VoluMill Spindle Speeds form.

HELP:

Click this to bring up the online help of the VoluMill Spindle Speeds form.

CLOSE:

Click this button to accept the change and close the VoluMill Pocket Modals form.

HELP:

Click this button to bring up the online help for the VoluMill Pocket Modals form.

6.33.2 VMPOCK

The VMPOCK statement syntax specifies the geometric elements that define the pocket boundary and islands. The current cutter and VOPMOD parameters are used to determine cutting parameters. The valid syntax for the VOMPOCK statement is shown below.

```
VMPOCK/bottom-plane-or-surface,top_plane,          $  
      dis  
      [OFFSET,]peri_geo[,islands]                      $  
      IN  
      [,PS,THICK,pthk]                                $  
      [,DS,THICK,dthk[,ofsthk]]                      $  
      [,OPEN,ind1[,THUR,ind2]]  
      ALL
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Where:

- “**bottom-plane or surface**” defines the bottom surface of the pocket. The specified plane or planar surface must be normal to the current tool axis.
 - “**top_plane**” specifies the plane at the top of the pocket. It must be parallel to the bottom plane. A distance “**dis**” can be specified that defines the distance above the bottom plane for the top plane.
 - “**OFFSET**” is only valid when the same surface is selected as the bottom surface, perimeter geometry, and optionally the island geometry. The optional “**IN**” parameter can be used to specify that the pocketing motion remain inside the perimeter of the bottom surface.
 - “**peri_geo**” defines the perimeter geometry of the pocket and may be one of the following.
 - Composite Curve
 - Closed Curve or Spline
 - Pattern
 - 360 degree Circle
 - The base name of a subscripted array of Points
 - A trimmed surface
- If a trimmed surface is specified as the perimeter, it should be the same planar surface used as the bottom plane.
- “**islands**” specifies the geometry to use as internal boundaries for the pocket motion to avoid. The valid geometry types are the same as for the perimeter geometry. The same rules apply to a trimmed surface, in that it should be the same as specified for the bottom plane and perimeter geometry.
 - “**PS,THICK,pthk**” specifies an optional part surface thick at which to keep the tool off of the bottom plane. If this parameter is not specified, then the active part surface thick will be used.
 - “**DS,THICK,dthk[,ofsthk]**” *dthk* specifies an optional boundary geometry thick at which to keep the tool away from the perimeter and island boundaries. If this parameter is not specified, then the active drive surface thick will be used. *ofsthk* specifies an option thick value for the OFFSET parameter. VoluMill only respects a negative thick of up to approximately 1/4 the diameter of the cutter.
 - “**OPEN,ind1/ALL[,THRU,ind2]]**” specifies open sides of the pocket boundary.

There can be multiple open sides for a single perimeter entity. To specify a range the parameter THRU must be given.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

The following command will use a composite curve to define the pocket boundary with the boundary being offset along multiple components of CV1.

VMPOCK/(PL/0,0,1,0),0.0,CV1,OPEN,1,3,5,THRU,8

The following command will use an array of points to define the pocket boundary with two open sides.

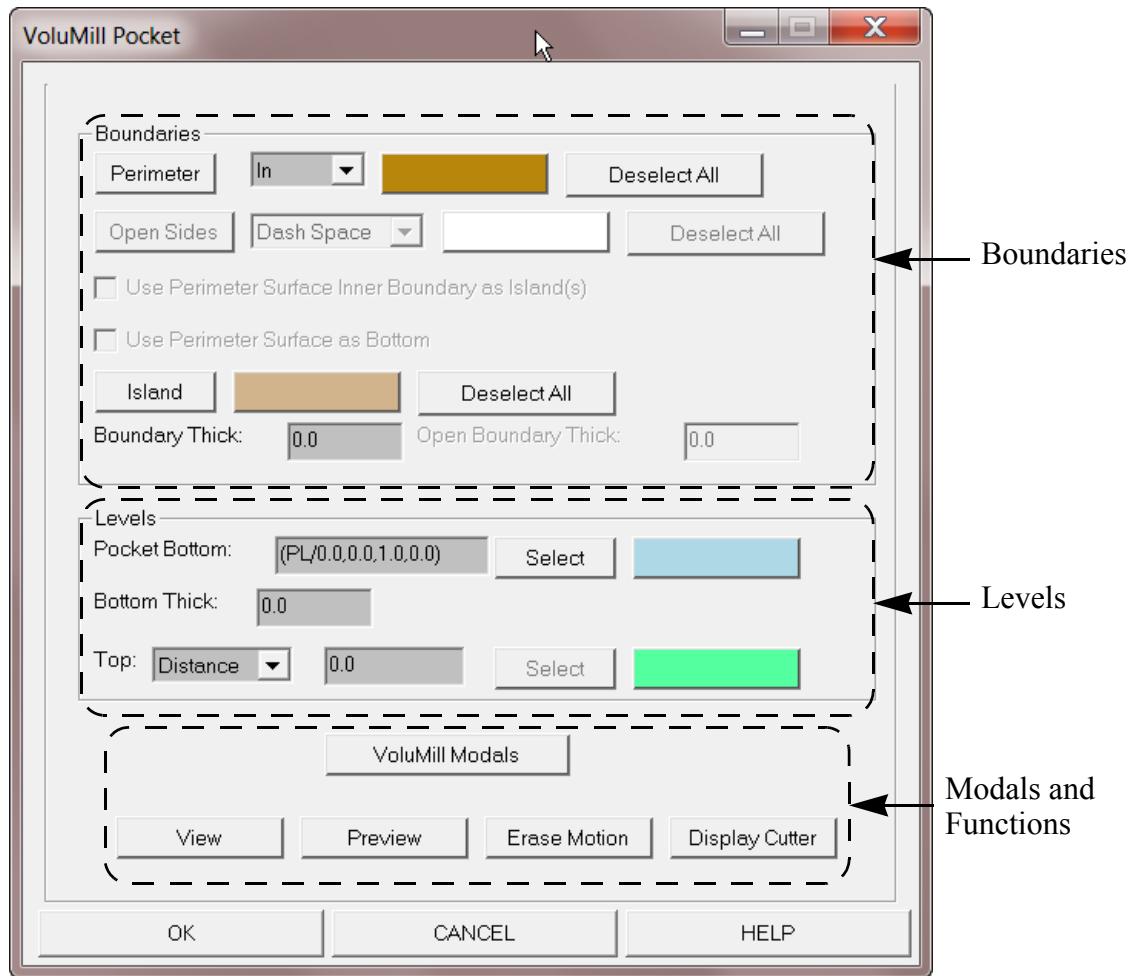
VMPOCK/(PL/0,0,1,0),0.0,PX,OPEN,1,THRU,6,10,THRU,14

Note that to use an array of points, the parameter THRU must be used to specify an open side where a composite curve can use a single component.

Following pages show the graphical interactive interface for this **VMPOCK** routine and a description of how to use this interface. This interface can be activated by using the following on screen menu icon sequence:



6 AUTOMATIC MOTION ROUTINE STATEMENTS



This form composed of the following sections: Boundaries, Levels, Modals and Functions.

Boundaries:

This section specifies the outside and inside boundaries of pocket(s).

Parameters:

Takes down the form, brings up the SELECT menu. Allows selecting a list of entities - each perimeter entity accounts for a separate VMPOCK command. Note that having selected more than one perimeter prevents you from selecting island geometry.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Any curve can be chosen as a perimeter. It should be closed however, otherwise the pocket algorithm will replace the end point to close it.

A surface can be selected as a perimeter. The algorithm will internally use a composite curve made of the surface outer boundary.

Direction-mod:

The options for the pocket perimeter allow you to specify a closed pocket (IN) or a pocket boundary where **NCL** will determine the open/closed side (OFFSET).

Color:

Chooses a color to highlight the selected perimeter geometry.

Deselect All:

Deselects the currently selected perimeter geometry entities.

Open Sides:

Allows selecting of open boundary sides. Open sides can be defined for composite curves and subscripted point arrays. To select the open side:

- Select the first point or component defining the open side
- Select the direction the open side will follow along the perimeter geometry
- Select the last point or component defining the open side

The open side will be displayed using the line style and color defined.

To define additional open sides for a single entity, you can press the Open Sides button again for each open side.

Line Style:

Display style for open sides of a composite curve. The line style will be used when displaying the components of the composite curve perimeter geometry that have been labeled as open.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Color:

Chooses a color to highlight the selected open sides of perimeter geometry.

Deselect All:

Deselects the currently selected open sides of perimeter geometry entities.

Use Perimeter Surface Inner Boundary as Island(s):

If the perimeter surface is trimmed and has inner boundaries, a user could choose to use the inner boundaries as pocket islands.

Use Perimeter Surface as Bottom:

A user could choose to use the perimeter surface as the pocket bottom. If chosen so, the Pocket Bottom fields become disabled.

Island:

Takes down the form, brings up the SELECT menu. Allows selecting a list of entities. Any closed curve can be specified as island geometry. Also, a trimmed surface with inner boundary(ies) could be chosen, in which case the algorithm will use the inner boundary curves.

Color:

Chooses a color to highlight the selected island geometry.

Deselect All:

Deselect the currently selected island geometry entities.

Boundary Thick:

Specify an optional drive surface thick for the pocket motion. This is corresponding to the “DS,THICK,dth” parameters in the VMPOCK command. VoluMill only respects a negative thick of up to approximately 1/4 the diameter of the cutter.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Open Boundary Thick:

Used for the pocket geometry (DS) thick parameter for open boundary sides when the Offset perimeter modifier is selected.

Levels:

Pocket Bottom:

Contains the current pocket bottom, unless a pocket perimeter surface is used as pocket bottom (see above).

Select:

Allows picking a plane or planar surface as pocket bottom.

Color:

Chooses a color to highlight the selected pocket bottom.

Bottom Thick:

Specify an optional part surface thick for the pocket motion. This is corresponding to the “PS,THICK,dth” parameters in the VMPOCK command.

Pocket Top:

Choose between using a plane or a distance for pocket top. The text field contains the current pocket top plane or distance.

Select:

Allows picking a plane or planar surface as the pocket top - active only when Plane is chosen.

Color:

Chooses a color to highlight the selected pocket top.

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Modals and Functions

VoluMill Modals

Opens the [VoluMill Modals form](#).

View:

Takes down the form(s) and enters dynamic viewing.

Preview:

Previews the VoluMill pocket motion. No command is output.

Erase Motion:

Erases all motion.

Display Cutter:

Displays the current cutter position.

OK:

Click this button to accept the setting, close the form, generate the motion and output all the corresponding source codes to the part program.

Cancel:

Click this button to disregard all the changes (if any) and close the form.

Help:

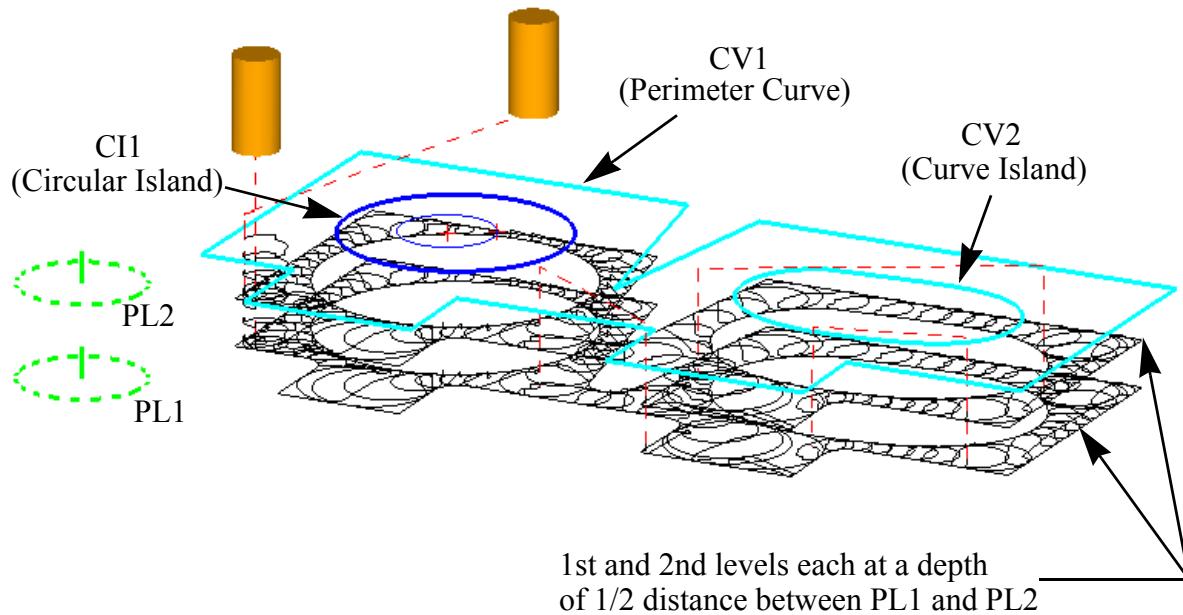
Click this button to open the online help for this VoluMill Pocket form.

Example:

```
CUTTER/0.5,0,1.375
VMPMOD/RAMP,10.0,CLW,CLRSRF,0.5,RAPTO,0.1,      $
          RTRCTO,0.1,0.1,DEPTH,-2,STEP,0.0,           $
          RADIUS,0.0,SIDE,ROUGH,FLUTES,3,LEVEL,        $
          FEDRAT,100,0.75,00.5,0,SPINDL,1000,         $
          0,0,DWELL,0.0
VMPOCK/PL3,PL4,CV10,CI1,CV9
```

6 AUTOMATIC MOTION ROUTINE STATEMENTS

Above statements would machine a pocket with two islands. The bottom of the pocket is at PL1. The top of the pocket is at PL2. The pocket would be cleared at two levels as shown below.



Miscellaneous Motion Statements

This chapter deals with all the miscellaneous motion statements such as COPY, DNTCUT, TOLER, THICK, GOUGCK, etc.

6.32 ARCSLP/ FILLET

This command enables or disables the automatic generation of fillet radii. When the command is active, **NCL** will automatically generate the specified fillet radius at the intersection of successive motion records. Circular motion record is output to the clfile when the created arc is planar along the tool axis. It is not limited to the XY-plane.

6.32.1 ARCSLP/ FILLET, rad, . . .

The valid syntax for the ARCSLP statement is:

```
ARCSLP/FILLET, rad[ , tol] [ , NOWARN] [ , ONCE ]           $  
                      WARN      COMBIN  
  
                      [ , INTERP ]           $  
                      LOCK [ , maxang ]  
  
                      [ , FEDRAT, fedrt, fmax, LEFT , cdia]  
                                         RIGHT
```

Where:

- | | |
|--------|---|
| rad | - Specifies the fillet radius to be generated. A value of "0" will disable this command. |
| tol | - Specifies the tolerance used to calculate the points on the fillet. |
| NOWARN | - Specifies not to output a warning message when a fillet radius cannot be generated, or maximum angular change in tool axis exceeded the allowable value. This is the initial condition. |
| WARN | - Specifies to output a warning message when a fillet radius cannot be generated, or maximum angular change in tool axis exceeded the allowable value. |
| ONCE | - Specifies NCL will only attempt to create the fillet between two consecutive motions. This is the default |

6 MISCELLANEOUS MOTION STATEMENTS

condition. A warning message will be output if a fillet cannot be generated and "WARN" is specified.

- | | |
|--------|---|
| COMBIN | - Specifies if a fillet cannot be built between two consecutive motions, then an attempt will be made to create a fillet between three consecutive motions. |
| INTERP | - Specifies the tool axis will fan from the starting location tool axis to the ending location tool axis while the tool moves along the fillet. This is the default condition. |
| LOCK | - Specifies the tool axis vector will remain the same along the entire fillet motion. The tool axis at the corner of the programmed motion will be used. This corner location is actually replaced by the fillet motion. |
| maxang | - Specifies the maximum angular change in degrees that the tool axis can have as compared to the starting fillet location and ending fillet location when "LOCK" is specified. A warning message will be output if the tool axis deviated by more than this amount and "WARN" is specified. |
| FEDRAT | - Specifies that NCL will calculate the true surface feedrate for the fillet generated motion. The feedrate will be increased for outside fillets and decreased for inside fillets in order that the programmed feedrate be maintained where the cutter touches the work piece. |
| fedrt | - Specifies the feedrate to maintain on the side of the cutter. If not specified, the current programmed feedrate will be used. |
| fmax | - Specifies the maximum feed rate that can be output with fillet motion. If not specified then the maximum feedrate will be 999. |
| LEFT | - Specifies that, for the purposes of calculating the true surface feedrate, the cutter is to the left of the work piece. If not specified, the current programmed tool condition is used. |
| RIGHT | - Specifies that the cutter is to the right of the work piece. If not specified, the current programmed tool condition is used. |

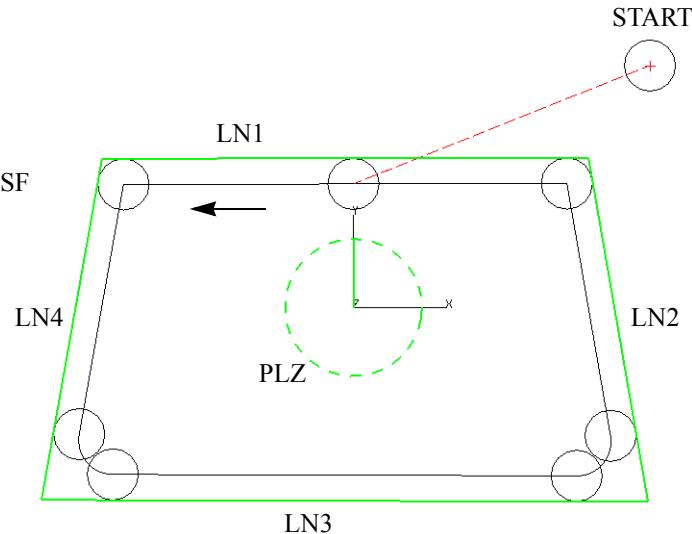
6 MISCELLANEOUS MOTION STATEMENTS

cdia

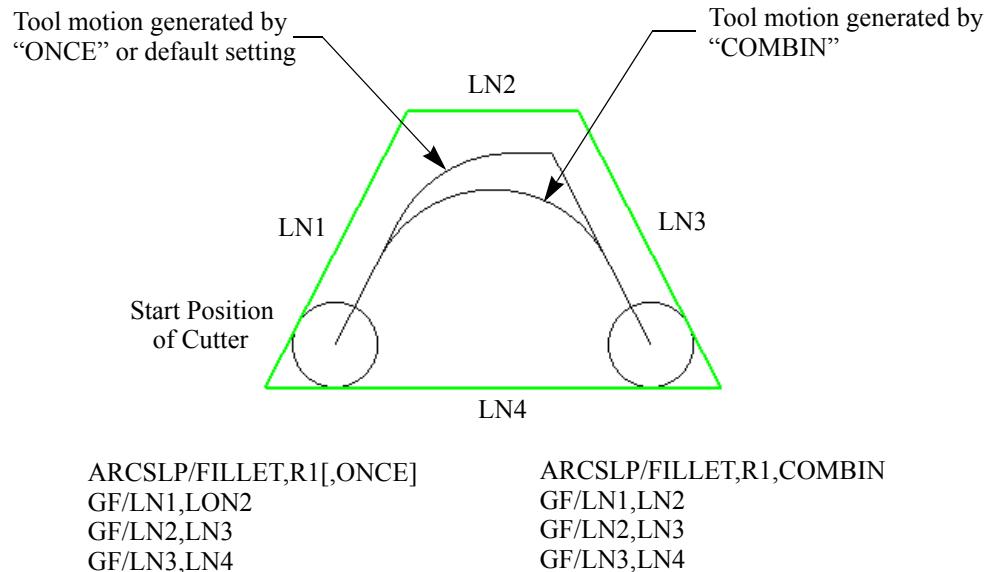
- Specifies the diameter of the cutter for the purpose of calculating the true surface feedrate. If not specified, the current programmed cutter diameter is used.

The figure below illustrates the use of this command:

```
CUTTER/.5,0,1
FROM/START
RAPID
CKSF=LINE/YAXIS
GO/PAST,LN1,PLZ,ON,CKSF
TLLFT
GORGT/LN1,TO,LN4
ARCSP/FILLET,0.5
GOLFT/LN4,TO,LN3
GOLFT/LN3,TO,LN2
GOLFT/LN2,TO,LN1
ARCSP/FILLET,0
GOLFT/LN1,ON,CKSF
RAPID
GOTO/START
```



The following example shows the different tool paths generated by "ONCE" and "COMBIN" when the fillet radius specified is too large between two consecutive moves.



6 MISCELLANEOUS MOTION STATEMENTS

6.32.2 ARCSLP/ FILLET, SAVE

This command is used to save the active settings of the ARCSLP/FILLET,rad,.. command. No changes will be made to the ARCSLP/FILLET,rad,.. settings by this command.

6.32.3 ARCSLP/ FILLET, RESTOR

This command is used to restore the active settings of the ARCSLP/FILLET,rad,.. command saved by the ARCSLP/FILLET,SAVE command. This command will activate ARCSLP/FILLET,rad,, if it was active when the ARCSLP/FILLET,SAVE command was issued.

6.33 CHKPTS

The CHKPTS statement is used to assist you in generating points for a Coordinate Measuring Machine (CMM) or for other purposes which require specific control of the tolerance and spacing of motion generated points. The syntax is:

```
CHKPTS/tol,maxdp,maxang [ ,MACRO,m1 [ ,PS ] ]  
DS  
CHKPTS/NOMORE  
CHKPTS/REMOVE
```

CHKPTS is used in a manner similar to the [GENPTS](#) command. When a continuous path motion statement ([GOFWD](#), [GOLFT](#), etc.) is executed while a CHKPTS statement is in effect, the CL points and tool axis vectors generated are analyzed and only those required to satisfy the *tol*, *maxdp* and *maxang* values specified in the CHKPTS statement are output to the CL file.

The *tol* parameter specifies the maximum chordal tolerance. This is the maximum distance from the control surfaces (DS and PS) to the mid-point of a line that passes through any two consecutive output points.

The *maxdp* parameter specifies the maximum distance between any two consecutive output points.

This can be used to insure that points are output at least every *maxdp* even when driving relatively flat surfaces.

6 MISCELLANEOUS MOTION STATEMENTS

The *maxang* parameter specifies the maximum angle change in degrees between any two consecutive tool axis vectors.

If the optional MACRO clause is specified, the generated points are not output to the CL file, but are stored in temporary variables.

When a CHKPTS/ NOMORE statement is encountered, the macro specified by m1 is automatically called and the first five macro variables are assigned the following values:

- Var #1 - The number of points generated.
- Var #2 - The name of a reserved variable containing the points.
- Var #3 - The name of a reserved variable containing the forward vectors.
- Var #4 - The name of a reserved variable containing the tool axis vectors.
- Var #5 - The name of a reserved variable containing the vectors normal to the part surface, if PS is specified, or the drive surface if DS is specified. PS is assumed if neither value was specified in the MACRO clause.

Therefore, the macro specified by m1 must have been defined using a minimum number of five variables.

The CHKPTS/ REMOVE statement will cancel the CHKPTS statement and remove any temporary points and vectors that have been generated. This is useful when you want to stop the generation of points, because of an error condition or a change of mind, but you do not want the MACRO call to take place, as would be the case if the CHKPTS/ NOMORE statement were used.

This would be similar to typing a *TERMAC or [*RESET/ CALL](#) when you want to prematurely stop the processing of a Macro or Loop.

The following example will create a set of points within a .05 inch chordal tolerance of the part and drive surface, a maximum of 1 inch apart and within a maximum of 30 degrees change in tool axis vectors between points.

6 MISCELLANEOUS MOTION STATEMENTS

Upon execution of the CHKPTS/ NOMORE statement the macro m1 will be called and will output motion to each of the generated points with the tool axis set to a vector that is normal to the drive surface.

```
M1 =MACRO/N1,PTE,VFW,VTA,VNRM  
DO/100,I=1,N1  
100: GOTO/PTE(I),VNRM(I)  
TERMAC  
CUTTER/0  
THICK/.125  
GOTO/PT1  
GO/SF1,PL1,SF2  
CHKPTS/.05,1,30,MACRO,M1,DS  
TLLFT,GL/SF1,SF3  
CHKPTS/NOMORE
```

6.34 CONTCT/ ON OFF

The CONTCT statement provides the ability to drive a curve as a wire in space. This mode is activated or deactivated by ON or OFF.

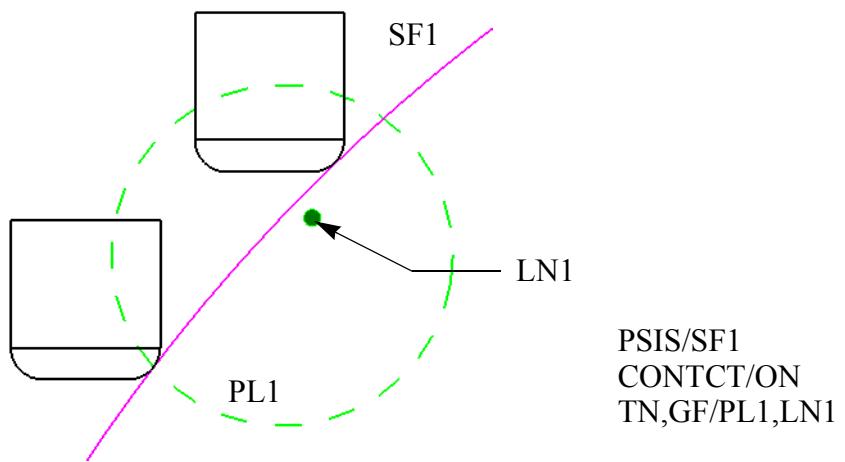
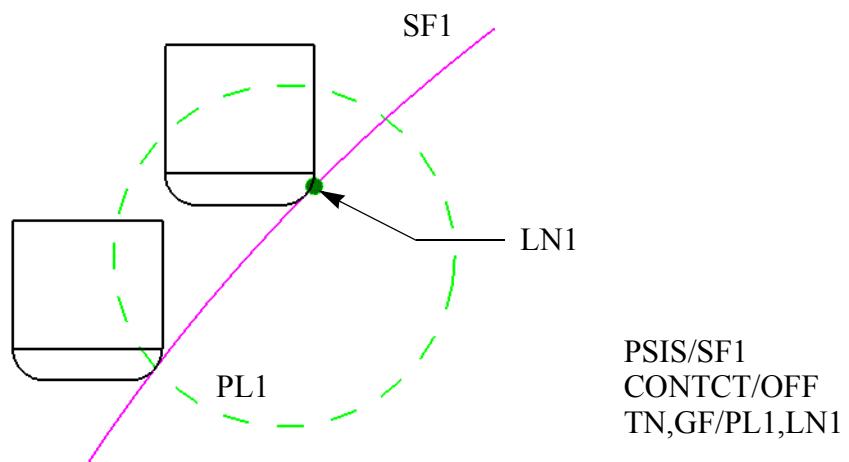
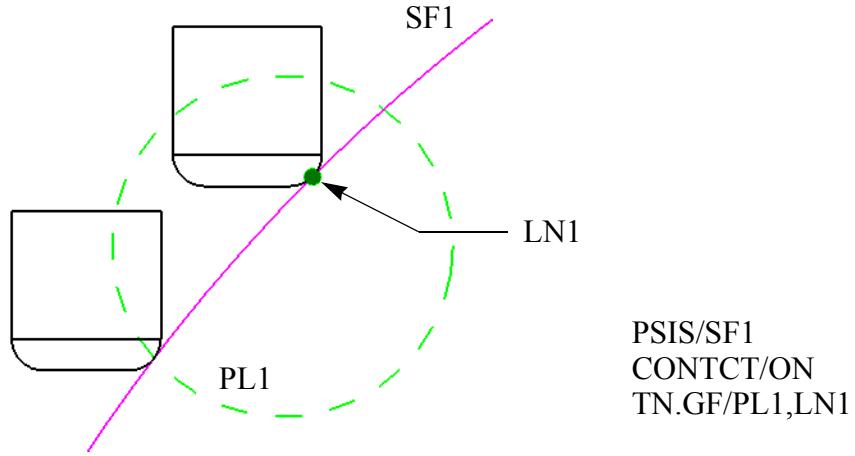
When ON is specified, **NCL** will keep a line, circle or curve drive surface in contact with the side or corner radius of the tool as appropriate rather than treat the tool as an infinitely long cylinder. If the drive surface drops below the bottom of the tool, contact will be maintained at a distance from the extension of the tool axis equal to the radius of the bottom flat portion of the tool. An angled cutter may be used with a line, circle or curve as the drive surface.

Note:

- CONTCT/ON is modal. Use CONTCT/OFF to turn it off.
- CONTCT/ON only works with fixed axis mode, no variable tool axis mode is allowed.

The following tool paths illustrate the results of CONTCT/ ON and CONTCT/ OFF. The default is CONTCT/ OFF.

6 MISCELLANEOUS MOTION STATEMENTS



6 MISCELLANEOUS MOTION STATEMENTS

6.35 COPY

The COPY statement notifies **NCL** to copy data from a location in the CLfile specified by an **INDEX - INDEX/NOMORE** set. The copied data is put at the current end of the CLfile. When more than one (1) copy is made by a COPY statement that causes some kind of alteration of the coordinate values to be made, the copies are altered as follows: The first copy is the original CLfile data altered by the specified method. The second copy is the coordinate values that were generated for the first copy altered by the specified method. The third copy would use the second copy's coordinates and so on. For example, if the original CLfile coordinates were "1, 2, 3" and the copy requested a translation of "4, 5, 6" then the first copy would be "5, 7, 9" (1+4, 2+5, 3+6). The second copy would be "9, 12, 15" (5+4, 7+5, 9+6).

6.35.1 COPY/ index [, SAME [, copies]]

Index specifies the identifying number of the **INDEX - INDEX/NOMORE** set of CLfile data. SAME indicates that the data is to be copied as is, not to be altered in any way. Copies specifies the number of times the data should be copied.

"copies" defaults to 1 if not specified. "COPY/index" is the same as "COPY/index,SAME,1".

6.35.2 COPY/ index, MODIFY, matrix [, copies]

Index specifies the identifying number of the **INDEX - INDEX/NOMORE** set of CLfile data. MODIFY indicates that the data to be copied is to be altered by applying the matrix against it before it is written to the current end of the CLfile. Copies specifies the number of times the data should be copied.

"copies" defaults to 1 if not specified.

6.35.3 COPY/ index, TRANSL, X-trans, Y-trans, Z-trans [, copies]

Index specifies the identifying number of the **INDEX - INDEX/NOMORE** set of CLfile data. TRANSL indicates that the data to be copied is to be altered by transforming the X, Y and Z values by the given X-trans value, Y-trans value and Z-trans value respectively as it is written to the CLfile. Copies specifies the number of times the data should be copied.

"copies" defaults to 1 if not specified.

6 MISCELLANEOUS MOTION STATEMENTS

6.35.4 COPY/ index, axis-rotation, angle [, copies]

Index specifies the identifying number of the INDEX - INDEX/NOMORE set of CLfile data. The axis-rotation specifies one of the following:

XYROT, YZROT or ZXROT

Each of these options indicates that the data to be copied is to be altered by transforming it through a matrix that will generate output data that is rotated about the Z, X or Y axis respectively in a counter-clockwise direction by an amount specified in degrees by the angle. Copies specifies the number of times the data should be copied.

“copies” defaults to 1 if not specified.

Example COPY Statements:

COPY/3, SAME, 2

(Copy INDEX 3 data two times without modifying data on the output)

COPY/1, MODIFY, MX3, 1

(Copy INDEX 1 data one time and modify the output by applying matrix MX3 against it)

COPY/6, TRANSL, 12.5, 3, .4, 15

(Copy INDEX 6 data fifteen times and modify the output by transforming the X values by 12.5, the Y values by 3 and the Z values by .4)

COPY/2, XYROT, 45, 2

(Copy INDEX 2 data two times and modify the output by rotating it about the Z axis in a counter-clockwise direction for 45 degrees)

Example of the use of a COPY statement in a part program:

```
INDEX/1
    series of motion commands. .
INDEX/1, NOMORE
COPY/1, TRANSL, 5, 0, 0, 2
```

6 MISCELLANEOUS MOTION STATEMENTS

Use of the **TRACUT** statement within the bounds of a COPY command:

Since the COPY command copies CL data generated between the **INDEX - INDEX/NOMORE** set, the data being copied has already been modified by the TRACUT command. Therefore, the TRACUT matrix is not re-applied to the motion being copied. The TRACUT statement has no effect on the copied motion other than that the motion being copied may have been initially modified by TRACUT. Furthermore, unlike APT, the tool is left in the last copied position after the COPY command has been executed.

6.36 CUT

The CUT statement causes **NCL** to resume generating a CLfile output following the use of a DNTCUT command. The CUT statement will cause the current location of the cutter to be output to the CLFILE. If this move is not desired, the DNTCUT/NOMORE statement should be used instead of the CUT statement.

The **DNTCUT - CUT** commands are used in situations where it might be necessary to use several intermediate motion commands to arrive at a desired position.

These intermediate moves are not output to the CLfile, only the final position. A use for the DNTCUT - **DNTCUT/NM** commands would be situations where tool motion is being used to generate geometry, using such commands as **GENPTS** or **POINT/TE**. In these cases it would usually not be desirable for any of the tool motion to be output to the CLfile.

The valid syntax construct for the CUT statement is:

CUT

6.37 DNTCUT

The DNTCUT statement controls the writing of generated motion to the CLfile. This causes **NCL** to stop writing motion information to the CLfile. This will continue until a CUT statement or a DNTCUT/ NOMORE statement is processed. See the CUT command for a more detailed explanation of this command.

6.37.1 DNTCUT/ NOMORE

This causes **NCL** to start writing motion to the CLfile at the next motion statement. See the CUT command for a more detailed explanation of this command.

6.38 FEDRAT

The FEDRAT statement is used to specify the desired feed rate for the cutter during a move, or a sequence of moves. Once a feed rate has been specified, it will remain in effect until a new feed rate is specified. **NCL** will allow any feed rate to be used and it is up to the user to make sure that what has been specified is valid for the particular machine tool. Many motion statements may have an optional feed rate value as the last item in the statement. This is known as a “tag on” feed rate. A “tag on” feed rate sets the primary feed rate to be used for that motion statement and any subsequently processed motion statements until the feed rate is changed by a FEDRAT statement or another “tag on” feed rate. For example:

```
GO/SF1, PLZ, SF2, 15  
FD=10  
GOFWD/SF1, PAST, SF2, FD
```

The valid syntax constructs for the FEDRAT statement are as follows:

6.38.1 FEDRAT/ feedrate [,IPM] IPR

The primary feed rate is specified. The optional modifier may be either IPM (inches per minute) or IPR (inches per revolution). If omitted, IPM is assumed.

6.38.2 FEDRAT/ [AT , dist-1 [, SCALE] , fs1 [, nfed] [, ONCE] \$ [[,] OUT [[, dist-2] [[, SCALE], fs2] [, nfed]] \$ [, ONCE] [, LENGTH , dis]

This syntax is used to initiate secondary feed rate control. Secondary feed rates can be used to slowdown the feed rate when approaching and/or moving away from a corner (a Drive Surface/Check Surface intersection.)

Where:

AT	Initiates slow down feed rates.
dist-1	Is the distance from the check surface at which the slow down feed rate will take effect. If the total move

6 MISCELLANEOUS MOTION STATEMENTS

is less than *dist-1*, then the entire move is made using the slow down feed rate.

fs1	Specifies the slow down feed rate value. If the value is preceded by the word <i>SCALE</i> , then <i>fs1</i> is applied as a factor against the current <i>general</i> feed rate to calculate the slow down feed rate.
nfed	Specifies the new <i>general</i> feed rate (optional).
ONCE	Specifies that the secondary feed rate will be applied to the next motion command only.
	The previous FEDRAT condition will then be restored for subsequent motion commands.
OUT	Initiates acceleration feed rates.
dist-2	Distance from the current position to the point where the <i>general</i> feed rate will be applied. If the total move is less than <i>dist-2</i> , then the entire move is made using the acceleration feed rate. If no value is specified for <i>dist-2</i> , then the value specified for <i>dist-1</i> will be used.
fs2	Specifies the <i>acceleration</i> feed rate value. If the value is preceded by the word <i>SCALE</i> , then <i>fs2</i> is applied as a factor against the current <i>general</i> feed rate to calculate the <i>acceleration</i> feed rate. If no value is specified for <i>fs2</i> then the value specified for <i>fs1</i> will be used.
LENGTH, dis	Specifies that the distance (<i>dist-1</i> or <i>dist-2</i>) that controls where the secondary feed rates will be output is calculated at a point that is a distance of <i>dis</i> up the current tool axis.

If the total move from the current position to the check surface is less than the sum of *dist-1* and *dist-2* then the smaller value of *fs1* and *fs2* will be used for the common area.

“*OUT, 0*” must be specified to turn the acceleration feed rate off. That is, “*FEDRAT/ AT, 0*” only turns off the slow down feed rate, not the acceleration feed rate.

6 MISCELLANEOUS MOTION STATEMENTS

Examples:

FEDRAT/AT,.5,5,OUT,.75,7

Specifies that the feed rate will start at 7 until the tool is .75 units from the current position, at this point the feed rate will change to the general feed rate. When the tool is .5 units from the check surface, the feed rate will slow down to 5 until the check surface is reached. Furthermore, if the acceleration parameters are the same as the slow down parameters only the word *OUT* need be specified to activate acceleration.

FEDRAT/AT,.25,5,OUT

is equivalent to:

FEDRAT/AT,.25,5,OUT,.25,5

The statements:

FEDRAT/30
FEDRAT/AT,.125,10
GOFWD/LN1,TO,LN2

Will cause the cutter to move along LN1 at a rate of 30 IPM until it gets to within .125 of LN2. At that point, the cutter will slow down to 10 IPM.

Then, if the next statement is:

GORGT/LN2,TO,LN3

the cutter will move along LN2 at 30 IPM until it gets to within .125 of LN3, at which point it will again slow down to 10 IPM.

If the optional *SCALE* is specified, the scalar will represent a factor of the primary feedrate instead of a feed rate value. Therefore, the statement:

FEDRAT/AT,.125,SCALE,.1

means that when the cutter gets to within .125 of the check surface, the feed rate will slow down to 10% of the primary feed rate.

6 MISCELLANEOUS MOTION STATEMENTS

6.39 GOUGCK

The GOUGCK statement notifies **NCL** that Gouge Checking should be performed during cutter motion generation. Specifically, gouge checking is important when the part surface is a contoured surface and the cutter is shaped so that one side of the cutter may be in contact with the part surface while the other side of the cutter is gouging that surface. An example would be a flat bottom cutter moving over a concave part surface. The default is no gouge checking.

6.39.1 GOUGCK/ PS , 0, DS, 0, CS, 0

1	1	1
2	2	2
3	3	3
4	4	

This causes gouge checking to be performed during cutter motion generation. Note that gouge checking requires more work to be performed by **NCL** during cutter motion, therefore motion will take longer to calculate with gouge checking.

The numbers indicate the level of gouge checking to be performed to the **Part Surface** (PS), **Drive Surface** (DS) and **Check Surface** (CS). Each surface can have its own level of gouge check. They do not need to be the same. Following is a description of the gouge check level.

Part Surface

- Level 0 Uses the tool look point from the previous step.
- Level 1 Calculates a new look point with each step.
- Level 2 Uses a look point from the front, center, and back of the tool at each step.
- Level 3 Uses a look point from the front, center, back, right, and left of the tool.

Drive Surface

- Level 0 Calculates a stand tool look point.
- Level 1 Same as Level 0.
- Level 2 In addition to the standard look point, uses a look point at the corner radius level.

6 MISCELLANEOUS MOTION STATEMENTS

- Level 3 Same as Level 2.
- Level 4 Is in effect when the Check Surface has the geometric type of a surface and the motion attribute for it is TO. Projections from several look points along the tool axis are considered, plus the projection from the forward look point at the corner radius level. The algorithm is trying to select the best surface projection, preferring actual surface point projections in the forward direction to extensions.

Check Surface

- Level 0 Calculates a standard tool look point.
- Level 1 Same as Level 0.
- Level 2 Uses look points at various heights of the tool: tool end point, corner radius, middle of the tool, and at the top of the tool. Also performs additional gouge avoidance when driving TANTO a surface.
- Level 3 Same as Level 2 plus even better gouge avoidance when driving TANTO to a surface.
- Level 4: Is in effect when the Drive Surface has the geometric type of a surface and the motion attribute for it is TLLFT or TLRGT. The projection from the surface side look point at the corner radius level is considered. If it finds a surface point within tolerance to the cutter, while surface points found by other projections are farther away from the cutter, the result is accepted as the current Drive Surface contact point.

A value of “0” all across is same as the statement GOUGCK/OFF.

A value of “1” all across is same as the statement GOUGCK/ON,1.

A value of “2” all across is same as the statement GOUGCK/ON,2.

A value of “3” all across is same as the statement GOUGCK/ON,3.

6 MISCELLANEOUS MOTION STATEMENTS

6.39.2 GOUGCK/ ON [, 1]

2

3

This causes gouge checking to be performed during cutter motion generation. Note that gouge checking requires more work to be performed by **NCL** during cutter motion, therefore motion will take longer to calculate with gouge checking. This statement does not give the user the ability to control the gouge check level for the Part Surface, Drive Surface or Check Surface individually.

The number that appears after the ON parameter indicates the level of gouge checking to be performed. 1 is the default.

See [GOUGCK/PS,n,DS,m,CS,p](#) statement for detail description of each level of gouge check.

6.39.3 GOUGCK/ OFF

This notifies **NCL** to stop gouge checking.

The GOUGCK command may be added to the end of any of the variable tool axis statements or may be specified in a statement by itself in which case it will apply to the tool axis mode currently in effect.

6.40 INDEX

The INDEX statement is used to indicate the beginning and end of a set of CLfile data.

6.40.1 INDEX/ id number

This construct defines the beginning of a set of CLfile data. The id number specifies the identifying number of the INDEX - INDEX/NOMORE set. It is the number that the COPY statement uses to identify which INDEX - INDEX/NOMORE set of data is to be copied.

6.40.2 INDEX/ id number, NOMORE

This construct defines the end of a set of CLfile data. The scalar is the identifying number of the INDEX - INDEX/NOMORE set.

One INDEX - INDEX/NOMORE set of statements may overlap another or be nested within another set.

Example INDEX Statements:

```
INDEX/2  
INDEX/5, NOMORE
```

Example of valid nesting of INDEX statements:

```
INDEX/1  
•  
•  
INDEX/2  
•  
•  
INDEX/2, NOMORE  
•  
•  
INDEX/3  
•  
•  
INDEX/1, NOMORE  
..  
INDEX/3, NOMORE
```

INDEX 2 is completely contained within INDEX 1. INDEX 3 begins inside of the set defined by INDEX 1 but ends outside of the INDEX 1 set.

6.41 MAXANG

The MAXANG statement controls the Angle of Change in the tool axis vector during all variable tool axis modes. The valid syntax construct for the MAXANG statement is:

```
MAXANG/angle[, ONCE]
```

6 MISCELLANEOUS MOTION STATEMENTS

The maximum angular change between each output point generated during a motion statement is specified by the "angle." If not specified, the default is five (5) degrees.

ONCE indicates the value specified will be used for the next **GO**, continuos motion (**GOFWD**, **GORTG**, etc.) or **RMILL** statement. The value will then revert to the value in effect at the time the MAXANG with the ONCE modifier was issued.

6.42 MAXDP

The MAXDP statement notifies **NCL** to take specific sized steps along the drive surface when it is looking for the check surface during a motion calculation.

The valid syntax construct for the MAXDP statement is:

```
MAXDP/max-step [,STEP,ON ] [,AUTO] [,max-loops]      $  
                  OFF      OFF  
  
                  [,min-dp] [,NOWARN] [,ONCE]  
                  WARN
```

The "max-step" indicates the length of step the processor takes when it is looking for the check surface. This syntax is useful when the cutter is located close to the check surface at the start of a motion command. The default value for MAXDP is 4 inches when units are in inches and 100 MM when units are in millimeters.

"STEP,ON" specifies that **NCL** will never output cl points more than a distance of "max-step" apart. "STEP,OFF" is the default setting.

If the optional word AUTO is used, **NCL** will automatically alter the step size value temporarily during the processing of a motion statement if it fails to find the check surface due to the current MAXDP value not being appropriate. This alteration is done in a repetitive, looping mode that is controlled by the "max-loops" and "min-dp" values. If a motion statement fails, the MAXDP step is changed by dividing the current value by four and then re-processing the statement.

This process is continually repeated until the statement successfully computes the location where the cutter contacts the check surface, or the value is modified the number of times specified in the "max-loops" option, or the modified value is smaller than the specified "min-dp" value. If no "max-loops" value is given, it will

6 MISCELLANEOUS MOTION STATEMENTS

default to 10. If no "min-dp" value is given, it will default to 10 times the current value of **TOLER**.

OFF indicates no automatic modification of the "max-step" value is to be done. In this mode, if the processing fails to compute the location where the cutter contacts the check surface with the current "max-step" value, an error is reported and no modification of the "max-step" value is done. AUTO is the default setting.

The words **WARN** and **NOWARN** tell **NCL** whether or not to output a warning to the user indicating that **MAXDP** has been internally modified to process a particular move as well as what the "max-step" value was changed to. The default is **NOWARN**.

ONCE indicates the values specified will be used for the next **GO**, continuous motion (**GOFWD**, **GORGT**, etc.) or **RMILL** statement. The values will then revert to the values in effect at the time the **MAXDP** statement with the **ONCE** modifier was issued.

Note: The **MAXDP/...,ONCE** command will only work with the STEP value if the *SET/VER,97 or earlier is specified.

Examples:

```
MAXDP/4,AUTO,5  
MAXDP/2,AUTO,8,.05
```

6.43 MULTAX

The **MULTAX** statement notifies the **NCL** processor to output tool axis vectors along with tool motion in the form X, Y, Z, I, J, K.

It is generally recommended that the desired **MULTAX** statement be placed in the part program before any motion statements are processed. The **MULTAX** condition should never be changed in middle of a series of **NCL** motion commands or adverse effects can occur. Certain **NCL** commands, such as **COPY**, assume that ALL points written to the CL file were written with the **MULTAX** condition in effect at the time the command is processed. Adverse effects can also occur when postprocessing a CL file containing multiple conditions of **MULTAX**.

There are, however, cases where a user may need to turn on and off the output of tool axis vectors to the CL file within a part program. Check the output carefully if this strategy is employed.

The valid syntax constructs for the **MULTAX** statement are:

6 MISCELLANEOUS MOTION STATEMENTS

6.43.1 MULTAX [/ ON]

This construct turns MULTAX on and causes data to be output to the CLfile in the format X, Y, Z, I, J, K.

6.43.2 MULTAX/ OFF

This construct turns MULTAX off and causes data to be output to the CLfile in the format X, Y, Z.

6.44 NUMPTS

The NUMPTS statement notifies the **NCL** processor to abort motion output if the number of generated CLfile points calculated during the processing of a motion statement exceeds the value given in the NUMPTS statement. The default value for the number of points if no NUMPTS statement is given is 100. The valid syntax construct for the NUMPTS statement is:

NUMPTS/points [,ONCE]

Points specifies the maximum number of points to be calculated during the processing of each motion generation statement.

ONCE indicates the value specified will be used for the next **GO**, continuous motion (**GOFWD**, **GORGT**, etc.) or **RMILL** statement. The value will then revert to the value in effect at the time the NUMPTS statement with the ONCE modifier was issued.

Example NUMPTS Statements:

NUMPTS/250
NUMPTS/25

6.45 PRINT

This form of the PRINT statement is used to control the number of CLfile coordinates to display in the scrolling window during an interactive session or to the print file (.pr) created during batch processing. See the Geometry Statements section for other forms of the **PRINT** statement. The two valid syntax constructs are:

6.45.1 PRINT/ SMALL

This construct causes only the first and last coordinates generated by each motion statement to be printed during batch processing or displayed in a scrolling window during interactive session. This is the default setting.

6.45.2 PRINT/ LARGE

This construct causes all coordinates generated to be printed during batch processing or displayed in a scrolling window during interactive session.

6.46 RAPID [/ Optional Parameters]

Rapid with no optional parameters caused the next motion to move at the rapid traverse rate. Consult the postprocessor reference manual (such as **PostWorks**) for allowable optional parameters.

6.47 REVERS/ ON OFF

This command enables the programmer to reverse the generated motion between the "REVERS/ON" and "REVERS/OFF" commands. The first goto point generated by the motion will be the last point output to the cl file. The **FEDRAT** and **CUTCOM** commands will be adjusted in the output clfile so that each reversed move would still have the correct feed rate and cutter compensation setting as it had in the programmed tool path, without requiring extra commands at the start and end of the move.

The motion displayed on the screen while the part program is being processed during interactive session will not be reversed. In order to playback the reversed motion the user must set the Source field in the Playback File Form to "Reversed".

6.48 SEQUNC

This command allows you to define separate tool paths for different sections of a part program. These tool paths can then be referenced when using the optional **NCL/IPV** (in-process tool path verification) module and when back plotting a CLfile. The SEQUNC command has the following syntax:

SEQUNC/label

6 MISCELLANEOUS MOTION STATEMENTS

Defines the start of a toolpath. *label* assigns a name to this toolpath and can either be a number (or scalar) or an alpha-numeric string of 1 to 20 characters in length

SEQUNC / END

Defines the end of a toolpath. An active toolpath will be ended whenever a SEQUNC/END command is encountered or another toolpath is started using the SEQUNC/*label* command.

The SEQUNC command is output to the CLfile and is used to mark the starting location of a toolpath. The SEQUNC CLfile record contains the following information:

1. Toolpath label
2. Previous tool position
3. Cutter definitions - both actual and displayed
4. CUTTER/ DISPLAY, . . . status
5. Current feed rate
6. MULTAX status
7. Current TRACUT matrix

6.49 SET

The SET statement is used to specify whether linear or circular interpolation data is to be output to the CLfile when circular motion is generated.

Note: SET is different from the *SET command.

6.49.1 SET/ MODE, CIRCUL

This notifies **NCL** to output circular interpolation formatted data to the CLfile whenever it can. **NCL** will output circular record if the drive surface is of type "CIRCLE" or "CYLINDRICAL SURFACE" normal to the part surface' and the part surface is either of type "PLANE" or "[PLANAR SURFACE](#)".

There was an optional integer value that could be appended to the SET/ MODE, CIRCUL command in versions of **NCL** prior to Version 3.0. This optional integer indicated the type of circular interpolation records that the post processor expected. If the postprocessor was obtained from **NCCS**, the integer should have always been "1". If the postprocessor was obtained from Westinghouse or AI, the integer should have always been "0". The default was "0". This is no longer required and should not be used on the statement.

When writing a program for a machine that has a postprocessor that uses circular interpolation type data, it is usually preferable to set the mode to circular interpolation since the circular interpolation mode generally produces a shorter output file than the same program using linear interpolation. The amount of difference depends on the amount of motion on circular type geometry.

For previous versions, **NCL** will only output circular record if the part surface is of the geometry type "PLANE". Surfaces created in previous versions must be analyzed with the **ANALYZ** command before they can be specified as part surfaces or drive surface. Otherwise, no circular record will be output even if the part surface is flat and the drive surface is of the cylindrical shape and normal to the part surface.

6.49.2 SET/ MODE, LINEAR

This notifies **NCL** to output linear formatted data to the CLfile at all times. The default mode is CIRCUL.

6.50 THICK

The THICK statement is used to define the offset from the controlling surface that the cutter must maintain. The word THICK may be abbreviated as TH. The valid syntax constructs for the THICK statement are:

6.50.1 THICK/ distance-PS [, distance-DS [, distance-CS]]

Distance-PS, specifies the offset distance from the part surface. Optional distance-DS, specifies the offset distance from the drive surface; if omitted, the value specified for the part surface is used.

Optional distance-CS, specifies the offset distance from the check surface; if omitted, the value specified for the drive surface is used. Initial offset values are 0.

6.50.2 THICK/ ps, ds, cs1 [, cs2 [, cs3 [, cs4 [, cs5]]]]

This statement provides you with the ability to define a separate THICK for each check surface when using multiple check surfaces. The "cs1" value will be used for any of the "csn" not specified.

6 MISCELLANEOUS MOTION STATEMENTS

6.50.3 THICK/ OFF

This construct sets all three offsets to 0.

Example THICK Statements:

```
THICK/.01  
TH/.01,.02  
THICK/.5,.5,1  
THICK/0,.5  
THICK/.5,.5,1,.75,.5,.25,0  
THICK/OFF
```

6.51 TOLER/ chor-tol [, pos-tol]

The TOLER statement specifies the acceptable cutter tolerance deviation from controlling surfaces during motion generation. The valid syntax construct for the TOLER statement is:

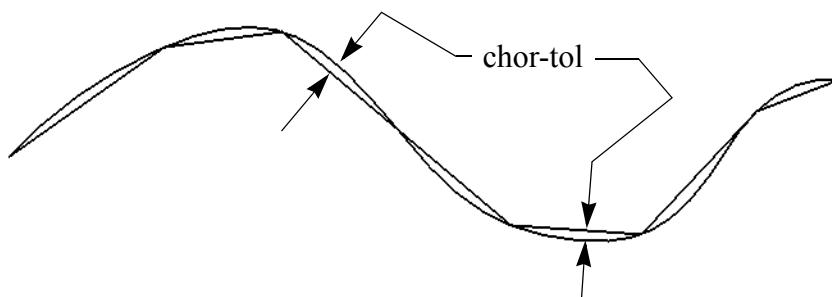
```
TOLER/chor-tol[,pos-tol]
```

Where chor-tol is the maximum chordal tolerance and pos-tol is the maximum distance from the drive surface and part surface at which the cutter will be positioned. The maximum from the check surface at which the cutter will be positioned is indirectly controlled by pos-tol.

The distance specifies the acceptable tolerance. If no TOLER is given, the default value is .001 (chor-tol), .001 (pos-tol) if UNITS are set to inches and .025, .025 if UNITS are set to MM.

Example TOLER Statement:

```
TOLER/.001,.0001 $$ (inches)  
TOLER/.025,.0025 $$ (millimeters)
```



6 MISCELLANEOUS MOTION STATEMENTS

The TOLER statement affects both motion and geometry. As a rule, the results will always be better than the tolerance setting.

While the statement only has two parameters, the following indicates what actually happens internally when the statement is applied.

default value	toler/.001, .001
internal value	toler/.001, .001, .0001

Notice the third value .0001.

For Motion:

```
toler/chordal, positional, positional  
          part-surf   check-surf  
          drive-surf
```

The following statements indicate what happens in each case:

`toler/.002`

Internally toler is set to toler/.002, .002, .0001.

`toler/.002, .001`

Internally toler is set to toler/.002, .001, .001.

If you then state:

`toler/.001`

In inches:

Internally toler is reset to toler/.001, .001, .0001.

In metric:

Internally toler is set to toler/.025, .025, .0001.

Note:

1. If only one value is specified with the TOLER statement, the second value, i.e. the "part-surf/drive-surf positional" value, is set equal to the first value while the third internally value, i.e. the "check_surf positional" value, will revert back to the default value which is always 0.0001 no matter what is the current units.

6 MISCELLANEOUS MOTION STATEMENTS

2. If two values are specified with the TOLER statement, the third internally value, i.e. the "check-surf positional" value, will be equal to the second value specified in the corresponding TOLER statement.
3. The second value specified must be smaller than or at most equal to the first value specified in the TOLER statement.

For Geometry:

Only the first value of toler is used.

```
toler/.010,.001
```

Only the value .010 is used for a multitude of items. It is used primarily for curves and surfaces, but can influence point definitions and the results of functions. For example,

- When defining a curve, not using a fit tolerance is better for the internal calculation for the best fit. Usually this does not affect you but you should be aware of it.
- When defining a curve/ fit "toler" is used to allow the curve to pass toler distance away from a defining point. A curve must pass through the start and end points, but any intermediate points, when using fit, can be toler distance away from the points.
- When defining a circle or curve, if the defining points are closer to each other than toler, **NCL** will generate an error.
- When obtaining a distance between a point and a surface.
- When obtaining the distance along a curve (length).
- When defining a point along a curve at a specified distance.
- When defining a surf/ fit or nsurf/ fit.

6.52 TRACUT

The TRACUT statement causes CLfile information to be translated through a matrix. The valid syntax constructs for the TRACUT statement are:

6.52.1 TRACUT/ matrix

This TRACUT (TRAnslate CUTter locations) command is used to apply an alternate coordinate system to the output cutter locations. This will usually be the

6 MISCELLANEOUS MOTION STATEMENTS

coordinate system on the machine. For example, if the part geometry were such that it needed to be rotated 90 degrees around the Z axis in order to fit on the machine the following statements could be used:

```
TRMX =MX/XYROT, 90  
TRACUT/TRMX
```

As demonstrated in the example a matrix is used to define the alternate (or machine) coordinate system. Any motion generated following the TRACUT statement would be output in the coordinate system defined by the matrix TRMX. This would continue until another TRACUT/mx or TRACUT/NM statement was encountered.

Note that TRACUT only affects the output data and does not affect the way **NCL** “sees” the geometry. To make **NCL** “see” both the geometry and tool motion (including the output data) in an alternate coordinate system use the **MODSYS** command.

6.52.2 TRACUT/LAST, matrix NOMORE

The TRACUT/LAST,matrix command allows to program a secondary TRACUT that is executed after the regular TRACUT until the TRACUT/LAST,NOMORE is encountered. When both TRACUT/matrix and TRACUT/LAST,matrix are in effect, CL points are first transformed by the TRACUT matrix, then by the TRACUT/L:AST matrix.

Example:

```
M1      =MATRIX/TRANSL,10,10,10  
M2      =MATRIX/XYROT,90  
TRACUT/MX1  
TRACUT/LAST, MX2  
          Motion statements transformed by MX1 and MX2  
TRACUT/LAST, NOMORE  
          Motion statements transformed by M1 only
```

6.52.3 TRACUT/ NOMORE

This construct notifies **NCL** that TRACUT is no longer in effect.

Example TRACUT Statements:

6 MISCELLANEOUS MOTION STATEMENTS

TRACUT/MX1
TRACUT/NOMORE

6.53 TRALST/ ON OFF

This construct turns ON/OFF the ability to see the translated motion points and vectors through the current TRACUT matrix before being output to the status window or written to the batch listing file. This statement will not affect the tool motion displayed in the graphic window. Use the [DRAFT/CUTTER,...](#) statement to see motion relative to the current TRACUT matrix.

Example TRALST Statements:

TRALST/ON
TRALST/OFF

6.54 UNITS/ MM INCHES

The UNITS statement is used to specify the input and output units. The unit may be inches or millimeters. When UNITS/ INCHEs is specified, the program variable [TOLER](#) defaults to .001 and [MAXDP](#) defaults to 4. When UNITS/ MM is specified, TOLER defaults to .025 and MAXDP defaults to 100.

The UNITS statement should be specified before any geometry or motion is generated or unpredictable results will occur. It is a good practice to have the UNITS statement be one of the first statements in the program.

Example UNITS Statements:

UNITS/ INCHEs (default)
UNITS/MM

The default may be INCHEs or MM by setting the parameter U_UNITS in the following file:

C:\NCCS\NCL100\interface\ncl.init

7

PROGRAM CONTROL STATEMENTS

7.1

CALL

The CALL statement invokes the execution of a previously defined macro. Upon completion of the macro's execution, control is returned to the statement following the CALL.

7.1.1

CALL/ macro-name [, parameter list]

This statement causes control to be passed to the macro specified by macro-name. The optional parameter list may be used to supply values for any parameters associated with that macro. Valid values are scalars, vocabulary words, geometry names and named nested geometry. See the section describing the MACRO statement for more information about MACRO parameters.

Example CALL Statements

```
CALL/M1  
CALL/M1, A=6, PRTSRF=PL1, DIR=TLRGT
```

NCL will attempt to automatically include a macro file when a CALL/macro-name statement is issued and “macro-name” is not previously defined. The file will be named “macro-name.mac” and can reside in the local (current) directory or the system directory (NCL_INCDIR).

7.2

CONTIN

The CONTIN statement is a null statement and is included only as a filler. The statement containing the DO loop terminator label may contain a regular **NCL** statement, however, many times the CONTIN statement is used for clarity. The following example illustrates a DO loop within another DO loop:

```
DO/L1, X=1.5, 4.5, INCR, 1.5  
    DO/L2, Y=1, 2  
        GOTO/X, Y  
    L2 :CONTIN  
L1 :CONTIN
```

The execution of these DO loops will cause the following:

7 PROGRAM CONTROL STATEMENTS

```
GOTO/1.5,1  
GOTO/1.5,2  
GOTO/3.0,1  
GOTO/3.0,2  
GOTO/4.5,1  
GOTO/4.5,2
```

Notice that the INCR for the outer DO loop specifies value of 1.5, therefore the outer DO loop is executed 3 times and the inner DO loop defaults to an INCR of 1 and executes 2 times.

It is illegal to branch out of a DO loop with a **JUMPTO** or an **IF** statement. However, a DO loop may be exited early with an **UNDO** statement.

7.3 DO

The DO statement allows the definition of a DO loop.

A DO loop is a way of automatically looping for a certain number of times through a range of statements. The properties of a DO loop are similar to a **LOOPST/LOOPND** loop, however, the DO loop is more flexible and in some ways more powerful. The valid syntax for the DO statement is:

```
DO/label, index-var=start-val, stop-var[, INCR, incr-var]
```

Where:

label	Specifies the ending statement label of the DO loop range. The ending statement may not be a MACRO CALL statement.
index-var	The name of the scalar variable which is the loop counter. The loop counter is automatically updated each pass through the looping range before the execution of the first statement in the loop. The loop counter contains the limit value or last updated value after all passes of the loop execution are completed or an UNDO statement is executed.
start-val	Specifies the initial value for the index-variable at the start of the loop.

7 PROGRAM CONTROL STATEMENTS

stop-var Specifies the limit value for the index-variable which will end the loop.

incr-var Specifies the amount that the index-variable will be increased each time through the loop.

If the start-value is greater than the stop-value and the incr-value is positive or the start-value is less than the stop-value and the incr-value is negative, the loop will execute once.

For example:

```
DO/ ABC, X=1, 10, INCR, 3
```

The incr-value may be positive or negative, however, it may not be zero. If INCR is omitted, the increment value will be 1.

A DO loop may appear at any point in a program, including within a MACRO, and within a LOOPST/ LOOPND pair. DO loops are also allowed within DO loops. A DO loop that is contained within another DO loop is called a nested DO loop. The following example illustrates a simple DO loop:

```
DO/100, INX=1, 10  
GOTO/INX, 1  
100:CONTIN
```

The above DO loop will loop 10 times, because the loop counter INX starts with a value of 1 and is increased by a value of 1 each time through the loop until it reaches a value of 10. Therefore, the GOTO statement will execute 10 times in the following manner:

1. GOTO/1, 1
2. GOTO/2, 1
3. GOTO/3, 1
- ...
- ...
10. GOTO/10, 1

At the end of the DO loop, INX will have a value of 10 and may be used as a normal scalar variable throughout the program.

A nested DO loop must be completely contained within the outer DO loop's limits. The following set of statements would be invalid because the inner DO loop's

7 PROGRAM CONTROL STATEMENTS

ending label id (INEND:) is found after the outer DO loop's ending label (OUTEND:).

DO/OUTEND , A=1 , 10	\$\$ Beginning of outer DO
DO/ INEND , B=2 , 5	\$\$ Beginning of inner DO
GOTO/A , B	
OUTEND:	\$\$ Ending label of outer DO loop
INEND:	\$\$ Ending label of inner DO loop

DO loops have the following restrictions:

- It is illegal to branch out of a DO loop, except using an UNDO statement.
- The increment value may not be zero.
- A MACRO CALL statement may not appear on the last line of a DO loop.
- It is legal to modify the value of the loop counter within the DO loop, however, this should be done with extreme caution as an infinite loop may occur.

7.4

FINI

The FINI statement notifies **NCL** that the current part program is complete. Processing of part program statements is stopped during batch process. This statement has the similar effect as the ***STOP** statement during interactive session.

7.5

FORMAT

The FORMAT statement controls the number of decimal places printed in the formatted variables passed via an **INSERT** or **PPRINT** statement using scalar variables.

7.5.1

FORMAT/SHORT

This causes four decimal places to be output for each scalar variable found in an **INSERT** or **PPRINT** statement.

SHORT is the default value.

7.5.2 FORMAT/LONG

This causes six decimal places to be output for each scalar variable found in an [INSERT](#) or [PPRINT](#) statement.

7.6 IF Statements

There are three types of IF statements. They are: IF (arithmetic), IF (Logical), and [IF-THEN-ELSE](#) structure.

All types of IF statement can only exist inside a LOOP or a macro definition except a [special case](#) of the IF(logical) statement.

7.6.1 IF (arithmetic)

The arithmetic IF statement is used to cause a conditional transfer of control to occur. An arithmetic IF statement may only appear within a macro or a loop. The valid syntax construct for the arithmetic IF statement is:

```
IF (expression) label-1, label-2, label-3
```

If the value of the expression is less than zero, control is passed to the statement specified by label-1. If the value is zero, control is passed to the statement specified by label-2. If the value is greater than zero, control is passed to the statement specified by label-3.

The statements specified by label-1, label-2 and label-3 must all be contained within the current loop or macro.

Example arithmetic IF Statements:

```
IF (X) L1, L2, L3  
IF (A- (B/3) **2) 10,10,20
```

7.6.2 IF (logical)

The Logical IF statement can be used to test a logical expression in order to cause one of two possible actions to be taken depending upon whether the logical expression is True or False. The valid syntax is:

```
IF (logical-expression) statement-id  
[ , ] NCL statement
```

7 PROGRAM CONTROL STATEMENTS

where:

"logical-expression" can be one of the following:

1. An expression comparing scalar values using the operators:

'LT' - less than
'LE' - less than or equal to
'EQ' - equal to
'NE' - not equal to
'GE' - greater than or equal to
'GT' - greater than

The expression must be of the form:

scalar-1 'operator' scalar-2

2. An expression which is a complex expression comparing two sub-expressions and containing the operators:

'AND' - sub-expression-1 'AND' sub-expression-2
'OR' - sub-expression-1 'OR' sub-expression-2

For example:

IF (A 'LT' B 'AND' A 'LT' C) L12

3. An expression or sub-expression may also contain the 'NOT' operator. This will have the effect of reversing the TRUE/FALSE value of the expression it is appended to.

For example:

IF ('NOT' A 'LT' B) GOTO/PT1

4. A geometry identifier. When this kind of expression is evaluated, the value of the expression will be TRUE if the entity is defined (exists) and FALSE if it is not defined.

"," is optional and can be specified with any **NCL** command except JUMPTO or statement label.

"statement-id" is a statement in the program that is branched to if the value of the logical expression is TRUE.

7 PROGRAM CONTROL STATEMENTS

"**NCL**-statement" is any valid **NCL** statement that does not cause an auto name generation.

A Logical IF statement will be processed as follows:

1. The expression will be evaluated going from left to right. Parentheses can be used and will cause expressions within the parentheses to be evaluated first. Any level of nested parentheses is allowed.
2. If the expression is evaluated to be TRUE, a JUMPTO the statement label given will be performed or the **NCL** statement will be processed.
3. If the expression is evaluated to be FALSE, no JUMPTO (or **NCL** statement) will be performed and the next sequential statement will be processed.

Restrictions:

1. A Logical IF (expression) JUMPTO/label or IF (expression) label statement may only appear in a MACRO, DO loop or LOOPST/LOOPND range.
2. The statement label or **NCL** statement used in a Logical IF statement must be associated with a statement within the current looping region range. It may not be in a calling MACRO's looping range.
3. A Logical IF statement that is testing for the existence of the entity may only contain a single geometry identifier. It may not be combined with any other operands or operators.
4. The "IF (expression) [,] Command" statement can be used anywhere in the program if "Command" is not a "JUMPTO/label" or "label", i.e. it is not restricted to appear only in a MACRO, DO loop or LOOPST/LOOPND range.

7 PROGRAM CONTROL STATEMENTS

7.6.3 IF-Then-Else Structures

The IF-Then-Else structure allows the programmer to set conditions for **NCL** to execute groups of commands based on one or more expression evaluations. The valid syntax is:

```
If (expression_1) Then  
  NCL statements  
Else if (expression_2) Then  
  NCL statements  
Else if (expression_3) Then  
  NCL statements  
Else  
  NCL statements  
Endif
```

where:

“expression_n” can be one of the following logical expression:

1. An expression comparing scalar values using the operators:

- ‘LT’ - less than
- ‘LE’ - less than or equal to
- ‘EQ’ - equal to
- ‘NE’ - not equal to
- ‘GE’ - greater than or equal to
- ‘GT’ - greater than

The expression must be of the form:

scalar-1 ‘operator’ scalar-2

2. An expression which is a complex expression comparing two sub-expressions and containing the operators:

- ‘AND’ - sub-expression-1 ‘AND’ sub-expression-2
- ‘OR’ - sub-expression-1 ‘OR’ sub-expression-2

For example:

IF (A ‘LT’ B ‘AND’ A ‘LT’ C) L12

7 PROGRAM CONTROL STATEMENTS

3. An expression or sub-expression may also contain the 'NOT' operator. This will have the effect of reversing the TRUE/FALSE value of the expression it is appended to.

For example:

```
IF ('NOT' A 'LT' B) GOTO/PT1
```

4. A geometry identifier. When this kind of expression is evaluated, the value of the expression will be TRUE if the entity is defined (exists) and FALSE if it is not defined.

"**NCL**-statement" is any valid **NCL** statement that does not cause an auto name generation.

The only required command in a block IF structure is the beginning IF statement and the ENDIF statement. The ELSIF or ELSE statements are optional.

The statements contained between THEN and the next ELSE/ELSEIF/ENDIF, or ELSE and ENDIF, are considered a block. The block will be executed when the expression contained in the preceding conditional is true. No more than one block per IF structure will be executed on each entry.

The IF THEN statement begins a block IF construct. The block following it is executed if the IF THEN conditional is true.

The ELSEIF THEN statement is an optional statement within a block IF construct. The block following it is executed only when all of the previous IF THEN and ELSEIF THEN conditionals are false, and its conditional is true.

The ELSE statement is also an optional statement. The block following it will be executed only when all of the previous IF THEN and ELSEIF THEN conditionals are false. There can be only one ELSE statement per block IF structure.

The ENDIF statement terminates the block IF construct. DO loops are allowed within the IF construct, but they must be wholly contained within a single block of the IF structure. They cannot cross the ELSE/ELSEIF and ENDIF boundary.

IF-THEN-ELSE statements are only permitted within a macro or looping region and may be nested.

A JUMPTO to a label inside an IF-THEN-ELSE range is not permitted. A JUMPTO from within an IF-THEN-ELSE range to a label outside the range is

7 PROGRAM CONTROL STATEMENTS

permitted if this label is in the current looping region or not in a calling MACRO's looping range.

The IF-THEN-ELSEIF/ELSE-ENDIF structure can be nested and it can be nested up to 100 levels.

7.7 IFTOL

The IFTOL statement allows the programmer to set the accuracy used when comparing two values in an IF statement. The syntax is as follows:

IFTOL/tol

Where tol specifies the allowable range of deviation between the two numbers. If the two numbers are within this range of each other they are considered to be equivalent.

For example:

```
A      =1.01
B      =1.0
IFTOL /.01
LOOPST
IF      (A' EQ 'B) JUMPTO/ID1
```

In the above example "A" would be considered equal to "B" because the two scalars are within the amount specified by the IFTOL statement.

7.8 INCLUD

The INCLUD statement allows the programmer to include the contents of another file into the part program.

The syntax for the INCLUD statement is:

INCLUD/file-name [, ON]

This statement will cause all the lines in the file name to be included into the part program. The lines will only be there during that run of **NCL** and will not be written into the part program if it is saved either at the end of the session or by using the *SAVEPP command. Included lines may be changed while running **NCL**.

interactively but the user should note that any changes made will not be saved. INCLUD statements within included files are allowed.

The statements from the INCLUDED file will not be printed in the first part of the batch listing unless the optional word "ON" is added after the "file-name."

7.9

INSERT

The string of alphanumeric characters following the INSERT statement are passed directly to the output NC program exactly as they appear. There is no checking done either by **NCL** or the postprocessor to determine if the data is correct for a particular machine/control combination.

The INSERT command provides a way to output data which is either not supported by the postprocessor or that is difficult to achieve using standard postprocessor commands.

For example:

```
INSERT G0 G40 G80 G49 Z0
```

would be output to the NC program exactly as shown.

The value of a scalar variable or the character string of a text variable may be passed to the postprocessor in an INSERT statement. To do this, place an "@" before the scalar/text variable's name in the INSERT statement. The text in the INSERT statement will be written to the CLfile and any scalar/text variable preceded by an "@" will have its value/string substituted in place of the variable's name.

For example, the statements:

```
DIA=1.25
CR=.125
CRT=FORMAT ("%4.3f", CR)
INSERT Load Tool with @DIA Dia and @CRT Corner Rad
```

will insert the following data in the CLfile:

```
Load Tool with 1.2500 Dia and .125 Corner Rad
```

The **FORMAT** statement can be used to control the number of decimal places output to the CLfile for a scalar variable.

Note: INSERT is different from the ***INSERT** command.

7 PROGRAM CONTROL STATEMENTS

7.10 JUMPTO

The JUMPTO statement causes an unconditional transfer of control to occur. A JUMPTO statement may only appear within a loop or macro. The valid syntax construct for the JUMPTO statement is:

```
JUMPTO/label
```

This syntax causes control to be passed to the statement specified by label. This statement must be contained within the current loop or macro.

Example JUMPTO Statements:

```
JUMPTO/10  
JUMPTO/L1
```

7.11 Logical-expression

A "logical-expression" can be one of the following:

1. An expression comparing scalar values using the operators:

'LT' - less than
'LE' - less than or equal to
'EQ' - equal to
'NE' - not equal to
'GE' - greater than or equal to
'GT' - greater than

The expression must be of the form:

```
scalar-1 'operator' scalar-2
```

2. An expression which is a complex expression comparing two sub-expressions and containing the operators:

'AND' - sub-expression-1 'AND' sub-expression-2
'OR' - sub-expression-1 'OR' sub-expression-2

For example:

```
IF (A 'LT' B 'AND' A 'LT' C) L12
```

7 PROGRAM CONTROL STATEMENTS

3. An expression or sub-expression may also contain the 'NOT' operator. This will have the effect of reversing the TRUE/FALSE value of the expression it is appended to.

For example:

```
IF ('NOT' A 'LT' B) GOTO/PT1
```

4. A geometry identifier. When this kind of expression is evaluated, the value of the expression will be TRUE if the entity is defined (exists) and FALSE if it is not defined. This type of logical expression can only be used inside a logical IF statement.

A logical expression can be used anywhere an arithmetic expression can appear. The logical expression will be evaluated giving a +1 result for a TRUE evaluation and a -1 result for a FALSE evaluation. This type of usage can even appear in statements outside of MACROs and LOOPS.

For example:

```
A=(1 'GT' 2)
```

would set "A" to -1 since (1 'GT' 2) is FALSE.

And the statement:

```
B=(A-3 'LE' 4)
```

would set "B" to 1 (TRUE) if "A" were 7 or less.

Note, however, that the "existence of an entity" type logical expression can only be used in a Logical IF statement.

For example:

```
B=(PT3)
```

will produce an error. To set B to a value based on the existence of PT3, the following could be done.

```
IF(PT3) S10          B=-1
```

```
B=-1          IF(PT3) B=1
```

```
JUMPTO/S20      or
```

```
S10:B=1
```

```
S20:
```

7 PROGRAM CONTROL STATEMENTS

7.12 LOOPND

The LOOPND statement is used to terminate a loop. The valid syntax construct for the LOOPND statement is:

```
LOOPND
```

7.13 LOOPST

The LOOPST statement is used to define the start of a loop. The loop extends to the following LOOPND statement. A LOOPST statement may not appear within a macro or within a loop. The valid syntax construct for the LOOPST statement is:

```
LOOPST
```

This statement notifies **NCL** that a loop is being defined. All subsequent statements are read by **NCL** and written to the part program file but not processed. When a LOOPND statement is found, control is transferred back to the statement following the LOOPST statement, and normal processing occurs.

The following example shows a loop:

```
LOOPST
I=0
L1 : IF (I=10) L2 , L2 , L3
L2 : I=I+1
JUMPTO/L1
L3 : LOOPND
```

At the end of this loop, the value of I will be 11.

7.14 MACRO

A MACRO is a group of **NCL** statements that can be processed later in a program using the CALL statement. The group of statements can be CALLED any number of times. MACROs can, and usually do, have variables associated with them. The variables are used in place of actual numbers, names, and vocabulary words inside the MACRO. The variables are then given values when the MACRO is CALLED. Thus each CALL to a MACRO can produce different results. The MACRO feature of **NCL** is a very powerful tool for creating standard methods for machining and defining geometry. Once a functional MACRO (or method) is created it can become a standard tool when producing **NCL** programs that require the

7 PROGRAM CONTROL STATEMENTS

functionality of that MACRO. It is, in effect, the user's way to create his own routines.

The MACRO statement signifies the start of this sequence of programming statements. The TERMAC statement signifies the end of the sequence. The statements in a macro are processed using the CALL statement. The valid construct for a MACRO statement is:

```
macro-name=MACRO [ /variable [=default-value] ]  
[ ,variable [=default-value] , . . . ]
```

The macro-name may be any name you wish to assign the macro; it is its identifier. The rules for naming a macro are the same as those that apply to the naming of geometry except that macro-name cannot be a subscripted identifier.

A macro variable is used to assign a name, value, or **NCL** vocabulary word to any variable used within the macro. Where a variable has not been given a default value, it must be assigned a value in the CALL statement. Values in a CALL statement override the default value in the MACRO statement for that call only.

A MACRO variable may be used anywhere in the statements within the MACRO. The default or assigned values assigned to the parameters will be substituted when the statements using the parameters are processed.

MACRO statements:

```
M1=MACRO/A,B
```

associated CALL statement CALL/M1,A=3,B=7

```
M2=MACRO/C=3,D=7
```

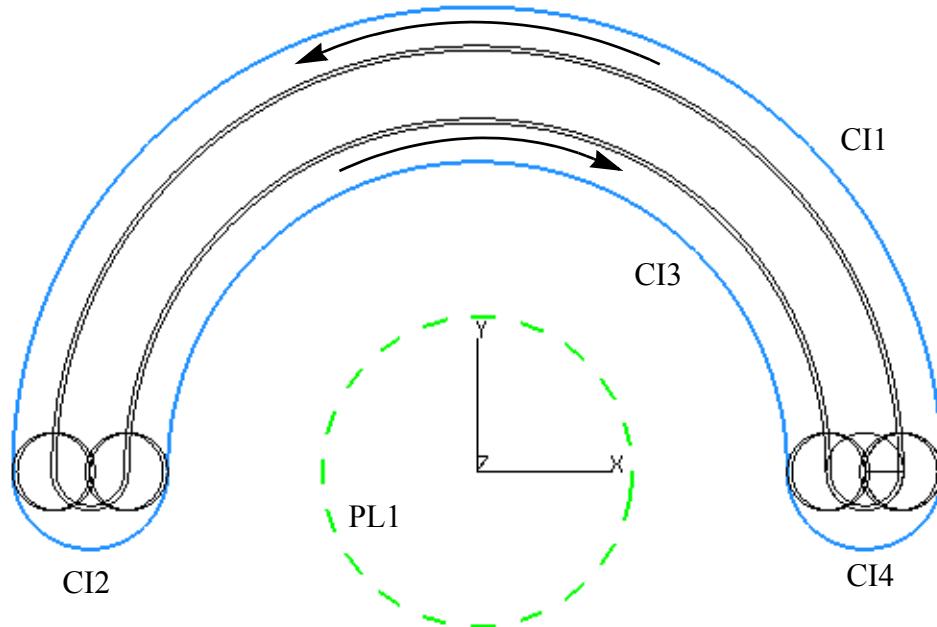
In the example that follows, a simple single variable macro is used to perform a rough and finish cut on a curved slot.

```
FROM   /STPT  
GD    /PL1,10  
$$  
M1    =MACRO/TH1  
TH    /0,TH1  
IV    /1,0,0  
PSIS  /PL1  
GO    /CI1,20
```

7 PROGRAM CONTROL STATEMENTS

```
TL, GL /CI1
GF    /CI2
GF    /CI3
GF    /CI4
GF    /CI4, TT, CI1
TERMAC
$$
CALL  /M1, TH1=.03 $$ + .03 FOR FINISH CUT
CALL  /M1, TH1=0
RP, GT /STPT
```

The following illustration shows the resultant cutter path.



Note: Macros may be redefined when CANON is ON.

7.14.1 Local Identifiers

Local identifiers are allowed to define inside the Macro definition. See Chapter 2 [Local Identifiers](#) section for details.

7 PROGRAM CONTROL STATEMENTS

7.15 **ON/ERROR,THEN, CONTIN[, WARN] STOP NOWARN label**

This command allows the user to control the program flow when a “generation” type error occurs (non-syntax related).

Where:

ERROR	Required for proper syntax.
THEN	Required for proper syntax.
CONTIN	NCL will continue processing after encountering an error. This is the default setting while in batch mode.
STOP	NCL will stop processing after an error is encountered. This is the default setting while in interactive mode. STOP and CONTIN will behave the same way while in batch mode (except for the output of error/warning messages).
label	Specifies a label to jump to after encountering an error.
WARN	Outputs the message associated with the error. This message will be output as an error when STOP is specified or as a warning when CONTIN or “label” is specified. WARN is the default setting.
NOWARN	Does not output the error message. NOWARN may not be specified with STOP.

This command is only valid within loops and macros. It is also local to the current executing macro and will not stay in effect when calling nested macros and loops. It will be restored to its current state after the nested macro/loop returns. This feature also allows all nested macros/loops to have their own local ON/ERROR condition.

This command will automatically cancel itself after an error is encountered. If the user wishes it to remain in effect, then this command will have to be reissued.

Messages output as warnings will more than likely not be seen while in interactive mode, due to the error line also being used as a status line.

7 PROGRAM CONTROL STATEMENTS

7.16 PPRINT

The PPRINT statement's contents are passed on to **NCL**'s output files (the APT source file and the CL file) and thus to the postprocessor. Many postprocessors, such as **PostWorks**, will output the text of a PPRINT statement as a comment in the final NC program. For example:

```
PPRINT Load Tool #1 - 1.0'' Diameter Ball End Mill
```

Will be output as a comment as follows:

```
(Load Tool #1 - 1.0" Diameter Ball End Mill)
```

Most postprocessors require that the DISPLAY/ON command be issued before PPRINTs are passed as comments to the NC program.

The value of a scalar variable or the character string in a text variable may be passed to the postprocessor in a PPRINT statement. To do this, place an "@" before the scalar/text variable's name in the PPRINT statement. The text in this PPRINT statement will be written to the CLfile and any scalar/text variable preceded by an "@" will have its value/string substituted in place of the variable's name.

For example, the statements:

```
DIA=1.25  
CR=.125  
CRT=FORMAT ("%4.3f", CR)  
PPRINT Load Tool with @DIA Dia and @CRT Corner Rad
```

will insert the following data in the CLfile:

```
Load Tool with 1.2500 Dia and .125 Corner Rad
```

The **FORMAT** statement can be used to control the number of decimal places output to the CLfile for a scalar variable.

7.17 PROMPT

There are four different forms of PROMPT command. The first form is used to allow class and description text to be attached to a scalar variable. The second form can be used to prompt the user for a scalar value or a geometric label while in

*RUN mode. The other two forms can be used within a macro definition to customize the appearance of the [Dynamic Macro form](#).

7.17.1 PROMPT To Allow Class And Description Text Attached To A Scalar Variable

The form of PROMPT command can be input anywhere after the corresponding scalar variable has been defined. The syntax for this form of PROMPT command is:

```
PROMPT/SCALAR, name [, class], description
```

Where:

name	Is the name of the scalar variable that has been defined.
class	Defines the class of the scalar variable (name) that has been defined. It can be a text string (must be enclosed by a pair of double quotes) or a text variable up to 20 characters in length. The class name is not case sensitive. If no class name is specified, then the string "Default" is assumed.
description	A text string (must be enclosed in a pair of double quotes) or a text variable up to 64 characters in length and contains the description of the scalar variable (name) that has been defined.

7.17.2 PROMPT In *RUN Mode

This form of PROMPT command can be input anywhere (except inside a macro) in an **NCL** part program and is used to prompt the user for a scalar value or for the name of a geometric entity. This is convenient when a single part program is used to program a family of parts with varying input dimensions. The syntax for this form of PROMPT command is:

```
PROMPT/name [, defvalue], prompt_txt[, min, max]  
defstring                                                 geotype
```

Where:

name	Is the name of the scalar or geometry to be defined.
------	--

7 PROGRAM CONTROL STATEMENTS

defvalue	Is the default value that name will be given if the user does not enter a response to the prompt. The entry will be shown on the prompt input line as being the default. This entry is also used in batch mode to assign a value to name since no user input is possible. An error message will be generated in batch mode if a default value/string is not specified.
defstring	Is the default string that name will be given if the user does not enter a response to the prompt. This can be a text variable or a text string (must be enclosed in a pair of double quotes). The entry will be shown on the prompt input line as being the default. This entry is also used in batch mode to assign a value to name since no user input is possible. An error message will be generated in batch mode if a default value/string is not specified.
prompt_txt	Defines the prompt which appears on the screen when the PROMPT command is encountered during an interactive session. This can be a text string (must be enclosed in a pair of double quotes) or a text variable up to 40 characters in length.
min,max	These optional parameters can be used to define the minimum and maximum values which the user can specify when defining a scalar.
geotype	Can be set to any valid geometry type (POINT, LINE, CIRCLE, etc.). When this parameter is set to a geometry type then name and "defstring" must be an alphanumeric string. This format is used for assigning a new name to an existing entity. This is useful when a family of parts type program requires certain entities to have a specific name.

Example:

```
PARTNO EXAMPLE PROMPT COMMAND
$$
LOADU /IMPELLER
$$
PROMPT/ANG,30,"Enter angle between each blade",   $
      5,45
```

7 PROGRAM CONTROL STATEMENTS

```
NBLADES=360/ANG  
PROMPT/PSRF, "SF1", "Enter name of pressure surf", SF
```

In the above example the following prompts will appear on the command line when this program is run interactively.

```
ENTER ANGLE BETWEEN EACH BLADE[Range:5.0000-45.0000]$  
[Def:30]
```

If the user enters a Carriage Return at the above prompt, then the scalar, ANG will be assigned a value of 30. If the user enters a number between 5 and 45, then the scalar ANG will be assigned the value entered. If the user enters a number that is less than 5 or greater than 45 an error message will be generated and the prompt will be reissued.

```
ENTER NAME OF PRESSURE SURF[Def:SF1]
```

If the user enters a Carriage Return at the above prompt, then the default name SF1 will be used to define PSRF. If SF1 does not exist an error will be generated.

With no overriding value or label specified, the result of the two PROMPT commands would be the same as if the following statements were entered:

```
*ANG=30  
*PSRF=S/ SF1
```

7.17.3 PROMPT To Customize A Dynamic Macro Form

This form of the PROMPT command is used to customize the macro input form when a macro is called dynamically via the screen menu (**Macros > Macro Call**). The syntax for this form of PROMPT command is:

```
PROMPT/name, [length,precision,]prompt_text $  
[ ,SUBSTR,RETAIN] [ ,NOW ] [,min,max ]  
LABEL      NEXT    word-list  
NUM        geo-list
```

Where:

name	Is the name of the macro variable to which the prompt will apply.
------	---

7 PROGRAM CONTROL STATEMENTS

length	This optional parameter is useful when a numeric (real or integer) value is expected. This sets the total length of the input field. By default this parameter is set to 9.
precision	This optional parameter is useful when a numeric (real or integer) value is expected. This specifies the number of digits to the right of the decimal point. Precision can be set to 0 to represent an integer. By default this parameter is set to 4.
prompt_text	This will be the prompt that appears in the dynamic macro form when a macro is called via the screen menu (Macros > Macro Call). This can be a text string (must be enclosed in a pair of double quotes) or a text variable up to 40 characters in length.
SUBSTR	This parameter determines how an entity that contains a subscript is output to the text field when it is picked. RETAIN Will place both the label and subscript in the text field. LABEL Will place only the label in the text field. NUM Will place only the subscript value in the text field.
NOW	When this parameter is specified for a Macro Argument and pick mode is entered for this argument's field on the Dynamic Macro Call form, then the Macro will be called immediately after selecting an entity from the screen as long as all other fields have a value. If all Macro Argument fields are not filled in, then NCL will not call the Macro and will process the field traversal normally. NCL will remain in picking mode after calling the Macro until either the <i>DONE</i> or <i>REJECT</i> button is hit. If <i>DONE</i> is used to terminate picking, then NCL will return to the Dynamic Macro Call form.

7 PROGRAM CONTROL STATEMENTS

NEXT	When this parameter is specified for a Macro Argument and pick mode is entered for this argument's field on the Dynamic Macro Call form, then the Macro will not be called automatically after selecting an entity, but will process the field traversal normally.
min,max	These optional parameters can be used to define the minimum and maximum input range for numeric input. The presence of these parameters cause numeric input to be expected.
word-list	A list of acceptable vocabulary words that can be entered for this prompt. The presence of word-list establishes this prompt as a toggle field in the dynamic input form. Each word in the list will represent one of the toggle choices. This list is limited to 20 words and the words must be valid NCL vocabulary words (i.e. XLARGE, YLARGE, TLLFT, TO, GOFWD, etc.).
geo-list	A list of acceptable geometry types that can be picked for this prompt. This is only valid if the push button for this prompt is utilized.

An [example](#) of this form of prompt appears after the next section.

7.17.4 PROMPT To Allow A Description Of A Macro

This form of prompt embedded within a macro definition will allow the user to display a description of the current macro along with the macro name. The syntax for this form of PROMPT command is:

```
PROMPT/name, [class,] [OUT ,] [DEFALT ,] $  
        OMIT      RETAIN  
  
        description
```

Where:

name Is the name of the macro currently being defined.

7 PROGRAM CONTROL STATEMENTS

class	Defines the class of the current macro being defined. It can be a text string (must be enclosed by a pair of double quotes) or a text variable up to 20 characters in length. The class name is not case sensitive. If no class name is specified, then the string “Default” is assumed. The currently defined macro will not be displayed in the Dynamic Macro list form if a class of “NONE” is specified.
OUT	Specifies that the CALL/macro statement generated by the Dynamic Macro Call form will be output to the part program file. This is the default state.
OMIT	Specifies that the macro will be processed and no CALL/macro statement will be output to the part program.
DEFALT	Displays the default values for all the macro parameters when the Dynamic Macro Call form is initially displayed. This is the default state.
RETAIN	Causes the data entered in the Dynamic Macro Call form to be saved and used as the default data the next time that this same Macro is called.
description	A text string (must be enclosed in a pair of double quotes) or a text variable up to 40 characters in length and contains the description of macro currently defined.

Note: Specifies a null description string (““) to specify class only.

An example of this form of prompt appears in the next section.

7.18

Dynamic Macro Calling

An **NCL** macro can be called by simply selecting the desired macro from a menu of currently defined macros. When the macro is called, the macro variables are presented to the user in an input form. The input form can be customized using the PROMPT command as described in the previous section. Variables can be set using text input, graphic pick of an entity, toggle choices, or by accepting the default value as defined in the macro definition.

7 PROGRAM CONTROL STATEMENTS

The following is an example of defining a macro using the PROMPT command and calling it dynamically.

1. Assume a program containing the following macro:

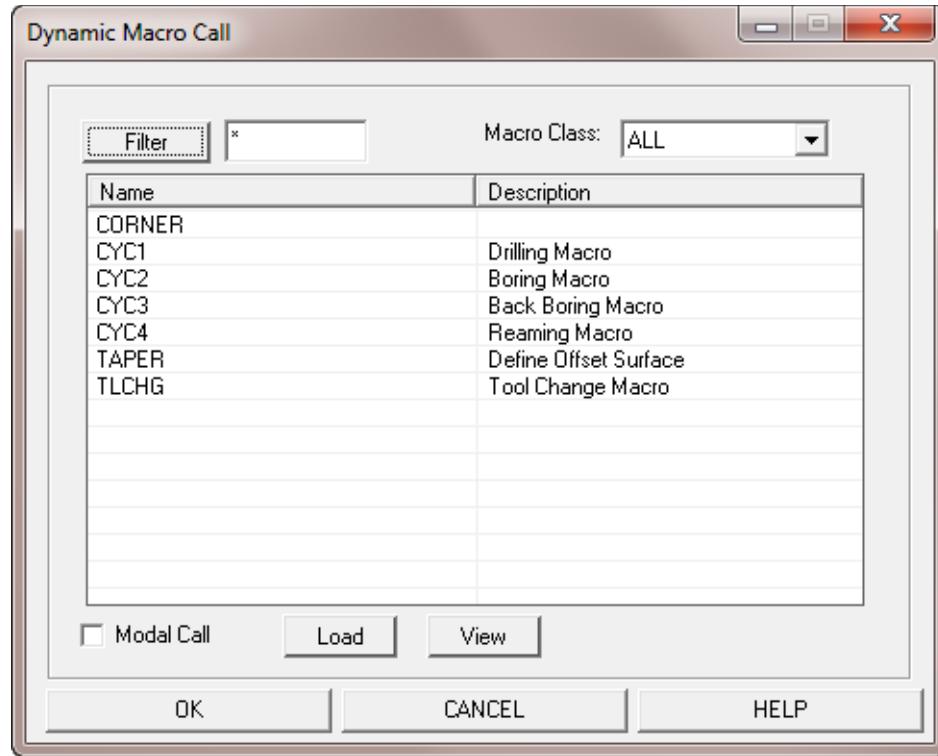
```
$$  
$$ Macro to define a tapering offset surface  
$$  
TAPER =MACRO/NPTS=20,ISF,OSF,MOD=YL,SO,EO  
PROMPT /TAPER,"Geometry",OUT,DEFALT,  
        "Define Offset Surface"  
PROMPT /NPTS,3,0,"No. of PTS in defining CV",  
        20,200  
PROMPT /ISF,"Name of input SF", SF  
PROMPT /OSF,"Name of output SF"  
PROMPT /MOD,"Offset direction modifier",  
        XLARGE,XSMALL,YSMALL,YLARGE,  
        ZLARGE,ZSMALL  
PROMPT /SO,4,2,"Starting offset value",.01,1  
...  
...  
TERMAC
```

2. Run the part program containing the macro interactively.
3. To call the macro select the following menu in the top of the screen:

MACROS > MACRO CALL

A form as shown in next page will appear. If there is any macro defined up to this moment, it will appear in this form. The **Load** button might also be used to load any macro defined in an external macro file. After the **Load** button is clicked, a window file browser will open to allow moving around the directory and load the required macro files. **NCL** will not put a copy of the macro files into the part program, instead an **INCLUD** statement with the full path file name will be output to the part program. Also the macro list will be updated to include the loaded macros.

7 PROGRAM CONTROL STATEMENTS

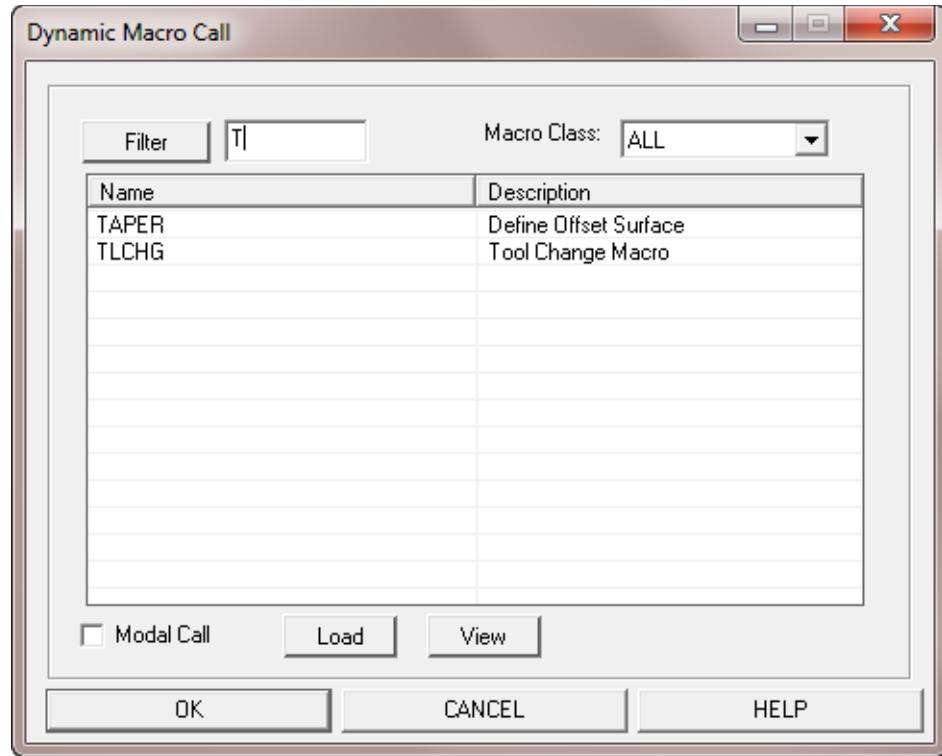


The description of the macro will be displayed in the list area along with the macro if it had been defined.

The user can also use the **Filter** text box to reduce the number of macros displayed in the list area.

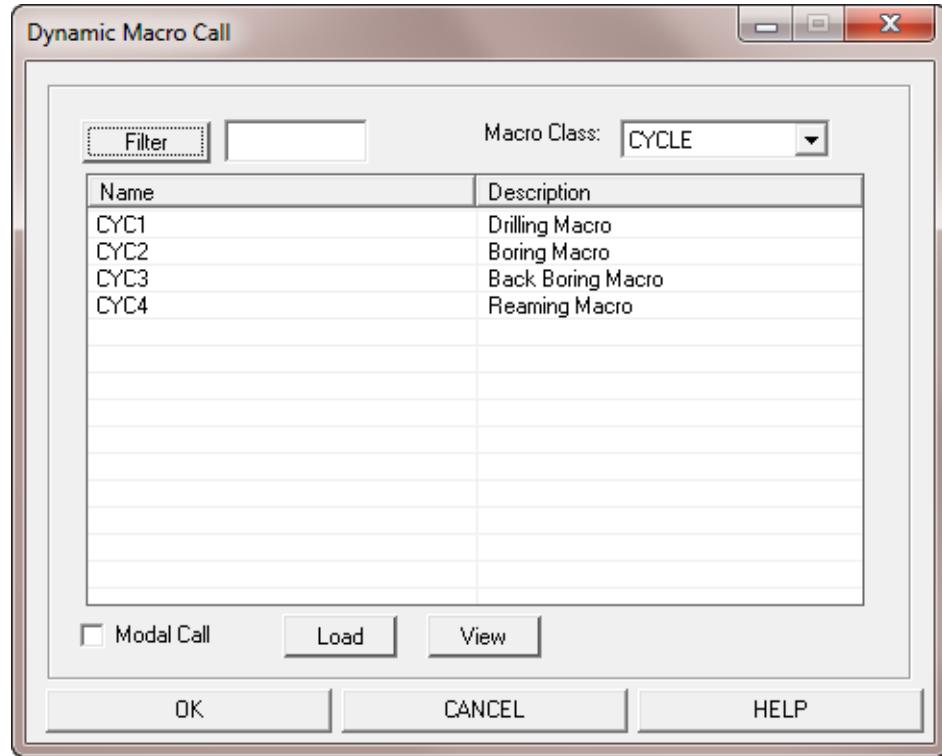
For example: If “t” is entered into the filter box, only the macros names with the first character equal to “t” will be displayed in the list area.

7 PROGRAM CONTROL STATEMENTS



The **Macro Class** is a toggle button which lists all the available class (Except the class of NONE which will never be displayed in this form.) of macros defined up to this point. The user can use the **Macro Class** toggle button to reduce the number of macros displayed in the list area. For example: if there is a class of CYCLE defined and the toggle button is toggled to this field, only the macros specified as CYCLE class will be displayed in the list area.

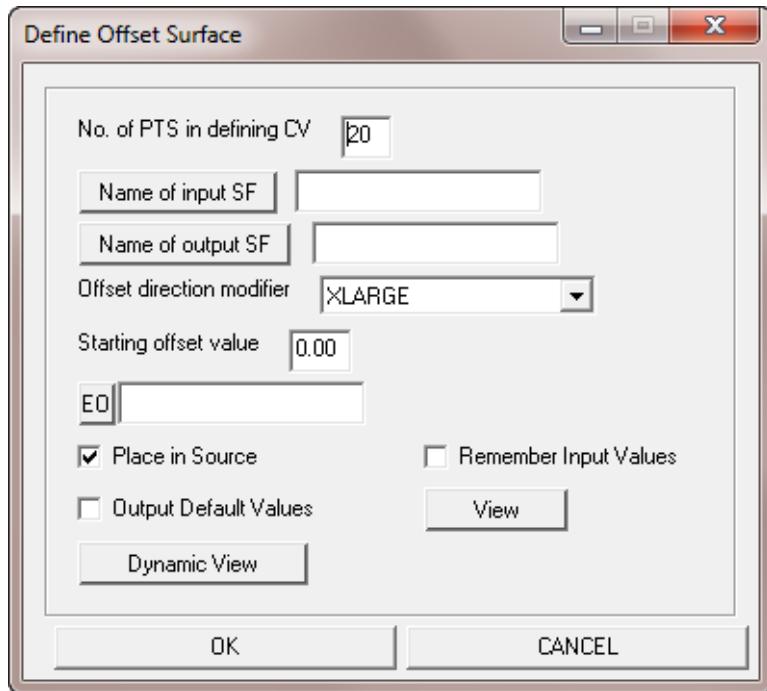
7 PROGRAM CONTROL STATEMENTS



The user can also click the View button to view a drawing or a graphic file (BMP, JPEG and GIF) creating previously. The drawing or graphic file must has the same name as the corresponding macro with a file extension of either ".dwg", ".bmp", ".jpg" or ".gif".

The **Modal Call** button should be pressed if the user wants to call the same macros several times in a roll. Otherwise, this macro call window will closed immediately after the call is finished.

4. Select the macro "TAPER" by highlighting it and then click the **OK** button.
The following form will appear:



Note:

- The default value of 20 is shown for the variable NPTS. The size of the input text field is for three digits of numerals.
- No length/precision or min/max restrictions were placed on ISF or OSF. Therefore, the user, can either place the cursor in the empty text field and enter the required information, or click the corresponding named button to change to pick mode and select an entity from the graphic area. The name of the selected entity would be used as input to the current field.
- The “Name of Input SF” button will only allow the user to pick surface type entity since a geometry list is used to restrict the allowable geometry type in the ISF prompt command.
- Dynamic Macro Call will not allow the use of "by locate" mode even after changing from Pick Mode to By Locate Mode after clicking the named button and the expression (PT/x,y) will be entered into the text field with x and y as the coordinate value of where the cursor is when the mouse button is clicked. **NCL** will generate error if the By Locate Mode is used.
- The input field for the variable MOD is a toggle field.
- If a number outside the range .01 to 1 is entered for the variable SO an error message would be generated. Also, the text field will show with a size of four digits.

7 PROGRAM CONTROL STATEMENTS

- If a non-numeric value were entered for NPTS or SO an error message would be generated.
- Because no PROMPT command was given for the variable EO the variable name appears at the prompt, the user, can either place the cursor in the empty text field and enter the required information, or could click the named button to change into pick mode and select an entity. The name of the selected entity would be used as input to the current field.
- The total number of prompted arguments can be displayed on one page of form depends on the screen resolution. If the number is more than what can be displayed in a page, the form will display a vertically scroll bar on the right side of the form and the user can use this scroll bar to see and access all the prompted arguments.

Clicking the Dynamic View button will close the form and activate the dynamic viewing mode. The form comes back after the dynamic viewing mode cancelled.

Upon entering the information click the OK button. The macro will be called. The macro call statement will be written to the part program if the Place in Source button is indented, otherwise, no macro call statement will be written to the part program. If the Output Default Value button is not indented, the passing arguments with defaulted values defined and not changed during the call will not be output to the part program. If the Remember Input Values button is indented, the data entered in the Dynamic Macro Call form will be saved and used as the default data the next time this same Macro is called.

7.19

READ

The READ statement allows the programmer to make a copy of the contents of another file into the part program.

The syntax for the READ statement is:

READ/ description, file-name

Where:

description This is a required field for syntax. This parameter can be any character or string of characters except the delimiter symbol ",". For APT compatibility, a number "2" should be entered in this description field.

file-name The name of the file to be read. The filename can be proceeded with path name.

7 PROGRAM CONTROL STATEMENTS

This statement will cause all the lines in the named file to be copied into the part program. This is different from the INCLUD statement for which a copy of the included file contents will not be put into the part program.

A nested **READ** statement within an **INCLUD** file will be treated the same as an **INCLUD** statement, meaning that the included file from the nested **READ** statement will not be saved as part of the original file or a copy of it will be put inside the part program.

The **READ** statement will be replaced with a **\$\$READ/2,file-name** statement in the part program after the processing of this statement.

7.20

REMARK

The **REMARK** statement allows the programmer to put a single line comment in the part program. The contents of the statement, other than the word **REMARK**, is ignored by **NCL**. Comments may also be added to any **NCL** statement by putting them after a single dollar sign (\$) or a double dollar sign (\$\$). The **REMARK** and **TITLES** statements are both processed the same way.

Example **REMARK** statement:

```
REMARK THIS IS A REMARK STATEMENT
```

7.20.1

REMARK/ ON OFF

This set of **REMARK** statements allows the programmer to put multi-line comments in the part program. **NCL** ignores all statements between the **REMARK/ON** and **REMARK/OFF** statements and treats them as comments. This can also be used to block off a section of the program by putting the section between the **REMARK/ON** and **REMARK/OFF** statements.

Example:

```
REMARK/ON
1st line of comment statement
2nd line of comment statements
...
last line of comment statements
REMARK/OFF
```

7 PROGRAM CONTROL STATEMENTS

7.21 **SYN**

The SYN statement allows the programmer to define synonyms for **NCL** vocabulary words within the part program (This is opposed to the use of the nclvoc.syn file). The syntax for the SYN statement is:

`SYN/new_word,old_word[,new_word,old_word[,...]]`

scalar		scalar
--------	--	--------

Where `new_word` is the user defined word (synonym) for the existing word `old_word`. A scalar can be used instead of an `old_word`, the scalar can be any positive integer or an integer value corresponding to a regular ***NCL*** vocabulary word. See Appendix: Vocabulary for the corresponding integer value.

For Example:

SYN/PK, POCKET, CN, CLONE, GA, 873

This method uses the same rules as the nclvoc.syn file with the following exceptions:

1. It is invalid to redefine an existing vocabulary word.
 2. A maximum of 50 synonyms may be defined within a single SYN statements
 3. Multiple SYN statements are allowed. The total number of existing vocabulary words, synonyms and user defined new words cannot be more than 1024.

Note: Output of ASCII APT file from the part program will be affected by the utilization of user defined synonyms. If the alphabetical order of the user defined synonym is lower than the original vocabulary word, the synonym will be output to the ASCII APT file, not the original vocabulary word. This applies only to the ACSII APT file and user defined synonyms, not the binary “.cl” file or the **NCL** default synonyms.

7.22 TERMAC

The TERMAC statement defines the end of a macro. The valid syntax construct for the TERMAC statement is:

TERMAC

7.23 TITLES

The TITLES statement allows the programmer to have comments in the part program. The contents of the statement, other than the word TITLES, is ignored by **NCL**. Comments may also be added to any **NCL** statement by putting them after a single dollar sign (\$) or a double dollar sign (\$\$). The **REMARK** and TITLES statements are both processed the same way.

Example TITLES and comment statements:

```
TITLES THIS STATEMENT WILL NOT AFFECT NCL OUTPUT
PT23=POINT/INTOF, LN1, LN3    $$ CORNER
$$                                INTERSECTION POINT
TOPSF=SURF/CV2A, VE2A,
CV3A, VE3A,          $ LEADING EDGE
CV4A, VE4A          $ MID-CURVE
                      $$ TRAILING EDGE
```

Note that neither the REMARK or TITLES statement supports text enclosed in parenthesis.

7.24 UNDO

The UNDO statement may only be used within a DO loop. It will cause **NCL** to exit the current **DO** loop in the same manner as if the loop counter had reached the limit. If a nested loop was being executed, the UNDO command will cause that loop to be exited but execution will continue for the outer DO loop.

For example:

DO/10, INX=A, B	DO/10, INX=A, B
IF (INX-11)L1, L1, L2	IF (INX 'GT' 11) UNDO
L2:UNDO	.
L1:CONTIN	OR
.	.
10:	10:

In the last example, if the value of INX ever exceeds 11, the UNDO statement will be executed causing the loop to end. The next statement executed will be the one after label 10:, and INX will have its last value.

7 PROGRAM CONTROL STATEMENTS

7.25 Restoring The Current Program In The Event Of An Abnormal Termination Of *NCL*.

In the event of a power failure or an unordered termination of ***NCL***, the current part program can be restored simply by loading “~#_file.ext” in the current working directory into an ***NCL*** session. Where “#” is a number indicating the last file saved and “file.ext” is the actual name of the part program file loaded. If a part program file is not loaded, then file.ext will be “~n_partpgm.ncl”.

Note: If exits ***NCL*** without saving after a temporary work file “~#_file.ext” is loaded/recovered any changes that occurred before the temporary work file “~#_file.ext” was created will be lost since the temporary work file “~#_file.ext” would be deleted immediately after exiting ***NCL***.

8

GRAPHIC DISPLAY CONTROL STATEMENTS

NCL will automatically display geometry and tool motion as they are created during an interactive session.

DRAFT commands can be placed into the **NCL** part program to modify the way geometry and tool motion are displayed.

The **DISPLAY (VISIBL)** and **ERASE (INVIS)** commands can be used to display and erase entities individually or in groups.

The ***SET/ADISPL** command can be used to specify the attributes of how subsequently defined curves and surfaces will be displayed. By default curves (including conics and circles) and surfaces are displayed according to a default tolerance of .001 inches (.025 MM). Surfaces are displayed as a 5x5 grid of iso-parametric lines (also referred to as U and V lines.)

The **CUTTER/DISPLAY** command can be used to modify the graphical representation of the cutter. See the **CUTTER** statement for details.

Tool motion generated while in a **DNTCUT** mode will not be displayed. When a **CUT** statement is encountered the cutter will be displayed at its current location.

By default curves (including conics and circles) and surfaces are displayed according to a default tolerance of .001 inches (.025 MM). Surfaces are displayed as a 5x5 grid of iso-parametric lines (also referred to as U and V lines.) These default settings can be changed using the ***SET/ADISPL** command.

8.1

Color

NCL supports 16 standard colors and a maximum of 48 user customized colors defined in the ncl_color.mod file.

Following is a list of the 16 standard colors with the corresponding color-label, the color-value and the RGB values.

<u>Color</u>	<u>Color-label</u>	<u>Color-value</u>	<u>R</u>	<u>G</u>	<u>B</u>
Default	DEFALT	-1	-	-	-
Black	BLACK	0	0	0	0
White	WHITE	1	255	255	255
Blue	BLUE	2	0	0	255
Red	RED	3	255	0	0

8 GRAPHIC DISPLAY CONTROL STATEMENTS

Green	GREEN	4	0	255	0
Magenta	MAGNTA	5	255	0	255
Yellow	YELLOW	6	255	255	0
Cyan	CYAN	7	0	255	255
Brown	BROWN	8	184	134	11
Light Tan	LTTAN	9	210	180	140
Light Blue	LTBLUE	10	173	216	230
Sea Green	SEAGRN	11	84	255	159
Orange	ORANGE	12	255	165	0
Pink	PINK	13	255	195	203
Purple	PURPLE	14	221	160	221
Grey	GREY	15	192	192	192
User Defined Color	color_label	16-63	0-255	0-255	0-255

8.2 Draft Commands

The DRAFT statement is used to define the screen graphic display and plotting variables to **NCL**.

8.2.1 Set Geometry Default Color

```
DRAFT/DEFALT, POINT , COLOR=color  
LINE  
CIRCLE  
PLANE  
VECTOR  
PNTVEC  
MATRIX  
PATERN  
SHAPE  
CURVE [ , subtype ]  
SURF
```

Where:

color This can be a value in the range of 0-63 or any valid color definition.

subtype This specifies the type of curve or surface and can have any one of the value as shown below:

8 GRAPHIC DISPLAY CONTROL STATEMENTS

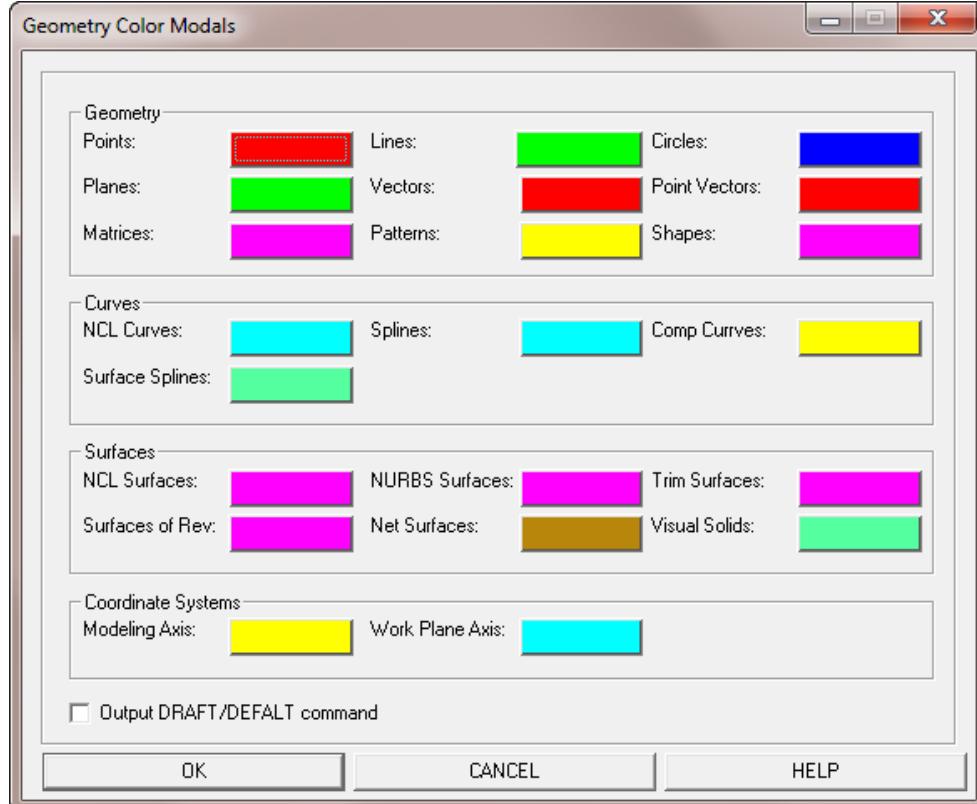
Type	Value
NCL Curve	0
Spline	1
Composite Curve	2
Surface Spline	4
NCL Surface	0
NSURF	1
Trimmed Surface	2
Surface of Revolution	3
Net Surface	4

Example:

```
DRAFT/DEFAULT, POINT, COLOR=PURPLE  
DRAFT/DEFAULT, SURF, 2, COLOR=YELLOW
```

It should be noted that while this command can be entered as shown, it can also be produced from the graphical interactive interface as shown below by using the following on screen menu sequences:

Defaults > Geometry Color Modals



8 GRAPHIC DISPLAY CONTROL STATEMENTS

8.2.2 Set Axes Default Color

```
DRAFT/DEFALT,MAXIS=color  
WAXIS
```

Where:

MAXIS	Set the Modeling Axes.
WAXIS	Set the Working Plane Axes
color	This can be a value in the range of 0-63 or any valid color definition.

8.2.3 To Control Attributes

```
DRAFT/MODIFY [=geo_list] [,COLOR=color] $  
[PEN=pn] [,LINTYP=SOLID (1)] [,LAYER=ln] $  
DASH (2)  
DOTTED(3)  
CENTER(4)  
PHANTM(5)  
DASHLN(6)  
DASHDT(7)  
DASHSP(8)  
[,LINWGT=STD (1)] [,SHADE=ON ] [,TRANS=n]  
MEDIUM(2) OFF  
HEAVY (3)  
EXHVV (4)  
[,MATERL=m ] [,EDGE=OFF ] [,MARKER=typ]  
name ecolor
```

Where:

geo_list	Can include a list of entity names, geometry types (i.e. POINT , LINE , CIRCLE , etc.), "THRU" constructions, or "ALL." if "geo_list" is not present then the attribute settings will apply for all subsequently defined geometry.
----------	---

8 GRAPHIC DISPLAY CONTROL STATEMENTS

color	This can be a value in the range of 0-63 or any valid color definition,
pn	Is the logical pen number attribute (1-256).
ln	Is the layer number (0-9999)
()	Number shown in parenthesis can be used in place of their corresponding minor word.

The following parameters only work with Surface type entities.

SHADE	Turns on or turns off surface shading.
n	Is the translucent value, between 1 (totally transparent) and 100 (non-transparent).
m	Number representing material in the range of 1-16.
name	Name of material, either as a text variable or quoted text string.
OFF	Turns off the surface edge display.
ecolor	Turns on the edge display and specifies the color. It can be a value in the range of 0-63 or any valid color definition, including DEFALT which uses the color of the surface as the edge color.

The following parameters only work with Points type entities (Points and Point-Pattern).

MARKER	Modify the point entities display type.
typ	Specify the marker type. Valid marker types are DOT(1), PLUS(2), STAR(3), CIRCLE(4), CROSS(5), TRIAN(6), DIMOND(7), SQUARE(8), DBLCIR(9), LRGDOT(10) and CUBE(11).

Example:

```
DRAFT/MODIFY=SF1, SF2, COLOR=RED, LINTYP=DASH, LAYER=5  
DRAFT/MODIFY=CI, LN, COLOR=6, LINTYP=0, LINWGT=HEAVY
```

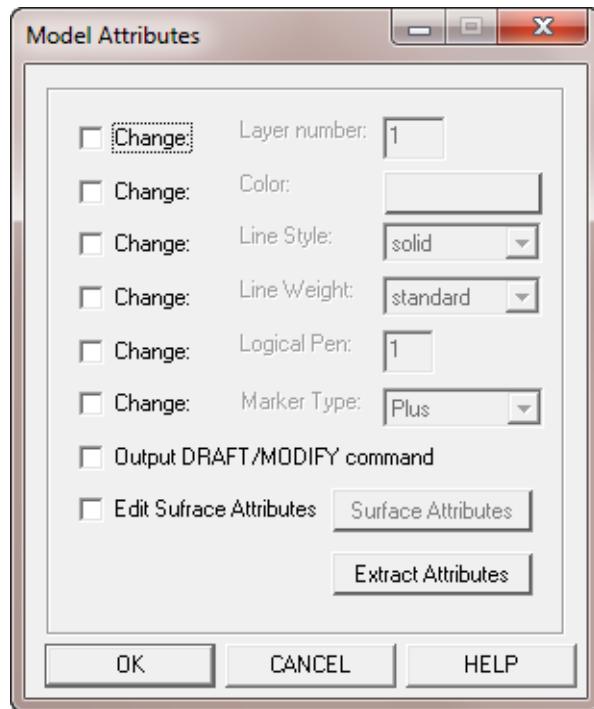
8 GRAPHIC DISPLAY CONTROL STATEMENTS

It should be noted that while this command can be entered as shown, it can also be produced from the graphical interactive interface as shown on next page by using any one of the following on screen menu sequences:

Edit > Mod Attribs

or

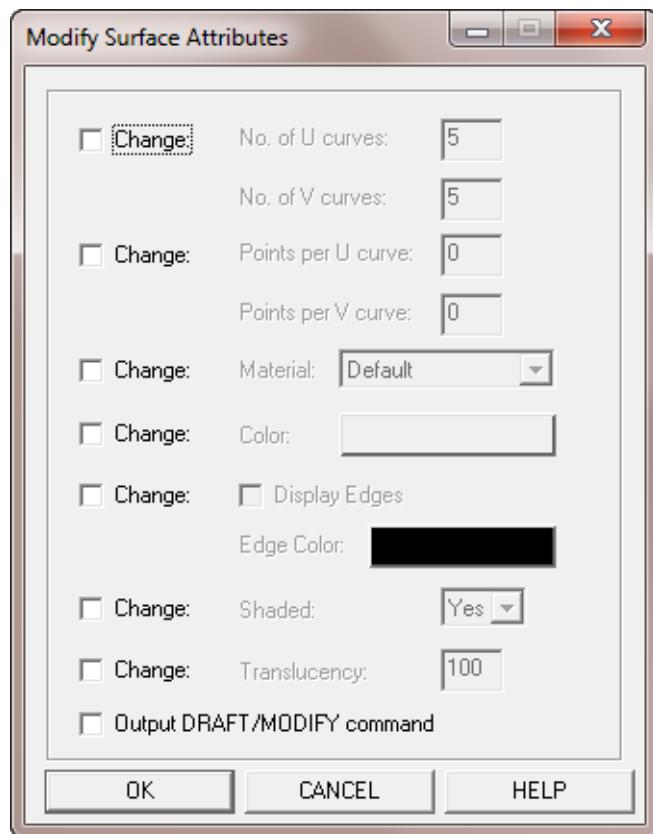
Defaults > Set Attribs



The surface parameters portion of this command can also be produced from the graphical interactive interface as shown on next page by using the following on screen menu sequences:

Edit > Surface > Mod Attrib

8 GRAPHIC DISPLAY CONTROL STATEMENTS



8 GRAPHIC DISPLAY CONTROL STATEMENTS

8.2.4 Modify Viewing Parameters

```
DRAFT/FORMAT=SINGLE , NAME=drw_view $  
      SIDE          Front  
      STACK         Back  
      FOUR          Top  
      FIVONE        Bottom  
      SIX           Left  
                  Right  
                  Isometric  
                  Left Iso  
                  Dimetric  
                  Left Dimetric  
  
[ , BORDER=ON ] [ , AXIS=ON ] [ , NAMED=ON ] $  
      OFF          OFF          OFF  
  
[ , SIZE=ON ] [ , MOTION=ON ] [ , MODE=WIRE ]  
      OFF          OFF          BOTH  
                  SHADE  
                  HIDDEN
```

Example:

```
DRAFT/FORMAT=STACKED, NAME=Front, Isometric, $  
      BORDER=ON, AXIS=ON, NAMED=ON, SIZE=ON, $  
      MOTION=ON, MODE=BOTH
```

The defaults for the attribute parameters are as follows:

```
BORDER=ON, AXIS=OFF, NAMED=OFF, SIZE=OFF, MOTION=ON
```

8.2.5 Selectively Modify Viewports

```
DRAFT/NAME=drw_view[ , BORDER=ON ] [ , AXIS=ON ]      $
    Front          OFF          OFF
    Back
    Top
    Bottom
    Left
    Right
    Isometric
    Left Iso
    Dimetric
    Left Dimetric

[ , NAMED=ON ] [ , SIZE=ON ] [ , MOTION=ON ]      $
    OFF          OFF          OFF

[ , MODE=WIRE   ]
    BOTH
    SHADE
    HIDDEN
```

If an attribute is not specified then it will remain in its current state.

8.2.6 Modify Or Create Views

```
DRAFT/VIEW=name, TRACUT
    REFSYS
    MATRIX=mx
    TOOL
    PARAMS [ [ , CENTER=pt ] [ , NORMAL=ve ]      $
            [ , YAXIS=ve ] [ , SCALE=n ] ]
```

Where:

- | | |
|--------|---|
| name | Is the name of a view to be modified or created. If the specified view already exists (displayed or not) it will be modified, otherwise a new view will be created. |
| TRACUT | Specifies that the view will correspond to the current TRACUT matrix. |

8 GRAPHIC DISPLAY CONTROL STATEMENTS

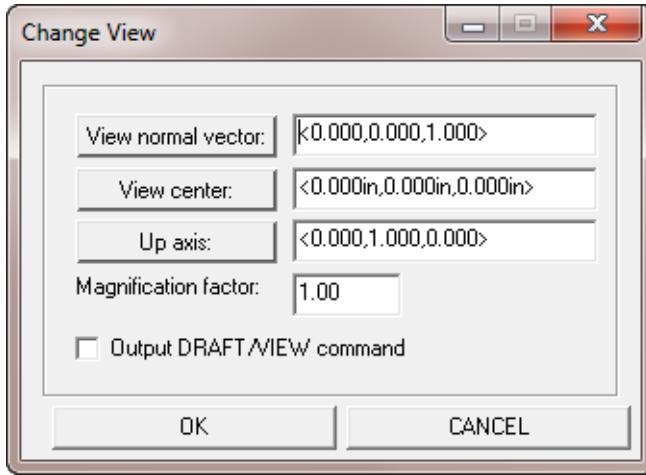
REFSYS	Specifies that the view will correspond to the current REFSYS matrix.
MATRIX	Specifies that the view will correspond to the matrix specified by mx.
TOOL	Specifies that the center of the view will be the current tool end point and the view normal will be the current tool axis.
CENTER	Are valid only with the PARAMS option. Default to
NORMAL	last specified condition if not specified.
YAXIS	
SCALE	
pt	Is a point representing the center of the view. The point can be specified as a valid point name or as X, Y, Z values.
ve	Is a vector representing the view normal (the vector you are looking down). The vector can be specified as a valid vector name or as I, J, K values.
n	Is a positive real number used to specify the view scale.
NOTE:	If “X, Y, Z” or “I, J, K” is specified instead of the point or vector label name, they must be specified in a non-MODSYS or a non-REFSYS condition.

Example:

```
DRAFT/VIEW=Front , PARAMS , CENTER=PT1 , NORMAL=VNY ,      $  
          YAXIS=VPZ  
DRAFT/VIEW=Top , PARAMS , SCALE=1.5  
DRAFT/VIEW=Front , TRACUT  
DRAFT/VIEW=View1 , TOOL
```

It should be noted that while the “DRAFT/VIEW=name,PRARMS,...” command for modification of the view can be entered as shown, it can also be produced from the graphical interactive interface as shown below by using the following on screen menu sequences:

View > Static > Change



8.2.7 Perform An Extreme Zoom

```
DRAFT/FIT,ALL           [, SAME] [, OMIT=omit_list]  
VIEW=name_list
```

Where:

- | | |
|-----------|---|
| ALL | Specifies that all current viewports will be fitted. |
| VIEW | Allows specific views to be fitted. |
| name_list | List of displayed views (view names or view numbers) to fit, separate by comma. |
| SAME | NCL calculates the “fit” box for each view in the command, select the view with the smallest scale and setting all named view to the parameters in the “smallest scale” view port. |
| OMIT | Allows specific views to be excluded from this command (mostly for use with the ALL parameter). |
| omit_list | List of displayed views (view names or view numbers) to be excluded from this command. |

8 GRAPHIC DISPLAY CONTROL STATEMENTS

8.2.8 Repaint The Screen Or A View

```
DRAFT/REDRAW, ALL  
VIEW=name  
n
```

Where:

- | | |
|------|---|
| ALL | Specifies that all current viewports will be redrawn. |
| VIEW | Allows a specific view to be drawn. |
| name | Is the name of a displayed view to redraw. |
| n | Is the number of the viewport to redraw. |

8.2.9 Reset Specified Views

```
DRAFT/NAME=name_list,RESET
```

Where:

- | | |
|-----------|---|
| name_list | Is a list of valid view names that will be reset to their default settings. |
|-----------|---|

8.2.10 Control Tool Motion Display

```
DRAFT/CUTTER [ , STEP=ns ] [ , COLOR=mc , cc ] $  
[ , SHANK , COLOR=sc ] [ , HOLDER , COLOR=hc ] $  
[ , LINTYP=mline_type] $  
[ , RAPID [ , COLOR=rc ] [ , LINTYP=rline_type] ] $  
[ , TRACUT=ON ] [ , TRANS=n ]  
OFF
```

Where:

8 GRAPHIC DISPLAY CONTROL STATEMENTS

ns	A cutter instance will be drawn each time the number of steps (calculated CL points) specified by "ns" has been reached.
mc	Is the color used to display the tool motion center line.
cc	Is the color used to display the cutter.
sc	Is the color used to display the shank.
hc	Is the color used to display the holder.
mline_type	Is the line style used to display the tool motion center line.
rc	Is the color used to display the tool motion center line of RAPID moves.
TRACUT	When set to ON will display the tool motion relative to the current TRACUT matrix (if any).
TRANS=n	Is the translucency for the shaded display of the cutter, shank, holder or the tool assembly. "n" can be any integer value between 1 and 100.
Valid line types are:	SOLID(1), DASH(2), DOTTED(3), CENTER(4), PHANTM(5), DASHLN(6), DASHDT(7), DASHSP(8).

Example:

```
DRAFT/CUTTER, STEP=10, COLOR=WHITE, YELLOW, RAPID,      $  
COLOR=RED
```

Note:

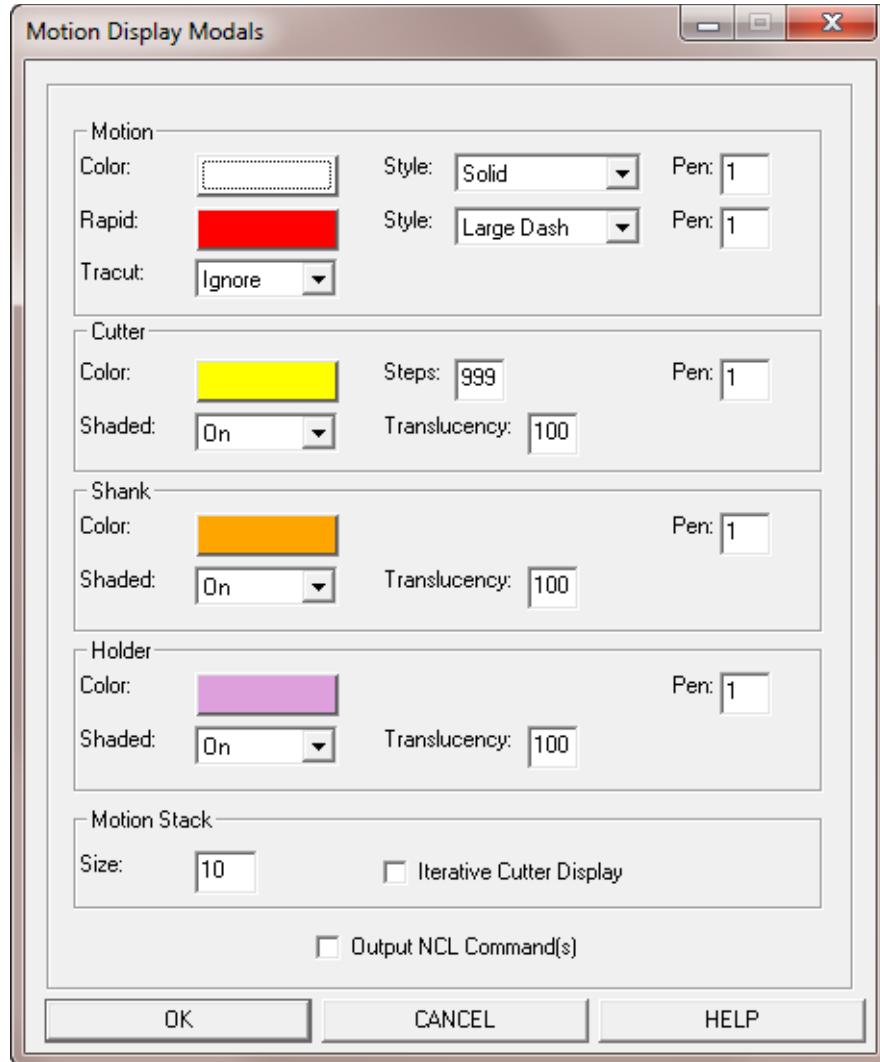
- With overlay plane "ON" and the cutter is displayed as "Wireframe", this statement has no effect on the cutter display color. The cutter will always displays in white color. However, this statement will display the cutter in the specified color during "Dynamic Viewing" activity.
- With overlay plane "OFF" and the cutter is displayed as "Wireframe", this statement will display the cutter image with the specified color.
- With overlay plane "ON", shading "ON" and the cutter is display as "SHADED", this statement will display the cutter, shank, and/or holder image with the specified color.

8 GRAPHIC DISPLAY CONTROL STATEMENTS

- This command allows you to change the color of the cutter, shank and holder displayed within **NCL**. The color of the cutter, shank and holder in **NCL/IPV** is controlled by the **NCL/IPV** Tool Modal and Edit Tool List forms.

It should be noted that while this command can be entered as shown, it can also be produced from the graphical interactive interface as shown on next page by using the following on screen menu sequences:

Playback > Modals



8.2.11 Label Display Properties

```
DRAFT/LABEL [ ,ON ] [ ,BOX,ON ] [COLOR,txt,bkd]      $  
          OFF           OFF  
  
          [ ,SIZE,wd,hgt] [ ,OFFSET,dist]             $  
  
          [ ,LEADER [ ,ON ] [ ,COLOR,lead] [ ,ARROW,ON ] ]  
          OFF           OFF
```

Where:

ON	Specifies that entity labels will automatically be displayed for the geometry types specified by "geo-type-list" as they are defined. This will have no effect on previously defined entities.
OFF	Specifies that entity labels will not automatically be displayed for the geometry types specified by "geo-type-list" as they are defined. This will have no effect on previously defined entities.
BOX,ON	Specifies that the labels will display in a box with a solid background.
BOX,OFF	Specified that the labels will not be displayed in a box.
COLOR,txt,bkg	Specifies the label text color and the box background color. "txt" is the text color. and "bkg" is the background color.
SIZE,wd,hgt	Specifies the label font size width and height in pixels. "wd" is the width value and "hgt" is the height value.
OFFSET,dist	Specifies the minimum distance between labels in pixels to avoid overlap.
LEADER,ON	Specifies that leader lines will be automatically displayed for all entities as they are defined or altered. This will have no effect on previously defined entities.
LEADER,OFF	Specifies that leader lines will not be displayed for all entities as they are defined or altered. This will have no effect on previously defined entities.

8 GRAPHIC DISPLAY CONTROL STATEMENTS

COLOR,lead Specifies the leader line color. “leader” is the leader line color.

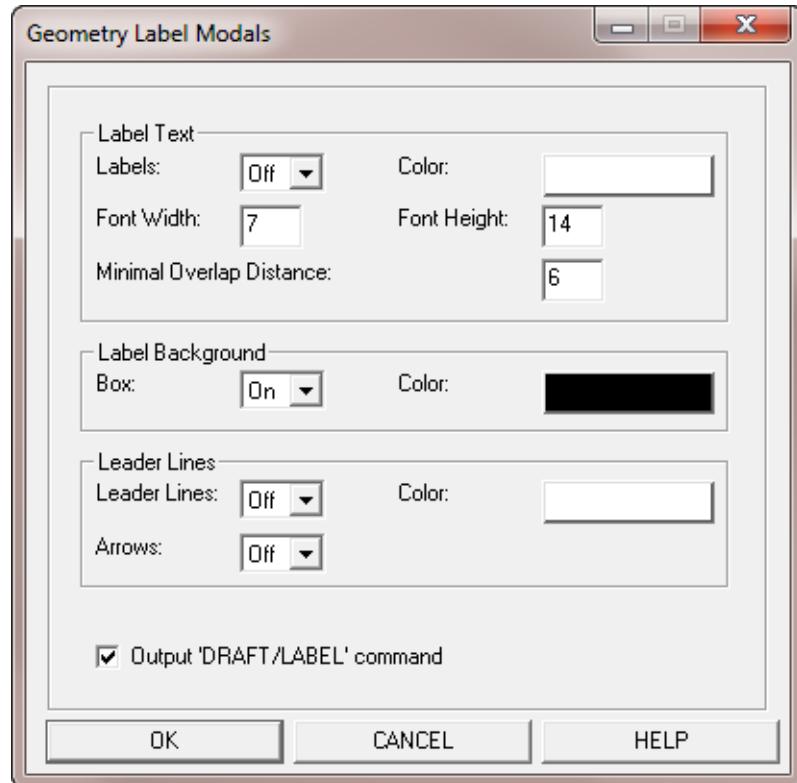
ARROW,ON Specifies the arrow heads should be displayed with all leader lines.

ARROW,OFF Specifies no arrow heads will be displayed with leader lines.

“BOX” and “Box Background Color” do not apply to MATRIX entity type. Matrix labels always display without a box and background color.

It should be noted that while this command can be entered as shown, it can also be produced from the graphical interactive interface as shown below by using the following on screen menu sequences:

Defaults > Label Modals



8.2.12 Label Display Control During Geometry Creation

DRAFT/LABEL,ON [,geo-type-list] [,LEADER,ON]
OFF OFF

Where:

ON	Specifies that entity labels will automatically be displayed for the geometry types specified by "geo-type-list" as they are defined. This will have no effect on previously defined entities. If "geo-type-list" is not specified then all entity will be affected.
OFF	Specifies that entity labels will not automatically be displayed for the geometry types specified by "geo-type-list" as they are defined. This will have no effect on previously defined entities. If "geo-type-list" is not specified then all entity types will be affected. OFF is the default setting for entity labels.
geo_list	A list of geometry types (i.e. PT, LN, CI, etc.) for which to apply the ON or OFF setting of the automatic entity label display.
LEADER,ON	Specifies that entity leader lines will be automatically displayed for the geometry types specified by "geo-type-list" as they are defined. This will have no effect on previously defined entities. If "geo-type-list" is not specified then all entity will be affected.
LEADER,OFF	Specifies that entity leader lines will not be automatically displayed for the geometry types specified by "geo-type-list" as they are defined. This will have no effect on previously defined entities. If "geo-type-list" is not specified then all entity will be affected.

8 GRAPHIC DISPLAY CONTROL STATEMENTS

8.2.13 Labels ON And OFF For Previously Defined Geometries

```
DRAFT/LABEL,ON ,MODIFY=geo_list[,LEADER,ON ]  
OFF OFF
```

Where:

ON	Specifies that entity labels will be displayed for the geometry items listed in "geo-list".
OFF	Specifies that entity labels currently being displayed will be turned off for the geometry items listed in "geo-list".
geo_list	Can include a list of entity names, geometry types (i.e. PT, LN, CI, etc.), "THRU" constructions, or "ALL."
LEADER,ON	Specifies that leader lines will be displayed for the geometry items listed in "geo-list".
LEADER,OFF	Specifies that leader line currently being displayed will be turned off for the geometry items listed in "geo-list".

Examples:

```
DRAFT/LABEL,ON,MODIFY=ALL
```

Will turn labels on for all existing entities.

```
DRAFT/LABEL,ON,MODIFY=LN1,LN2,PT,CI
```

Will turn labels on for LN1, LN2 and all existing points and circles.

```
DRAFT/LABEL,ON,MODIFY=PKPTS(1,THRU,PKPTS(NUM(PKPTS)))
```

Will turn labels on for the entire point array PKPTS.

8.2.14 Location Of Geometry Labels

```
DRAFT/LABEL,ALTER,geo[,POINT,]pt          $  
                                VECTOR, vec  
  
[,LEADER,ON[,pt]]]  
OFF
```

Where:

geo Is the name of a geometric entity whose label location will be altered.

pt Is the name of an existing point, point-vector, X, Y, Z values for a point at which the label will be displayed.

vec Is the name of an existing vector, point-vector, or the I, J, K values of a vector by which to offset the current label location. Both the vector's magnitude and direction are considered. The minor word VECTOR must be present when "vec" is specified.

If no point or vector is specified then the label for "geo" will be displayed at its default location.

LEADER,ON[,pt] Specifies leader lines will be displayed for the entity whose label location is being altered. "pt" is an optional point or X,Y,Z values of a point on the entity at which the leader line will point to. If no point is specified then the leader line will point to the default label location.

LEADER,OFF Specifies leader lines will not be displayed for the entity whose label location is being altered.

Example:

```
DRAFT/LABEL,ALTER,SF1,PT5  
DRAFT/LABEL,ALTER,SF1,VECTOR,.5,.5,0
```

8 GRAPHIC DISPLAY CONTROL STATEMENTS

8.3 DISPLAY Statement:

The DISPLAY statement is used to graphically display geometric entities and tool path data. The syntax for the various forms of the DISPLAY statement follow:

Example DISPLAY Statements:

```
DISPLAY/PT23
DISPLAY/LN2
DISPLAY/VE
DISPLAY/CV1,100
DISPLAY/CV1,0
DISPLAY/PT,LN,CI,SF1,SF2
DISPLAY/PKPTS(1,THRU,20)
DISPLAY/SF,15,15
DISPLAY/SF1,30,3,50,5
DISPLAY/SF,0,3,0,3
DISPLAY/ALL
DISPLAY/CU
DISPLAY/LAYER,4,THRU,8
```

8.3.1

DISPLAY/curve-name [,number-of-points]

DISPLAY/surface-name [, number-of-points, number-of-u-lns, \$
number-of-points, number-of-v-lns]

DISPLAY/surface-name [, number-of-u-lns, number-of-v-lns]

By default the accuracy of the display of curves and surfaces are determined by the current setting of the *SET/ADISPL,TOLER command.

The default accuracy is .001 inches (.025 MM.) In other words a curve displayed on the screen will be accurate within a .001 inches chordal tolerance of the mathematical curve. Likewise, the iso-parametric lines used to display a surface will be accurate to within a .001 inches chordal tolerance of the surface.

The above commands provide methods for changing the default method used to display curves and surfaces.

Curve-name is the name of a curve or b-spline to be displayed. The optional number-of-points is a scalar value indicating the number of points used to draw the curve. This is used to control the accuracy of the display and has no affect on the mathematical representation of the curve. Once used to display a curve the number-of-points becomes an attribute of the curve and is stored in the unibase.

The attribute can be changed by specifying a new value for number-of-points or by specifying a 0 for number-of-points. If a 0 is specified then the current value of ***SET/ADISPL,TOLER** will be used. If number-of-points is omitted the curve will be displayed using its existing attributes (if it had previously been displayed using the number-of-points option) or the current setting of ***SET/ADISPL,TOLER**.

Surface-name is the name of surface to be displayed.

The optional number-of-points is a scalar value specifying the number of points used to display the corresponding iso-parametric lines. The number-of-u-lns is a scalar value used to specify the number of iso-parametric lines to display in the U direction of the surface. The number-of-v-lns specifies the number of iso-parametric lines to display in the V direction of the surface. The U and V direction of a surface are based on how the surface was originally defined. Once used to display a surface the number-of-points for both U and V becomes an attribute of the surface and is stored in the unibase. The attribute can be changed by specifying a new value for number-of-points or by specifying a 0 for number-of-points. If a 0 is specified then the current value of ***SET/ADISPL,TOLER** will be used. If number-of-points is omitted the surface will be displayed using its existing attributes (if it had previously been displayed using the number-of-points option) or the current setting of ***SET/ADISPL,TOLER**.

If only the number-of-u-lns and number-of-v-lns is specified then the display accuracy of the iso-parametric lines will be determined by the current value of ***SET/ADISPL,TOLER**.

8.3.2 DISPLAY/ plane, AT, point

Will display a defined plane at a specific point. The point will be at the center of the dashed circle that is drawn to indicate the plane's position. If the point does not lie on the plane, it will be projected normal to the plane and the projected point will be used as the dashed circle's center.

8.3.3 DISPLAY/ geometry-list

This construct will display all entities specified by geometry-list. geometry-list is a comma separated list of entity names and/or entity types. Entity types can include one or more of the following: **ANOTE, POINT, LINE, CIRCLE, PLANE, VECTOR, POINT-VECTOR, CURVE, SOLID, SURFACE, MATRIX**. The THRU construct is also supported for subscripted arrays and entities named using the default **NCL** naming convention.

8 GRAPHIC DISPLAY CONTROL STATEMENTS

In addition, the CV and SF entity types may be followed by the specification for number-of-points for curves and number-of-points, number-of-u-lns, number-of-points, number-of-v-lns or number-of-u-lns, number-of-v-lns for surfaces (as explained earlier in this section.)

Example DISPLAY commands using this format:

```
DISPLAY/SF1,CI,CV,100  
DISPLAY/SF,30,3,30,3,PT,LN  
DISPLAY/PKPTS(1,THRU,20),CI1,THRU,CI10,LN1,PT
```

8.3.4 DISPLAY/ALL

This construct will display all entities.

8.3.5 DISPLAY/ CUTTER

This construct will display the cutter at its current location.

8.3.6 DISPLAY/ matrix-name [, ax, boxx [, boxy, boxz]]

The displayed matrix appears as three vectors corresponding to the matrix axes and the wire frame of a parallelepiped (box). The default line style is dotted. The colors of a matrix will change depending upon how the matrix is being used. The following colors are used for various conditions:

default	-	magenta
REFSYS	-	cyan
MODSYS	-	red
TRACUT	-	green

With reference to the syntax in the DISPLAY/ matrix-name statement:

- ax - Is the axis length used to draw all matrix axis arrows (default length is 1 inch or 25 mm).
- boxx - Is the length of the edge parallel to the x-axis used to draw the box (the default length is .75 inches or 18.75 mm).
- boxy - Is the optional length of the edge parallel to the y-axis used to draw the box (boxx will be used if boxy is not specified).

8 GRAPHIC DISPLAY CONTROL STATEMENTS

boxz - Is the optional length of the edge parallel to the z-axis used to draw the box (boxx will be used if boxz is not specified).

By default a matrix is not displayed. The following control commands may be used to turn on or off the automatic display of matrices:

```
*SET/DISPLAY,MX  
*RESET/DISPLAY,MX
```

8.3.7 DISPLAY/ LAYER, n1, n2, . . . [, THRU, nn] [, nm]

This construct causes the selected combination of layer numbers (0 thru 9999) to be displayed.

Example:

```
DISPLAY/LAYER,1,4,6,thru,10,100
```

8.3.8 DISPLAY/ MAXIS WAXIS

Displays model axis (MAXIS) or the work plane axis (WAXIS).

8.4 ERASE Statement

The ERASE statement is used to erase (make invisible) geometric entities and tool path data. The syntax for the various forms of the ERASE statement follows:

Example ERASE Statements:

```
ERASE/PT23  
ERASE/LN2  
ERASE/VE  
ERASE/CV1  
ERASE/PT, LN, CI, SF1, SF2  
ERASE/PKPTS(1, THRU, 20)  
ERASE/LN1, CI1, THRU, CI10, PT  
ERASE/ALL  
ERASE/MOTION  
ERASE/GEO
```

The valid syntax for the ERASE statement is:

8 GRAPHIC DISPLAY CONTROL STATEMENTS

8.4.1 ERASE/ ALL

This construct causes all geometry that has been displayed to be erased.

8.4.2 ERASE/ geometry-list

This construct will erase all entities specified by geometry-list. Geometry-list is a comma separated list of entity names and/or entity types. Entity types can include one or more of the following: **ANOTE**, **POINT**, **LINE**, **CIRCLE**, **PLANE**, **VECTOR**, **POINT-VECTOR**, **CURVE**, **SOLID**, **SURFACE**, **MATRIX**. The **THRU** construct is also supported for subscripted arrays and entities named using the default **NCL** naming convention.

8.4.3 ERASE/ LAYER, n1, n2, ... [, THRU, nn] [, nm]

This construct causes the selected combination of layer numbers (0 thru 9999) to be erased.

Example:

```
ERASE/LAYER,10,99,5,THRU,22
```

8.4.4 ERASE/ MAXIS WAXIS

Erases the display of the model axis (MAXIS) or the work plane axis (WAXIS).

8.4.5 ERASE/ MOTION

This construct causes all motion generated graphical data to be erased.

8.5 INVIS Statement

The INVIS statement is an alternate syntax for the **ERASE** statement and works exactly the same way.

8.6 VISIBL Statement

The VISIBL statement is an alternate syntax for the [DISPLAY](#) statement and works exactly the same way.

9 NCL CONTROL COMMANDS

9 NCL CONTROL COMMANDS

NCL control commands are implemented to allow the user to perform various utility functions while running **NCL**. These functions range from showing geometry on the screen to inserting statements into the part program file. Control commands are acted on as received by **NCL** and not written into the part program file (Except as explained in the "##" command section). All control commands may be abbreviated by their first two letters, except where an ambiguity occurs. Control commands are always preceded by the character "*".

9.1 *CONSOL

The CONSOL command causes the processing mode to become CONSOLE INPUT which is the normal default mode. This command is normally used to leave the insert mode of processing. The word CONSOL may be abbreviated as CO.

During interactive session, the mouse cursor can move to the icon menu and click on it to accomplish the same task.

9.2 *DELETE

The DELETE command causes lines to be deleted from the part program file. Care should be taken not to delete lines that have labels or are lines in a macro that has been defined already. The word DELETE may be abbreviated as DE.

*DELETE/scalar-1[, scalar-2]

This construct will cause all lines from the line specified by scalar-1 to and including the line specified by optional scalar-2 to be deleted. If the optional scalar-2 is not given, then only the line specified by scalar-1 will be deleted.

Example DELETE commands:

```
*DELETE/15  
*DE/3,12
```

9.3 *EDIT

The *EDIT command may be used to alter the contents of an **NCL** part program line without having the line processed. Any line in the program file may be edited whether it has been processed or not. Caution should be taken in altering lines that

have already been processed since the next time the program is processed, the altered line may cause different results. The word EDIT may be abbreviated as ED.

9.3.1 *EDIT [/ positive-scalar]

This construct will cause **NCL** to display the statement that is assigned the line number indicated by positive-scalar and allow the programmer to make changes to it. If no positive-scalar is given, the current line will be used.

The line may be changed by typing over existing characters, inserting additional characters to the statement, deleting characters. When all editing of the line has been completed and the "UP arrow" or the "DOWN arrow" key pressed, the line will be put back into the part program file but not processed in any other way by **NCL**.

Example EDIT commands:

```
*EDIT  
*ED/24
```

9.4 *EDT

The *EDT command allows the user to temporarily leave an **NCL** interactive session and enter a text editor for the purpose of editing the current part program file. The text editor used is determined by the setting of the environmental variables `UL_NCL_EDIT` and `UL_NIS_EDIT` which are defined in the `\NCCS\NCL100\interface\ncl.init` or the users `user.init` file.

```
*EDT
```

To return to **NCL**, simply EXIT from the EDITOR.

During interactive session, the mouse cursor can be moved to the "EDT" button and click on it to accomplish the same task.

9 NCL CONTROL COMMANDS

9.5 *FIND And *FINDTK

The *FIND and *FINDTK commands allow the programmer to find either a sequence of characters or a token (i.e. an identifier, a vocabulary word or a special character) in the part program. The syntax is as follows:

```
*FIND/find-sequence  
*FINDTK/token
```

The *FIND command will find the sequence of characters specified wherever they occur in the part program whereas the *FINDTK command will find only characters that form a token.

Example FIND and FINDTK commands:

*FIND/PT	will find both PT and PT1 but
*FINDTK/PT	will find PT but not PT1.

9.6 *INPUT

The *INPUT command causes **NCL** to go into input mode. In input mode, any statements entered are written into the part program file but they are not processed. Any statement or command starting with an "*" will cause **NCL** to leave the input mode. The word INPUT may be abbreviated as INP. The format for the INPUT command is:

```
* INPUT
```

9.7 *INSERT

The *INSERT command causes **NCL** to go into insert mode. In insert mode, any statements written into the part program file are inserted at the current line counter, and all existing statements that follow are moved down in the file. A *RUN command notifies **NCL** to leave insert mode and begin processing the program. The *CONSOL command returns you to the processing mode at the point of the insert and is typically the method used for exiting the insert mode. The word *INSERT may be abbreviated as *INS.

During interactive session, the mouse cursor can be moved to the icon menu and click on it to accomplish the same task.

9.8 *LOADPP

The *LOADPP command allows the programmer to replace the current **NCL** Part Program workfile with another file.

*LOADPP/ [file name]

Note: If the optional file name is not specified, the name of the part program file given when **NCL** was initiated is used.

CAUTION should be taken in using this command for it will erase your present file before performing the replacement. **MAKE SURE YOUR CURRENT FILE IS SAVED FIRST!**

9.9 *PAUSE

The PAUSE command causes **NCL** to go into a temporary stop mode. Normal processing continues when the return key is hit. No commands or statements are accepted other than the *QUIT, *START and *STOP commands when **NCL** is stopped by a *PAUSE command. This command is used mainly to momentarily stop processing after an imbedded show command (*S/) to examine the displayed results. A *PAUSE command is valid only if it is read from the part program file. The word PAUSE may be abbreviated as PA.

*PAUSE

The *PAUSE command will not close the scrolling window after pressing Return to continue.

9.10 *RESET

The RESET command is used to modify **NCL** control options. The word RESET may be abbreviated as RE.

Example RESET Commands:

```
*RESET/ADISPL  
*RESET/APTSRC,CIRCUL  
*RESET/APTSRC,REMARK  
*RESET/AUTOST  
*RESET/APTSRC,VERIFY  
*RESET/CANON
```

9 NCL CONTROL COMMANDS

```
*RESET/CALL, [n]
*RESET/DISPLAY[, geometry list]
*RESET/INDENT
*RESET/MOTION
*RESET/NOWARN
```

Note: The commands *SET/APTSRC and *RESET/APTSRC are not toggle commands that only the last one specified in the part program file will be applied to the generation of the APT file.

The valid syntax constructs for the RESET command are:

9.10.1 *RESET/ ADISPL

This construct is used to turn off the Automatic Display of Geometry on the graphics terminal as each piece of geometry is defined.

9.10.2 *RESET/ APTSRC, CIRCUL

This construct notifies **NCL** to stop outputting APT circular records. All subsequent motion generated by driving a circle will be output as GOTO points. This is the default setting. See the *SET/APTSRC,CIRCUL command for an explanation of APT circular records.

9.10.3 *RESET/ APTSRC, DATA

This construct notifies **NCL** to stop outputting the REMARK program-information line to the start of the APT source file. By default the REMARK program-information line is output to the beginning of the APT file.

9.10.4 *RESET/ APTSRC, IPV

This construct notifies **NCL** to stop outputting the corresponding CL record number to the APT source in front of all the GOTO points.

9.10.5 *RESET/ APTSRC, IPVCOM

This construct notifies **NCL** to stop outputting the PPRINT IPV commands to the APT source file.

9.10.6 *RESET/ APTSRC, VERIFY

This construct notifies **NCL** to stop outputting APT source circular records in the VERIFY format.

9.10.7 *RESET/ APTSRC, REMARK

This construct is used to notify **NCL** to stop outputting **NCL** source statements to the APT source file. By default **NCL** source statements are output to the APT source file as comment lines (preceded by \$\$) prior to the APT motion record the **NCL** source statement is responsible for generating. Using this command will reduce the size of the APT source file.

Example:

```
RAPID
GOTO / 5.9908, -0.9521, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.2972, -0.9524, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -1.0838, -0.1193, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -1.0712, 0.1314, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -1.0328, 0.8804, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.9897, 1.8706, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.9563, 2.8225, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.9396, 3.4038, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.9246, 3.9853, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.0232, 4.7463, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / 5.9691, 3.8717, 1.6667, 0.0000000, 0.0000000, 1.0000000
```

See [*SET/ APTSRC, REMARK](#).

9.10.8 *RESET/ APTSRC, REMARK, ACTIVE

This construct is used to notify **NCL** to stop outputting REMARK statements to the APT source file. This is the default setting.

See [*SET/ APTSRC, REMARK, ACTIVE](#).

9 NCL CONTROL COMMANDS

This construct is used to output or not output a warning message for the Automatic Tool Start feature. "tol" and "angtol" are the positional tolerance and angular tolerance to control this feature. Automatic Tool Start feature will be enabled and no warning message will be output when both the positional difference and the tool axis vector between the current position and the calculated starting position is less than "tol" and "angtol". If either the positional difference between the current position and the calculated starting position is bigger than "tol", or the angular difference between the current tool axis vector and the calculated starting tool axis vector is bigger than "angtol", the Automatic Tool Start feature will not be enabled and a warning message will be output. If "tol" and "angtol" are not specified, the Automatic Tool Start feature will be turned off and always output a warning message.

The optional parameter "OMIT" or "RETAIN" specifies whether to output the theoretical calculated starting point of the move after a warning message was output and the mismatch accepted. If "RETAIN" is specified, the theoretical calculated starting point of the move will be output to the cl file if either the position between the current point and the calculated point is more than "tol", or the angular difference between the current tool axis vector and the calculated starting point tool axis vector is more than "angtol". If "OMIT" is specified, this calculated move will be eliminated. The default condition is "RETAIN".

See *SET/AUTOST.

9.10.10 *RESET/AUTOUV

This construct instructs **NCL** to turn off the automatically use of the "U" and "V" values calculated in the previous motion statement for the current motion statement. For more information, see [chapter 6 Continuous Motion Statements](#).

9.10.11 *RESET/ CANON

This format is used to notify **NCL** to NOT allow the definition of geometry if the specified identifier has already been assigned. It performs the same function as [CANON](#)/ OFF.

9.10.12 *RESET/ CALL [, n]

This construct is used to "back out" of "n" MACRO calls, including any active loops. "n" specifies the number of [MACRO](#) calls to terminate (an internal *TERMAC will be issued for each call). If "n" is not specified, then all currently active MACRO calls and loops will be terminated and [NCL](#) will skip to the line after the last CALL or LOOPND statement. This command will be ignored if the user is currently defining a LOOP or a MACRO, or if there are no LOOPS or CALLS active.

9.10.13 *RESET/ CASE

This construct is used to instruct the text string functions [FINDEX](#), [RINDEX](#), [STRCMP](#) not to be case sensitive.

9.10.14 *RESET/ DISPLAY [, geometry-type-list]

This construct is used to turn off the automatic graphic display of geometric entities. By default all geometric entities (with the exception of matrices) are graphically displayed as they are being defined. Valid entries for the geometry list are [POINT](#), [LINE](#), [CIRCLE](#), [PLANE](#), [VECTOR](#), [POINT-VECTOR](#), [CURVE](#), [SURFACE](#), [MATRIX](#), [PATERN](#), [NSHAPE](#) and [SOLID](#). Of course the synonym for each of these types could also be used. As many geometry types as desired, separated by commas, may be used in a *RESET/DISPLAY command. If no geometry list is given then the automatic graphic display of all entities is stopped. This command is often used inside of a [MACRO](#) which defines construction geometry that the user would prefer not to see. The [*SET/DISPLAY](#) command can be used to turn back on the automatic display of geometry.

Example:

```
*RESET/DISPLAY
*RESET/DISPLAY, PT, VE, PV
*RESET/DISPLAY, LINE, CURVE
```

9.10.15 *RESET/ EXPCL

This construct is used to disable the expanded motion CL file option.

See [*SET/ EXPCL](#) later in this chapter.

9 NCL CONTROL COMMANDS

9.10.16 *RESET/ INDENT

This construct is used to notify **NCL** to set the "ALL" and the "SEP" indent values back to column one (1). See the [*SET/ INDENT](#) command for further explanation on the use of statement indentation.

This command is same as the command

* SET/ INDENT , OFF .

Example:

```
1: *SET/ INDENT, ALL, 5
2:      CA/ON
3:      MULTAX/ON
4:      PT1=POINT/1, 0, 0
5:      *RESET/ INDENT
6:      PT2=POINT/2, 0, 0
7:      PT3=POINT/3, 0, 0
```

9.10.17 *RESET/ MOTION

This construct is used to stop the display of tool motion coordinates to the scrolling text window (when it is open) during an interactive session and to the print file (.pr) created when batch processing.

9.10.18 *RESET/ NOWARN

This construct is used to turn off the NOWARN option, causing **NCL** to stop at each warning.

9.10.19 *RESET/ PAUSE

This construct is used to notify **NCL** to disable the [*PAUSE](#) command.

9.10.20 *RESET/RUNCMD

This construct is used to notify **NCL** to disable the command mode when a stopping condition for [*RUN](#) is met after it was enabled by the [*SET/RUNCMD](#) command.

9.10.21 *RESET/ SCHECK

This construct is used to turn off the automatic correction of the so called "bow-tie effect" when defining ruled surfaces, fit surfaces and 4 curves surfaces through curves that run in opposite directions.

9.10.22 *RESET/ STATLN

This construct is used to notify **NCL** to disable the automatic display of Status Line information on the error message line. This information is normally displayed in the same place error messages are shown whenever there is no error message active.

The line is re-displayed each time any of the information on it changes. This can take a significant amount of time when running a large program interactively so this command can be used to save the time it takes to display this information. See [*SET/ STATLN](#) and [*SHOW/ STATLN](#) for related information.

9.10.23 *RESET/ STOP

This construct is used to notify **NCL** to disable the [*STOP](#) command. Normally the [*STOP](#) command will cause **NCL** to stop processing during an interactive session. This ***RESET/STOP** will cause **NCL** not to stop processing whenever a [*STOP](#) command is encountered in an interactive session.

9.10.24 *RESET/STPCMD

This construct is used to notify **NCL** to disable the command mode when a [*STOP](#) command is executed after it was enabled by the [*SET/STPCMD](#) command.

9.11 *RUN

The RUN command causes **NCL** to enter run mode. The word RUN may be abbreviated as RU.

This command causes **NCL** to begin processing the statements in the part program file. Processing continues until one of the following conditions is encountered: an error, a [*PAUSE](#) command, a [*STOP](#) command, a FINI statement, or the end of the part program file.

9 NCL CONTROL COMMANDS

To process one statement at a time simply enter COMMAND mode and press the ENTER key to process each line.

9.11.1 *RUN/ scalar

This command causes **NCL** to process to but not including the statement line number specified by the scalar.

9.11.2 *RUN/ MACRO

This command causes **NCL** stop running at the first line of the next called macro

9.11.3 *RUN/ TERMAC

This command causes **NCL** stop running at the end of the current or next called macro

9.11.4 *RUN/ LOOPST

This command causes **NCL** stop running at the first line of the next executed loop (as specified by a LOOPST or DO command)

9.11.5 *RUN/ LOOPND

This command causes **NCL** stop running at the end of the current or next executed loop (as specified by a LOOPST or DO command).

9.12 *SAVEPP

The *SAVEPP command allows the programmer to save the current **NCL** part program workfile at any time during an interactive **NCL** session.

*SAVEPP/ [file name]

If the optional file name is not specified, the name of the part program file given when **NCL** was initiated is used.

9.13 *SET

The SET command is used to modify **NCL** control options.

Example SET Commands:

```
*SET/ADISPLY  
*SET/APTSRC, REMARK  
*SET/APTSRC, CIRCUL  
*SET/APTSRC, CUTTER  
*SET/AUTOST  
*SET/CANON  
*SET/DISPLAY, geometry-type-list  
*SET/ELIMIT, scalar  
*SET/geometry-type, scalar  
*SET/INDENT, ALL, scalar  
*SET/VERSION, vflag  
*SET/MOTION  
*SET/NOWARN  
*SET/VER, vflag
```

Note: The commands *SET/APTSRC and *RESET/APTSRC are not toggle commands that only the last one specified in the part program file will be applied to the generation of the APT file.

The valid syntax constructs for the SET command are:

9.13.1 *SET/ ADISPL

The *SET/ADISPL command is used to establish the default method for displaying curves and surfaces. *SET/ADISPL commands can appear anywhere in an **NCL** part program and will only affect subsequently defined entities. The syntax for the various forms of the *SET/ADISPL command follow.

Example *SET/ADISPL commands:

```
*SET/ADISPL, TOLER, .01  
*SET/ADISPL, 3, 3  
*SET/ADISPL, 50, 3, 3  
*SET/ADISPL, 50, 5, 50, 3  
*SET/ADISPL, 100, 30, 3, 10, 2
```

9 NCL CONTROL COMMANDS

9.13.2 *SET/ ADISPL, TOLER, tol

This construct is used to establish the accuracy, based on a chordal tolerance specified by “tol”, at which subsequently defined curves and surfaces will be displayed. The default tolerance is .005 (.127 MM). A larger value of “tol” will result in a less accurate display but better graphics performance.

9.13.3 *SET/ ADISPL, [number-of-points,] number-of-u-lns, number-of-v-lns \$

This construct is an alternate method for establishing the accuracy at which subsequently defined curves will be displayed and the number of U and V iso-parametric lines that will be used to display subsequently defined surfaces. The accuracy of the iso-parametric lines used to display surfaces will be determined by the current setting of *SET/ADISPL,TOLER. If the optional number-of-points is not specified then only surfaces will be affected by this command. See the DISPLAY statement for a complete explanation of number-of-points, number-of-v-lns and number-of-u-lns.

9.13.4 *SET/ ADISPL, [number-of-points,] number-of-points, number-of-u-lns, number-of-points, number-of-v-lns \$

This construct is an alternate method for establishing the accuracy at which subsequently defined curves and surfaces will be displayed.

The command also establishes the number of U and V iso-parametric lines that will be used to display surfaces.

The first specification for number-of-points applies only to curves and if omitted only surfaces will be affected by this command. See the DISPLAY statement for a complete explanation of number-of-points, number-of-v-lns and number-of-u-lns.

9.13.5 *SET/ APTSRC, CIRCUL [, IPV] VERIFY

If *SET/APTSRC,VERIFY is not specified, this construct will cause motion generated by driving a circle to be output to the APT source as an APT circular record in the following format:

GOFWD/ (CIRCLE/ . . .) , ON, (PLANE/ . . .)

This will ensure that circular interpolation will ultimately be output to the NC program.

If the optional parameters "IPV" or "VERIFY" is specified, or the commands "*SET/APTSRC,VERIFY" or "*SET/APTSRC,IPV" is specified, this construct will cause motion generated by driving a circle to be output to the APT source as an APT circular record in the following format:

```
GOTO/xstart,ystart,zstart  
CYCLE/xcenter,ycenter,zcenter,i,j,k,raduis  
GOTO/xend,yend,zend
```

The direction of the i, j, k vector will determine the direction (CLW, CCLW) of the circle.

The optional parameter "IPV" will also cause the corresponding CL record number output to the APT source in front of all the GOTO points.

***SET/ APTSRC, CUTTER, APT**

NCL

PPRINT

scalar

This construct will specify which format of CUTTER statement to output to the APT source file.

“APT” specifies a seven parameter APT cutter statement, as specified in most commercially available APT systems to be output.

“NCL” specifies a standard **NCL** CUTTER statement to be output.

“PPRINT” specifies the **CUTTER** statements are commented out (\$\$) and only CUTTER statements with the word **PPRINT** in front of them will be output to the APT source file. This is useful when the part program contains pseudo (false) CUTTER statements and it is the intent of the user to use the APT source file as input to an NC Verification system where only the actual cutters should be recognized. The user should note that **NCL** makes no attempt to determine the validity of a PPRINT CUTTER statement. Thus any format may be specified.

“scalar” specifies the number of digits to output to the right of the decimal point for the CUTTER statements output to the APT source file. “scalar” must be in the

9 NCL CONTROL COMMANDS

range of 0 to 8. Entering a vale of zero (0) will result in the current setting for *SET/APTSRC,LOW or *SET/APTSRC,HIGH being used.

Note: The *SET/APTSRC,CUTTER,scalar command must be specified after the *SET/APTSRC,LOW or *SET/APTSRC,HIGH command, otherwise the “scalar” will not has any effect on the CUTTER decimal places output.

9.13.7 *SET/ APTSRC, DATA

This construct notifies **NCL** to output the REMARK program-information line to the start of the APT source file. By default the REMARK program-information line is output to the beginning of the APT file.

9.13.8 *SET/ APTSRC, IPV

This construct will cause the corresponding CL record number output to the APT source in front of all the GOTO points if it is used by itself.

If this command is specified together with the command "[*SET/APTSRC,CIRCUL](#)", a new style of circular motion record same as the "[*SET/APTSRC,VERIFY](#)" will be output to the APT file.

9.13.9 *SET/ APTSRC, IPVCOM

This construct notifies **NCL** to output the *PPRINT IPV* commands to the APT source file.

9.13.10 *SET/ APTSRC, LOW HIGH

This construct allows the user to expand on the number of digits output to the right of the decimal point for APT source files.

Where the following statements are affected as indicated:

CUTTER - LOW	= 3 places to the right of the decimal / HIGH=5
MOTION - LOW	= 4 places to the right of the decimal / HIGH=6
CIRCULAR - LOW	= 4 places to the right of the decimal / HIGH=6

It is recommended that HIGH be used when creating APT source files for **NCL/IPV**.

9.13.11 *SET/ APTSRC, REAL

Causes integer values contained in post-processor commands to be output as real numbers to the APT source file. The default is to output integer values as integer number to the APT source file.

9.13.12 *SET/ APTSRC, REMARK

Will cause the **NCL** source statement that generated the subsequent GOTO points to be output as a comment line (\$\$) in the APT source file. This is the default setting.

Example:

```
RAPID
$$ GODLTA/SF12
GOTO / 5.9908, -0.9521, 1.6667, 0.0000000, 0.0000000, 1.0000000
$$ GOFWD /SF1,PAST,SF13
GOTO / -0.2972, -0.9524, 1.6667, 0.0000000, 0.0000000, 1.0000000
$$ GOFWD /SF13,PAST,SF2
GOTO / -1.0838, -0.1193, 1.6667, 0.0000000, 0.0000000, 1.0000000
$$ GOFWD /SF2,PAST,SF6
GOTO / -1.0712, 0.1314, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -1.0328, 0.8804, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.9897, 1.8706, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.9563, 2.8225, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.9396, 3.4038, 1.6667, 0.0000000, 0.0000000, 1.0000000
GOTO / -0.9246, 3.9853, 1.6667, 0.0000000, 0.0000000, 1.0000000
$$ GOFWD /SF6,PAST,SF7
GOTO / -0.0232, 4.7463, 1.6667, 0.0000000, 0.0000000, 1.0000000
$$ GOFWD /SF7,PAST,SF16
GOTO / 5.9691, 3.8717, 1.6667, 0.0000000, 0.0000000, 1.0000000
```

See [*RESET/ APTSRC, REMARK](#).

9.13.13 *SET/ APTSRC, REMARK, ACTIVE

This construct is used to notify **NCL** to output the REMARK statements to the APT source file. The default setting is not to output the REMARK statements.

REMARK statements output to the APT source file can contain the @var parameters that will be expanded to contain the variable's value.

See [*RESET/ APTSRC, REMARK, ACTIVE](#).

9 NCL CONTROL COMMANDS

9.13.14 *SET/ APTSRC, TRACUT [, matrix]

This construct **NCL** to apply a separate **TRACUT** matrix to the output APT source file. If a matrix is not specified, then the matrix previously specified will be used. This matrix will also apply to simulation files created for use with **NCL/IPV**.

Since the tracut matrix only applies to created APT source files, the last *SET/APTSRC,TRACUT command specified will be applied to the entire APT source file. The user may not specify separate matrices for a single APT source file.

9.13.15 *SET/ APTSRC, VERIFY

This construct instructs *SET/APTSRC,CIRCUL to output a new style of circular motion record to the APT source. The style is compatible with some NC Verification systems such as "VERICUT."

This format is as follows:

```
GOTO/xstart,ystart,zstart  
CIRCLE/xcenter,ycenter,zcenter,i,j,k,raduis  
GOTO/xend,yend,zend
```

The direction of the i, j, k vector will determine the direction (CLW, CCLW) of the circle. Note that ***SET/APRSRC,CIRCUL** must be specified before this special format can be output. This construct by itself has no effect for the APT source output.

9.13.16 *SET/ AUTOST [, tol, angtol [, RETAIN]] OMIT

This construct is used when the Automatic Tool Start feature is desired.

When this option is set, **NCL** will automatically move the tool to be in contact with the drive surface and/or the part surface prior to a move if it is not already in contact with the drive surface and/or part surface. With the ***RESET/AUTOST** statement, **NCL** will show the current location of the tool, the calculated position of the tool if it were in contact with the drive surface and/or the part surface, and ask the question "IS MISMATCH ACCEPTABLE?" The effect of setting AUTOST is to assume that the answer to this question is always yes.

"tol" and "angtol" are used in conjunction with the optional parameter "OMIT" or "RETAIN". "tol" specifies the positional tolerance and "angtol" specifies the angular tolerance for **NCL** to output the theoretical calculated starting point of the move. If "RETAIN" is specified, the theoretical calculated starting point of the move will be output to the cl file if either the position between the current point and the calculated point is more than "tol", or the angular difference between the current tool axis vector and the calculated starting point tool axis vector is more than "angtol". If "OMIT" is specified, this calculated move will be eliminated. The default condition is "RETAIN" with "tol" at 0.002 and "angtol" at 1.0.

Note: Do not confuse the "tol" and "angtol" values of this statement with the "tol" and "angtol" values of the *RESET/AUTOST statement. These two values for the *SET statement are used to set the lower limits that the calculated cl point will be output if RETAIN is specified. The *RESET statement uses these two values to determine the upper limits that Automatic Tool Start feature will be allowed.

See [*RESET/AUTOST](#) command.

9.13.17 *SET/ AUTOUV

This construct instructs **NCL** to automatically use the "U" and "V" values calculated in the previous motion statement for the current motion statement. This is the default setting. For more information, see [chapter 6 Continuous Motion Statements](#).

9.13.18 *SET/ CANON

This construct causes the same action as [CANON/ ON](#).

9.13.19 *SET/ CASE

This construct is used to instruct the text string functions [FINDEX](#), [RINDEX](#), [STRCMP](#) to be case sensitive.

9 NCL CONTROL COMMANDS

9.13.20 *SET/CMDLEN, n

This construct is used to specify the maximum length (40 to 1026) of each command line generated by the interface. This includes the commands manually entered through the Command Prompt Mode.

Note: While it is permissible to set the command line length to be shorter than 72 characters it is not recommended, particularly when Source Formatting is in effect. Shorter command line lengths could result in lines of code being broken up in unpredictable ways and truncation of PPRINT and INSERT statements when written to the CL and APT source files.

9.13.21 *SET/CMDCOM, n

This construct is used to specify the starting comment column in the part program. This indirectly specifies the maximum length of each command line. The maximum command line length equals to “n-1”.

Note: While it is permissible to set the starting comment column to be shorter than 73 it is not recommended, particularly when Source Formatting is in effect. Shorter starting comment column could result in lines of code being broken up in unpredictable ways and truncation of PPRINT and INSERT statements when written to the CL and APT source files.

9.13.22 *SET/ DISPLAY [, geometry-type-list]

This construct is used to turn on the automatic display of the specified entity types as they are being defined. Valid entity types are: PT, LN, CI, PL, VE, PV, CV, SF, MX and SOLID. All entity types, with the exception of matrices, are automatically displayed unless a ***RESET/DISPLAY** command had been issued disabling the display of a particular entity type. If no geometry list is given with this command, then the automatic graphic display of all entities (matrix included) will be on.

9.13.23 *SET/ ELIMIT, scalar

This construct is used to set the **NCL** Error Limit number. This number, specified by scalar, will be the maximum number of syntax errors allowed in a batch run of **NCL** before the run will be terminated. The default error limit is 25.

9.13.24 *SET/ EXPCL

This construct is used enable the expanded motion CL file option. At the beginning of each motion, a 5210 record will be output containing the following:

- tool end point
- tool axis
- forward vector
- tool contact point on the part surface
- part surface normal at the contact point (on the same general direction as the tool axis vector)
- tool contact point on the drive surface
- drive surface normal at the contact point (pointing towards the tool, null if TLON)

Motion is output as 5200 records with the same format as 5210 records. At the end of each motion, a 5220 record is output containing the following:

- the tool end point
- forward vector
- tool contact point on the **check surface**
- check surface normal at the contact point (pointing in the forward direction)

9.13.25 *SET/ geometry-type, scalar

This construct is used to adjust the numbers used by **NCL** when assigning names during Automatic Identifier generation with “Prefix” type labelling system. The next generated identifier of the type geometry-type will have the associated number specified by scalar.

Note: This will not work if the Automatic Identifier generation is the “Subscript” type labelling system.

**9.13.26 *SET/ INDENT, ALL, scalar
SEP, scalar
OFF**

This construct is used to notify **NCL** to indent the part program statements.

9 NCL CONTROL COMMANDS

The "ALL" format of the INDENT command will cause statements processed after this command to be indented to the scalar-value column in the part program file.

The "SEP" format of the INDENT command will cause the first separator character in the **NCL** statements that follow to be indented to the scalar-value column. Separator characters are: equal signs (=) and slashes (/). The "ALL" and "SEP" formats may both be in effect at the same time.

Example of indenting statements using "SEP":

```
*SET/INDENT, SEP, 10  
  
1: PT1      =POINT/1, 2, 3  
2: LN1      =LINE/2, 2, 7, 3
```

If the indenting will cause any part of the statement to be shifted off the end of the line, the indenting is not done for that statement.

```
*SET/INDENT, OFF
```

The "OFF" format has the same effect as a [*RESET/INDENT](#) command, that is, it resets the "ALL" and "SEP" indentation column numbers to 1.

Example *SET/ INDENT commands:

```
*SET/INDENT, ALL, 5  
*SET/INDENT, SEP, 12  
*SET/INDENT, OFF
```

9.13.27 *SET/ MOTION

This construct is used to cause tool motion coordinate values to be displayed in the scrolling window (if it is opened) during an interactive session and to the print file (.pr) during batch processing. This is the default setting.

9.13.28 *SET/ NOWARN

This construct causes **NCL** to continue processing when a warning is encountered. When the NOWARN option is reset, **NCL** will stop every time a warning occurs. This has no effect in batch mode.

9.13.29 *SET/ PAUSE

This construct is used to notify **NCL** to enable the [*PAUSE](#) command after it was disabled by the [*RESET/PAUSE](#) command.

9.13.30 *SET/RUNCMD

This construct is used to notify **NCL** to enable the command mode when a stopping condition for [*RUN](#) is met after it was disabled by the [*RESET/RUNCMD](#) command.

9.13.31 *SET/ SCHECK

This construct automatically correct the so called "bow-tie effect" when defining ruled surfaces, fit surfaces and 4 curves surfaces through curves that run in opposite directions. This is the default setting.

9.13.32 *SET/ STATLN

This construct is used to notify **NCL** to enable the automatic display of Status line information on the error message line.

This information is normally displayed in the same place error messages are shown whenever there is no error message active. The default mode is to re-display the line each time any of the information on it changes. See the [*RESET/ STATLN](#) and [*SHOW/ STATLN](#) for related information.

9.13.33 *SET/ STOP

This construct is used to notify **NCL** to enable the [*STOP](#) command after it was disabled by the [*RESET/STOP](#) command.

9.13.34 *SET/STPCMD

This construct is used to notify **NCL** to enable the command mode when a [*STOP](#) command is executed after it was disabled by the [*RESET/STPCMD](#) command.

9 NCL CONTROL COMMANDS

9.13.35 *SET/ TRIMMED,FACE BASE DEFALT

This command will set the default method for driving trimmed surfaces. Specifying FACE will cause all trimmed surfaces to be driven as trimmed. Specifying BASE will cause the underlying surface for all trimmed surfaces to be driven. Specifying DEFALT will cause the value stored in each trimmed surface (set using the [REDEF/ sf, FACE](#) command) to be used to determine how a trimmed surface is driven. The default value is DEFALT.

9.13.36 *SET/ VER, vflag

The “vflag” construct is used to specify the version of **NCL** in which the current part program was created. The version flag is read by later versions of **NCL** and will cause certain functions to use the same algorithms as were used in the version specified by “vflag”. The use of the version flag is to ensure that older programs process in the same way as they did in previous versions. In most cases the version flag will not be necessary. However in cases where you receive errors (especially motion related errors) that did not occur in the earlier version, try the version flag to see if the errors go away.

The “*SET/VER, vflag” command will affect only certain motion calculations as will be noted in the Release Notes for each version. It will have no bearing on geometric enhancements, control statement processing, Unibase processing, etc. (unless otherwise noted).

It is the intent of the developers to recognize and apply the version flag to any changes that would likely affect the processing of old programs, particularly in the area of tool motion.

The following are valid version flags that can be used with the “*SET/VER, vflag” command.

VER, 8.1
VER, 8.2
VER, 8.209
VER, 8.3
VER, 8.4
VER, 9.0
VER, 9.007
VER, 9.008

```
VER, 9.1
VER, 9.2
VER, 9.3
VER, 9.4
VER, 9.5
VER, 9.6
VER, 9.7
VER, 9.8
VER, 9.9
```

It is recommended that the appropriate version flag (for the version in which the part program has successfully processed) will be placed near the top of the program.

The developers of **NCL** attempt to recognize and apply the version flag to any changes that would likely affect the processing of older programs. Even so, in some cases it is possible that the version flag will not solve the problem. In these cases it may be necessary to modify the program in some way (adding a **MAXDP**, changing **TOLER**, inserting an **INDIRV**, etc.) in order to make the program process in the new version.

9.13.37 *SET/ WLIMIT, scalar

This construct is used to set the **NCL** Warning Limit number. This number specified by scalar, will be the maximum number of warnings allowed in a batch run of **NCL** before the run will be terminated. The default warning limit is 25.

9.14 *SHOW

The SHOW command is used to display information during an interactive session in a scrolling window. SHOW can be entered at the keyboard, selected by menus or put directly into the part program file. SHOW commands which are in the part program will automatically open the scrolling window to display the information. A ***PAUSE** command can be placed directly after a SHOW command so that the user can analyze the displayed information. Otherwise the window is opened, information is displayed and the program continues to process.

The information displayed by a SHOW command which is in the part program will also be printed to the print file (.pr) during batch processing. The SHOW command can be abbreviated as S or SH.

9 NCL CONTROL COMMANDS

9.14.1 *SHOW/ identifier

This construct causes **NCL** to display information about the identifier. If the identifier is:

1. a geometry type, the geometry type and the canonical form are displayed.
2. a scalar, the current value is displayed.
3. a symbol, the word "symbol" is displayed.
- 4.) a symbol instance, the origin of the instance and the word "symbol instance" are displayed.
- 5.) an annotation, the annotation string is displayed.
6. a text variable, the text string is displayed.
7. a DATA statement, the content of the DATA statement is displayed.
- 8.) a macro label, the word "macro" is displayed.
- 9) a macro parameter, the word "macro parm" is displayed if not during a macro call. The corresponding type information is displayed if during a macro call.
10. undefined, error message "Identifier not previously defined" is displayed.

9.14.2 *SHOW/ ADISPL

This command causes **NCL** to display the current values in use for displaying curves and surfaces. See [DISPLAY](#) for more details.

9.14.3 *SHOW/ CUTTER

This construct causes **NCL** to display the current cutter information.

9.14.4 *SHOW/ FEDRAT

This construct causes **NCL** to display the current values being used for the primary and secondary feedrates.

9.14.5 *SHOW/ FILES

This construct causes **NCL** to display the name of the part program file being processed and indicate if a CLfile or APT source file is being created.

9.14.6 *SHOW/ MAXANG

This construct causes **NCL** to display the current **MAXANG** value. See **MAXANG** for details.

9.14.7 *SHOW/ MAXDP

This construct causes **NCL** to display the current value being used for **MAXDP**. See **MAXDP** for details.

9.14.8 *SHOW/ MODALS

This construct displays **CUTTER**, **THICK**, **NUMPTS**, **MAXDP**, **UNITS**, **FEDRAT**, **TOLER**, **MAXANG**, **TLAXIS**, **ADISPL** and the tool condition.

9.14.9 *SHOW/ MODSYS

This construct displays the current setting of **MODSYS** and the associated matrix information if MODSYS is on.

9.14.10 *SHOW/ NUMPTS

This construct displays the current **NUMPTS** value.

9.14.11 *SHOW/ REFSYS

This construct displays the current setting of **REFSYS** and the associated matrix information if REFSYS is on.

**9.14.12 *SHOW/ SOURCE [, scalar]
*SHOW/ SOURCE, S**

This construct allows the user to display the contents of the current part program file in a scrolling window during an interactive session. The word SOURCE may be abbreviated as SRC.

The first construct causes **NCL** to display 9 lines of the part program starting at the current line number. If the optional scalar value is given, **NCL** will display 9 lines of the part program starting at the line number specified by scalar.

9 NCL CONTROL COMMANDS

The second construct causes **NCL** to display the 9 lines of the part program starting at the current line number and then enter a scrolling mode. The user can then press the Enter key to scroll through the program. The following commands are accepted while in the scrolling mode:

- F - Sets the scroll direction to forward (toward the end of the file), this is the default direction.
- B - Sets the scroll direction to backward (toward the beginning of the file).
- Q - Quits the scrolling mode. The scrolling window will not be closed by itself. Close the window by using the [*WINDOW/CLOSE](#) command.
- Enter - Causes the program to scroll in the currently active direction.

9.14.13 *SHOW/ STATLN

This construct is used to notify **NCL** to display the current contents of Status line information on the error message line. The line is normally automatically re-displayed each time any of the information on it is changes unless the [*RESET/ STATLN](#) command has been processed. The ***SHOW/ STATLN** will "temporarily override" the ***RESET/ STATLN** mode and immediately display the current Status line contents. See [*SET/ STATLN](#) and [*RESET/ STATLN](#) for related information.

9.14.14 *SHOW/ THICK

This construct causes **NCL** to display the current **THICK** parameters on the terminal screen.

9.14.15 *SHOW/ TOLER

This construct causes **NCL** to display the current value of **TOLER**.

9.14.16 *SHOW/ TOOL

This construct causes **NCL** to display the current tool information on the terminal screen. This information consists of the current Tool Location, the current Tool Axis, and the current Forward Sense vector.

9.14.17 *SHOW/ TLAXIS

This construct will show the current tool axis mode (TT, DS, NORMAL,PS, SAME, etc.) at any time during an interactive session.

9.14.18 *SHOW/ TRACUT

This construct displays the current setting of **TRACUT** and the associated matrix information if TRACUT is on.

9.14.19 *SHOW/ UNITS

This causes the current **UNITS** setting to be displayed. UNITS are inches or millimeters.

9.14.20 *SHOW/ vocab-word

This causes the reserved number assigned to the vocab-word to be displayed.

9.15 *SKIP

The SKIP command causes **NCL** to “skip” to a different line in the part program file. Statements that are skipped by this command are left in the part program file but are not executed. Caution must be used to avoid skipping over statements such as **LOOPST**, **LOOPND**, **MACRO** and **TERMAC**.

For example, it is valid to be processing a loop, and then to skip over the **LOOPND** statement, however **NCL** will not terminate the loop until it processes the **LOOPND** statement. The word SKIP may be abbreviated as SK.

Example SKIP Commands:

```
*SKIP  
*SKIP/7  
*SKIP/TO,134  
*SKIP/-3  
*SKIP/TO,END
```

The valid syntax constructs for the SKIP commands are:

9 NCL CONTROL COMMANDS

9.15.1 *SKIP

This construct will cause **NCL** to skip forward one line in the part program. It should be noted that the user can, while in COMMAND mode, use the up and down arrow keys to move through the part program file.

Therefore, the *SKIP command is generally used when a fairly large number of lines need to be skipped or when the user knows what line number he wishes to skip to.

9.15.2 *SKIP/ scalar

This construct will cause **NCL** to skip either forward or backward in the part program file by the number of lines specified by the scalar. A positive value will skip forward and a negative value will skip backward.

9.15.3 *SKIP/ TO, scalar

This construct will cause **NCL** to skip to the line number of the part program specified by the scalar.

9.15.4 *SKIP/ TO, END

This construct will cause **NCL** to skip to the end of the part program file.

9.16 *STOP

The *STOP command will cause **NCL** to stop processing. The line following the *STOP command will become the current part program line number. The user may then enter selections from the menus or enter COMMAND mode to enter commands from the keyboard.

The *STOP command can be placed at strategic places in the part program, such as the beginning of a motion sequence, so that the user may enter COMMAND mode and “step” through the part program line by line for a more careful analysis.

9.17 *SYSTEM [/ system-command [, OFF]] ON

The SYSTEM command by itself without any optional parameters allows the user to temporarily leave an interactive session and enter the computer's operating system. **NCL** will display the appropriate command for returning to your interactive session as you enter the operating system window. On most systems the return command is the word "exit".

The optional parameter "*system-command*" is the operating system command to execute. The "*system-command*" can either be a text string in quotes or a text variable.

"ON" specifies an operating system window will be opened. "OFF" specifies no operating system window will be opened and is the default condition.

9.18 *TIME

The *TIME command will display the amount of time that has elapsed from when the last *TIME command was given and the amount of time that has elapsed from the beginning of the current interactive session. The word TIME may be abbreviated as TI.

9.19 *VER

The *VER command in addition to displaying the current version number of **NCL**, will automatically open the scrolling window if it is not already open.

9.20 *WINDOW [/ OPEN] And *WINDOW/ CLOSE

These commands will allow the opening or closing of the scrolling window while in an interactive session.

The word WINDOW may be abbreviated WI.

The command *WINDOW will cause the window to be opened. That is, the word "OPEN" is optional.

9 NCL CONTROL COMMANDS

9.21 ** Command

A ** command serves two functions. The first is to insert any of the above control commands (commands that must begin with a *) into the part program file during an interactive session. The second is to suppress the processing of a control command during the definition of a Loop or a Macro.

NCL control commands, when entered from the command line, are not put into the part program file. The “**” syntax can be used when the user desires to insert a control command into the part program during an interactive session. The control command is entered into the part program preceded by the “**” characters.

When a control command is embedded in a Loop or a Macro it is recommended that the command be preceded by the “**” characters. This will prevent **NCL** from attempting to act upon the control command when the Loop or Macro is being defined. This is desirable, for example, when a SHOW command appears in a Macro for the purpose of displaying information about a variable which will not be defined until the Macro is actually called.

The following commands will not be processed during the definition phase of a Loop or a Macro when in RUN mode, even when preceded by a single “*”. However, when “stepping through” the program line by line in COMMAND mode these statements are processed. Thus, the “**” technique is still recommended.

- *PAUSE
- *RESET
- *SET
- *SHOW
- *STOP

9.22 NON-CONTROL Command * Statements:

Any statement that is preceded with a “*” in column one that is not recognized as a valid control type command will be treated as a regular **NCL** statement and processed but not put in the part program file.

***NCL** statement

Example * statements:

*CANON/ON

```
*TSTPT=PT/1,SS4,3.5  
*DRAFT/SCALE=.75
```

9.23 NON-CONTROL Command * Statements In LOOPS And MACROS

The following "*" commands read from the part program file during the definition of statements in a **MACRO**, **LOOPST/ LOOPND** or **DO** loop regions are NOT processed at that time.

- *DBSHOW
- *PAUSE
- *RESET
- *SET
- *SHOW
- *STOP

They are processed during the execution phase of the looping region. This eliminates the need to input these "*" commands in system type macros with the "****" as was the case in earlier versions of **NCL**. However, because of the items listed below, the "****" technique is still recommended. Unlike older versions, one of the asterisks (*) is not removed by **NCL** in the definition phase of a macro or loop. Thus, subsequent processing of the program will not produce errors.

Note:

1. If the user inputs a "*" command into a macro or a loop in interactive mode, the system will try to process the command. To get around the problem, input the "*" command using the "****" technique. This will input the statement into the part program but will not process it at that time.
2. When running in line by line mode (pressing the carriage return at each statement) and the system reads a "*" command inside a macro or a loop in the definition phase, it will try to process that command. This can be worked around by using the "****" technique.

10 NCL/IGES

The name IGES is an acronym which stands for the Initial Graphics Exchange Specification. This standard has been developed to allow the exchange of geometric and graphic data between dissimilar CAD/CAM systems.

The **NCL/IGES** converter imports and exports ASCII IGES files. Imported files are converted to **NCL** Unibase files and **NCL** Unibase files are exported to IGES files.

NCL/IGES is implemented as a subset of the U.S. Department of Commerce IGES Version 5.0 supporting those entities that allow a direct conversion into **NCL** entity types.

10.1 Startup

To start **NCL/IGES**:

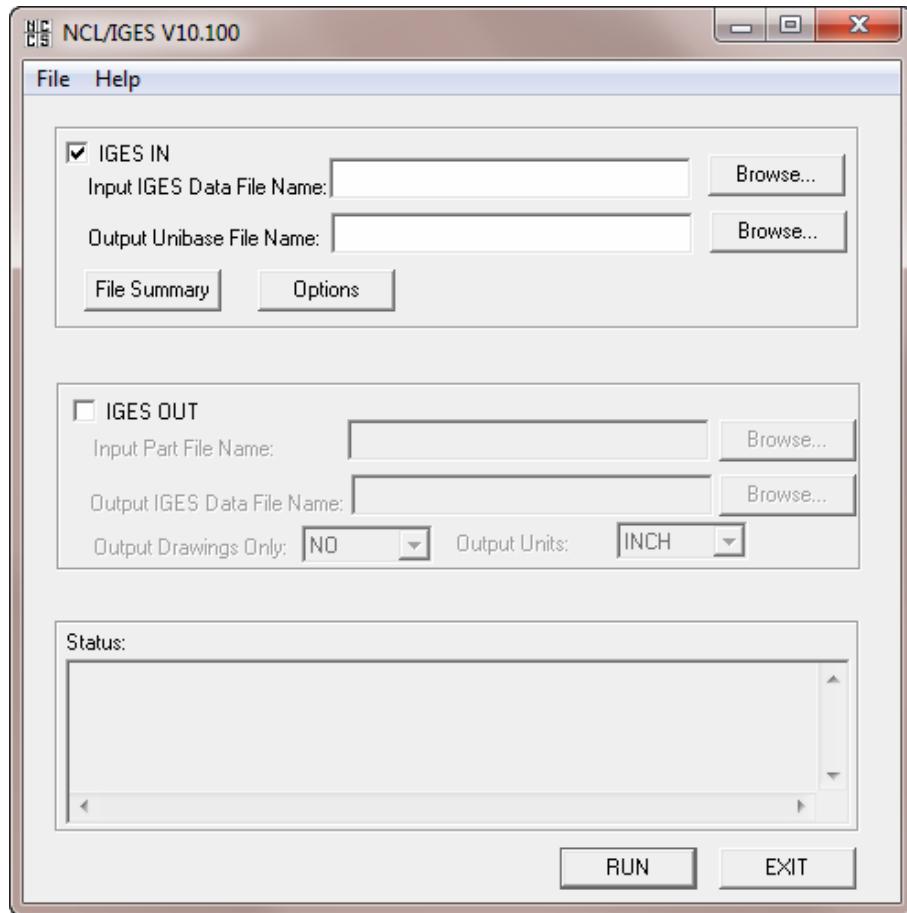
From within **NCL**, click *FILE > IGES*

NCL/IGES can also be started as followed:

Double click the IGES shortcut icon in the **NCCS** folder on the Desktop.

10.2 Importing IGES Files

Start **NCL/IGES** as previously described. The form on the next page will appear:



10.2.1 IGES IN

This button will be depressed by default. Press this button to switch from “IGES OUT” mode to “IGES IN” mode.

10.2.2 Input IGES Data File Name:

Enter the name of the IGES file you wish to import. Use the *Browse* button to browse for a file.

10.2.3 Output Unibase File Name:

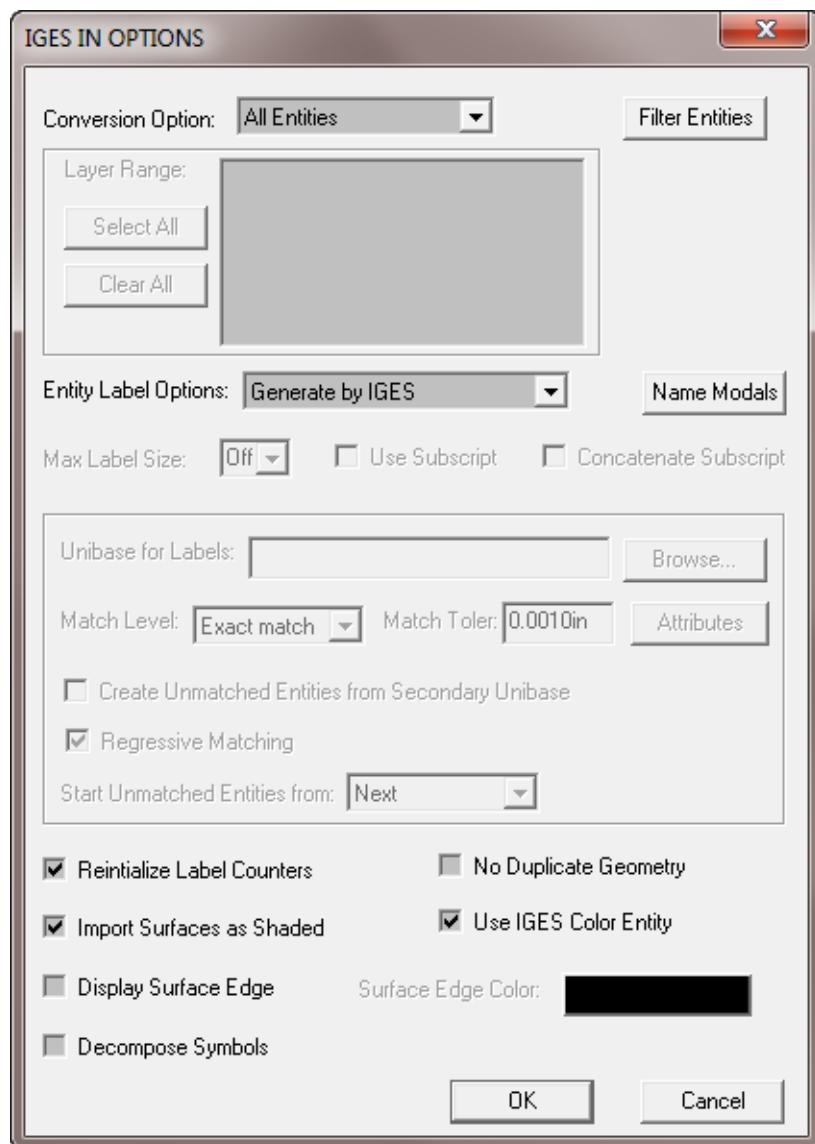
Enter the desired name of the Unibase file which will contain the converted entities. If an existing file is specified it will be overwritten. If an output file is not specified the input file name will be used with a .u extension.

10.2.4 File Summary

Click this button to obtain a summary of the entities contained in the IGES file. The summary data will be displayed in the Status Window. This data will also be written to a file named *input.lst*, where *input* is the name of the input IGES file.

10.2.5 Options

Click this button to set various conversion options. The following “IGES IN” Options form will appear:



Conversion Option

All entities

Select this option to convert all IGES entities (accept the DRAWING entity) regardless of which layer the entities are on.

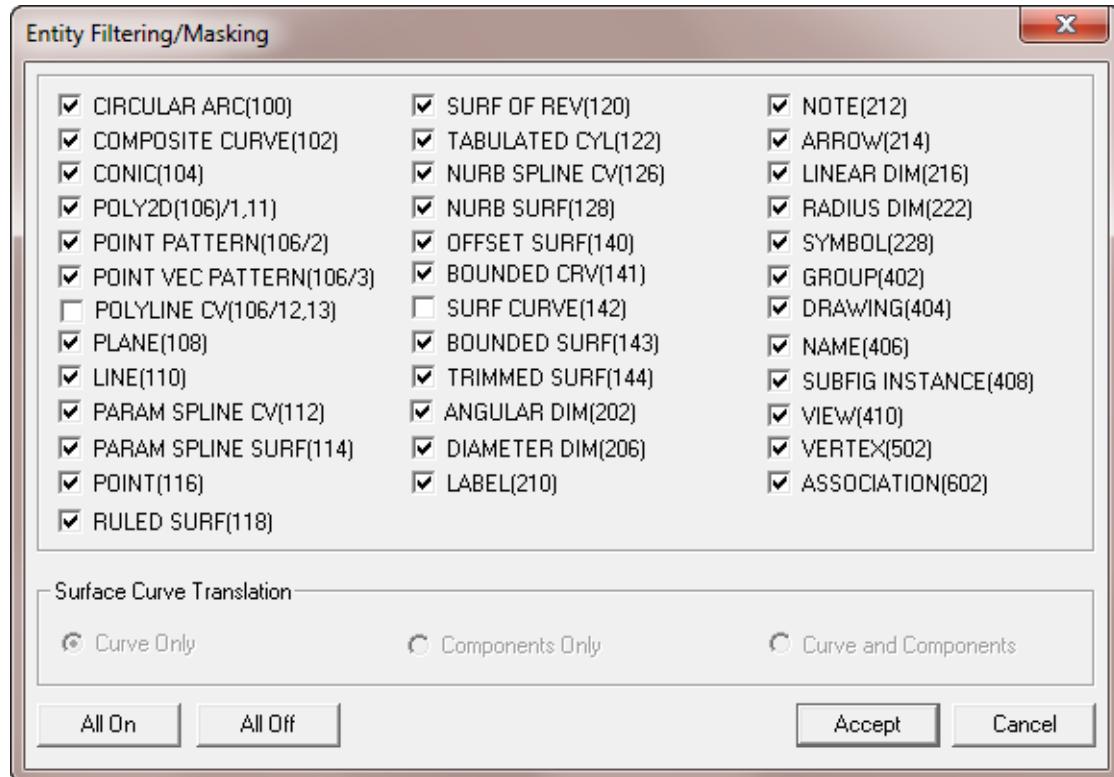
By layer

Select this option to convert entities by layer. When this option is selected the Layer Range list box will be activated and list of layers that exist in the input IGES file will be displayed.

Click on the layers you wish to convert. Click *Select All* to select all layers. Click *Clear All* to clear previously selected layers.

Filter Entities

Click this button to filter out specific entity types from the conversion. The form shown below will appear.



Depress or check the box next to the entity types you want to translate.

Check item “POLYLINE CV (106/12,13)” will cause IGES to translate the corresponding data as a Rational B-Spline curve, otherwise it will be translated as a pattern.

Check item “SURF CURVE (142) will activate the Surface Curve Translation Section with the following options.

Curve Only

The surface curve is translated, but the components are not translated independently.

Components Only

The components of the surface curve are translated but the surface curve is not.

Curve and Components

The surface curve and its components are translated.

All On

Press this button to translate all entities.

All Off

Press this button to filter out all entities.

See the [Imported Entity Mapping](#) and [Exported Entity Mapping](#) sections for detail of the entity types. Also see the [View Entity \(Type 410\)](#) and the [Drawing Entity \(Type 404\)](#) for details of these two entities.

ACCEPT

Click this button to accept the filter settings and exit the form.

CANCEL

Click this button to exit the form without saving the filter settings.

Entity Label Option

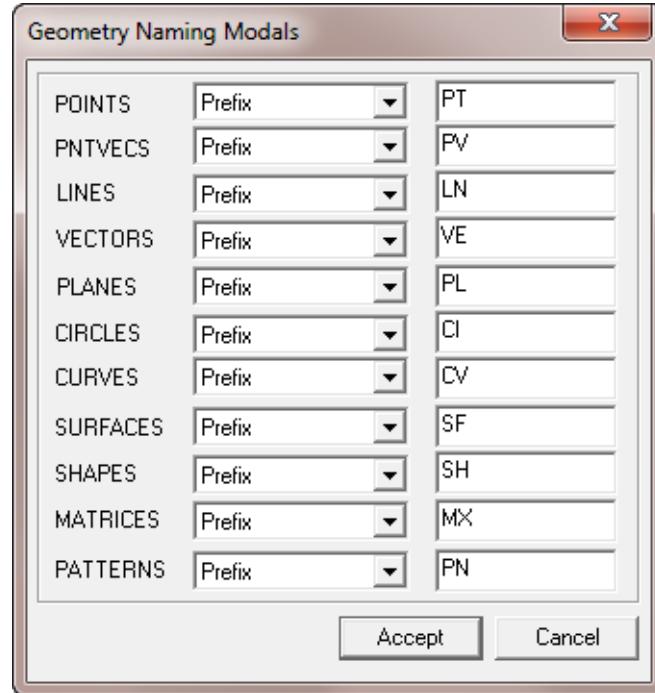
The entity label options are used to determine how the converted entities will be labeled (named). Click this button to produce the form shown on next page.



Generate by IGES

Click this option if you want **NCL/IGES** to automatically assign a name to the imported entities according to the “Name Modals” setup. This is the default option.

Click the “**Name Modals**” button to open the following form.



Choose the desired naming convention (prefix or subscript) by clicking the toggle button next to the entity type.

When Prefix is used entities will be named using the prefix characters (up to 20 alpha characters) shown in the box to the right of each entity type and

10 NCL/IGES

a number, starting at 1. The default setting of this form uses the conventional **NCL** autonaming convention.

For example:

Points	PT1, PT2,...PTn
Lines	LN1,LN2,...LNn
Circles	CI1,CI2,...CIn
Planes	PL1,PL2,...PLn
Curves	CV1,CV2,...CVn
Surfaces	SF1,SF2,...SFn
Paterns	PN1,PN2,...PNn

When Subscript is used entities will be named using the characters (up to 20 alphanumeric characters) shown in the box to the right of each entity type and a subscript number, starting at 1.

For example:

<u>Entity Type</u>	<u>Right Column Entry</u>	<u>Entity Names</u>
Points	PTT	PTT(1),PTT(2),...PTT(n)
Lines	LNN	LNN(1),LNN(2),...LNN(n)
Circles	CII	CII(1),CII(2),...CII(n)
Planes	PLL	PLL(1),PLL(2),...PLL(n)
Curves	CVV	CVV(1),CVV(2),...CVV(n)
Surfaces	SFF	SFF(1),SFF(2),...SFF(n)
Paterns	PNN	PNN(1),PNN(2),...PNN(n)

Click the ACCEPT button to accept the current settings and exit the form.

Click the CANCEL button to exit the form without saving any changes.

From IGES Label Field or From IGES Property

Click either one of this option will activate the following buttons:

Max Label Size

This specifies the maximum size of the labels allowed.

Use Subscript

This specifies labels as subscripted name.

Concatenate Subscript

This specifies labels created by concatenating the label and subscript fields of the IGES file. This is usually for IGES files generated by UG-II.

Using Existing Unibase

Click this option if an IGES file which contains a model that is theoretically the same as an existing unibase with some minor differences. In this case, **NCL/IGES** will compare the entities in the input IGES file with entities in the specified unibase. If the compared entities are exactly the same, then the names used in the existing unibase will be used for the same entities in the IGES file, otherwise a new name will be used.

There are four levels of matching with this option. All entities with an exact match will be translated first. Once all matching entities are translated, **NCL/IGES** will compare the unmatched entities searching for the closest match. Depending on the entity type there are up to four levels to determine the closest match:

Point:

- Level 1 - Searches all remaining unmatched points for the closest point.
- Level 2:4 - Not used.

Lines:

- Level 1 - The closest line that has the same length and direction.
- Level 2 - The closest line that has the same direction.
- Level 3 - The line with the closest start point.
- Level 4 - Not used.

Circles:

- Level 1 - Circle with the same center point, radius and central angle but with reversed normal vector.
- Level 2 - Circle with the same center point, central angle and normal vector but with different radius.
- Level 3 - Circle with the same radius, central angle and normal vector.
- Level 4 - Closest circle with the same central angle and normal vector.

Planes:

- Level 1 - Plane with the same normal vector and the closest point.
- Level 2:4 - Not used.

Polylines and Patterns:

- Level 1 - Polyline or pattern with all points translated.
- Level 2 - Polyline or pattern with the same number of points.
- Level 3:4 - Not used.

Point-vectors:

- Level 1 - Point-vector with the same length, direction and closest point.
- Level 2 - Point-vector with the same direction and the closest point.
- Level 3:4 - Not used

Curves:

- Level 1 - Closest curve that is an exact match but has been translated.
- Level 2 - Closest curve with same sized bounding box and with end points translated along a vector to a different position.
- Level 3 - Closest curve with end points translated along a vector to a different position.
- Level 4 - Not used.

Rational B-spline Surfaces and Mesh Surfaces:

Planar Surface:

- Level 1 - Planar surface with same normal, same distance and the closest boundary.
- Level 2 - Planar surface with same normal and with the closest boundary.
- Level 3:4 - Not used.

Spherical Surfaces:

- Level 1 - Spherical surface with same center and radius.
- Level 2 - Spherical surface with same center point, but different radius.
- Level 3 - Spherical surface with the closest center point.
- Level 4 - Not used.

Cylindrical Surfaces:

- Level 1 - Closest cylindrical surface with the same radius, axis vector and height.
- Level 2 - Cylindrical surface with the same start point, radius and axis vector but with a different height.
- Level 3 - Cylindrical surface with the same start point and axis vector but with a different height and radius.
- Level 4 - Cylindrical surface with the same axis vector, closest radius.

Cone Surfaces:

- Level 1 - Closest cone surface with the same radius, angle, axis vector and height.
- Level 2 - Cone surface with the same start point, radius, angle and axis vector but with a different height.
- Level 3 - Cone surface with the same start point, angle and axis vector but with a different height and radius.
- Level 4 - Not used.

Freeform Surface:

- Level 1 - Same surface but reversed in U and/or V.
- Level 2 - Closest surface with same shape.
- Level 3 - Surface with same corner points.
- Level 4 - Surface with same corner points to within 5 times tolerance

Trimmed Surfaces:

Base surface goes through the same qualifications as untrimmed surfaces. If the base surface match for the level being tested, then

- Level 1 - Surface with the same number of trimming curves and a bounding box that intersects the bounding box of the surface being matched.
- Level 2 - Surface with a bounding box that intersects the bounding box of the surface being matched.
- Level 3:4 - Not used.

All the remaining unmatched entities will be labeled with the standard prefix for that type of entity, and the number will start one higher than the highest number used for that type of entity. The label of any entity not used in the existing unibase will be displayed in the “*Status*” area of the main form. This information will also be placed into the “*.lst*” file.

Unibase for Labels

This feature is only activated when the “Using Existing Unibase” option is selected. Enter the file name of the secondary unibase or use the *Browse* button to browse for a secondary unibase file

Match Level

This feature is only activated when the “Using Existing Unibase” option is selected. There are five choices for this toggle button, they are: Exact Match, Level 1, Level 2, Level 3, and Level 4.

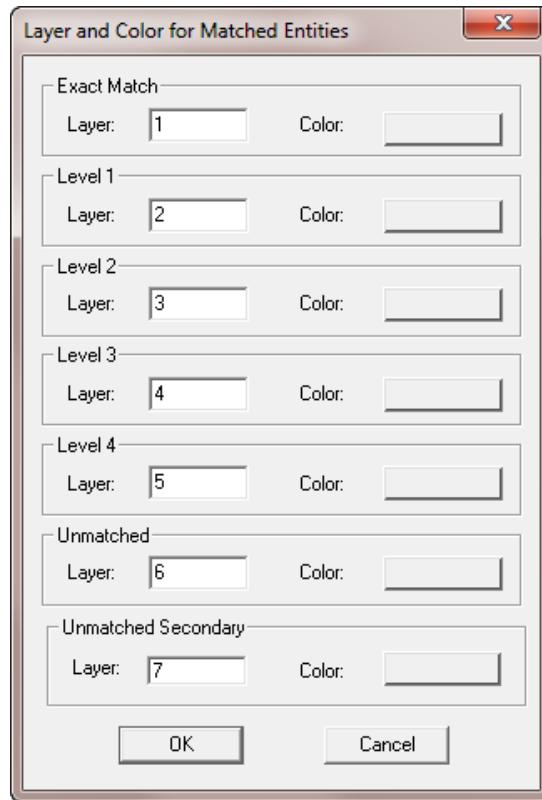
Matching Toler

This feature is only activated when the “Using Existing Unibase” or “No Duplicate Geometry” option is selected. This specifies the tolerance under which two entities are considered to be matched or duplicated. This value will be used as is in “Exact” matching or “No Duplicate Geometry”. This value will be expanded by different internal specified factors for the other levels of matching.

Attribute

This feature is only activated when the “Using Existing Unibase” option is selected. Click this button to open the form as shown on next page.

This form is used to assign the matching/unmatching entities put in different layers and colors according to the matching levels



Create Unmatched Entities from Secondary Unibase

This feature is only activated when the “Using Existing Unibase” option is selected. Check this item to create unmatched entities from the secondary unibase, i.e. the existing unibase.

Regressive Matching

This feature is only activated when the “Using Existing Unibase” option is selected. Check this item to initialize regressive matching. Regressive matching matches each IGES entity with both the available (unmatch) unibase entities and the other entities of the same type that were previously matched on this level. If this option is not checked IGES would search an entity in the secondary unibase that matches the entity from the IGES file per the rules base on the current level of matching. As soon as a match is found, the new entity is assigned the label of the corresponding entity from the secondary unibase.

Start Unmatched Entities from

This feature is only activated when the “Using Existing Unibase” option is selected. There are two choices for this toggle button, they are: Next or Secondary. This controls how the unmatched entities labels number in the new unibase started. The unmatched entities label number can be started from the next highest entity count or the entity after the highest matched entity in the secondary unibase.

Re-initialize Label Counters

Check this item to re-initialize the automatic label generation counters when translating multiple files in a single IGES run.

No Duplicate Geometry

Check this item will allow only one copy of duplicate entities to be translated.

Import surfaces as shaded

When this item is checked, imported surfaces will be flagged as shaded. When the resulting Unibase is loaded into **NCL** and *View > Modals > Shading* is set to *On*, the imported surfaces will be shaded. Otherwise the imported surfaces must be set to *shaded* while in **NCL**.

Use IGES Color Entity

When this item is checked, imported entities will be assigned with color specified by the Color Entity (314), otherwise the default **NCL** colors will be assigned to the imported entities. If the color specified in the Color Entity is not supported by **NCL**, the closest matched **NCL** color will be used.

Display Surface Edge

When this item is checked, imported surfaces will be rendered with their edges displayed.

Surface Edge Color

Defines the color to display surface edges with. 'Default' uses the same color as the surface is displayed in, while the other choices select an actual color.

Decompose Symbols

Check this item to decompose symbols during the translation of an IGES file.

OK

Click this button to accept the changes and close the IGES in Options form.

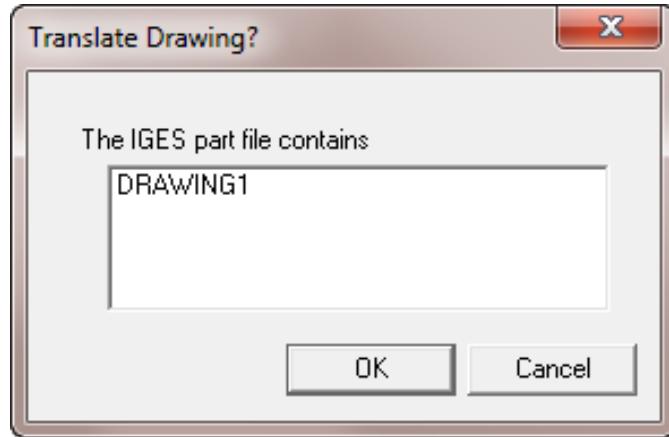
Cancel

Click this button to cancel all the changes and close the IGES in Options form.

10.2.6 RUN

Click this button to begin the conversion. IGES header information, percentage of file processed, and translation results will be displayed in the status window. The header and result data is also written to the *output.lst* file, where *output* is the name of the output Unibase file.

If the IGES file contains the IGES DRAWING entity you will be presented with the following form giving you the option to translate the DRAWING data or the 3D model data.



Click *CANCEL* to translate only the 3D model data. This would be the normal course of action.

To translate a *Drawing*, click on the Drawing you want to translate then click *OK*.

See the [Translating the IGES Drawing Entity](#) section for more details about *Drawings*.

10.2.7 EXIT

Click this button to exit **NCL/IGES**.

10.3 Viewing The Translated File

From within **NCL** click *FILE > LOADU* and click on the **NCL/IGES** created Unibase. The Unibase file can also be loaded using the **UBFN** or **LOADU** command within a **NCL** part program file.

10.4 Translating The IGES View Entity (Type 410)

The IGES View Entity (type 410) is translated into **NCL** views with the name IGES1, IGES2, etc. depending on how many views are present in the IGES file. To determine if the IGES translation program has translated views, read the *File Summary* that appears after an IGES file is translated or review the contents of the .lst file that is created during the translation process.

If you have already loaded the resultant Unibase, click *VIEW > MANAGE > LOAD VW* to list and/or load views.

The intent of the IGES View Entity differs depending on the exporting system. The following are the most common scenarios:

- Each IGES created view would simply display the 3D model in a different spacial orientation, much the same way that **NCL** views are displayed. In this case the entire model can be viewed in either the IGES created views and/or any **NCL** created view. In such cases the IGES views have no significant intent other than to simply “view” the model.
- Each IGES view, while perhaps representing a different spacial orientation of the model, may contain entities that are not visible in any other view. For example, *drafting* entities may appear in the IGES1 view but not in any other view. Similarly, the IGES2 view may display a fixture base that is not present in any other view. In other cases an IGES view may contain only a few entities, perhaps representing a sectioned view of the model.

When entities appear only in a particular view, such as previously described, they are known as *view dependent* entities. View dependent entities can *only* be displayed in the view on which they are dependent. These entities do, however, exist in model space and can be accessed by **NCL**'s motion and geometry commands. To see a view dependent entity you must load the view that contains

that entity. In some cases none of the views will contain a complete view of the model. If this is the case you will want to remove the *view dependent* attributes of the model.

10.4.1 Removing View Dependency

There are a couple of ways to work around view dependency. The easiest way is to utilize the “Filter Setup” Form of the “IGES IN” section. First, click the *OPTIONS* button, then *Filter Entities* button. Once the filter set up form opened, undepressed the “VIEW 410” button and click *ACCEPT*. In this way, the view dependency will not be translated.

The other way is to retrieve the model from the Unibase using **NCL**’s *UBFN* and *GET* commands. When entities are retrieved in this manner the model’s view attributes are not retrieved. For example:

```
UBFN /iges_created_unibase.u
GET /ALL
UBFN /CLOSE
SAVEU/new_unibase.u
```

It should be noted that when using this technique drafting entities will not be retrieved.

Another way to remove view dependency is to remove the view dependent flag from the original IGES file. This could be done using the following *gawk* script. For example:

Using a text editor, create a file call *view.awk* with the following content.

```
# Program to remove view dependency from an IGES file

BEGIN {icol=41; i=icol-1; j=icol+8; k=80-icol-7}
{
if(substr($0,73,1) == "D")
  if(substr($0,74,7)%2 == 1) print substr($0,1,i) "bbbbbbbb0" substr($0,j,k)
  else print
else print
}
```

where: “**bbbbbbb**” represents 7 blank spaces

Next, process the IGES file through awk using the following command:

10 NCL/IGES

```
awk -f view.awk old.igs > new.igs
```

Where *old.igs* is the name of the original IGES file and *new.igs* is the name of the file being created which does not contain the view dependent attributes.

10.5 Translating The IGES Drawing Entity (Type 404)

During the translation process **NCL/IGES** will inform you if one or more *Drawings* exist in the IGES file and will give you the option of translating a *Drawing* or the *model* data (for details see the RUN option above). In the event that a *Drawing* does exist you should process the IGES file once to create a Unibase file containing only *model* data and then again to create a Unibase file (or files) that contain only *Drawing* data.

An IGES Drawing is simply a list of “views” and locations for those views in *drawing space*, the end result being a electronic blueprint which can be plotted and used for reference.

NCL/IGES translates the IGES Drawing entity into a **NCL** view. The views are named *DRAWING1*, *DRAWING2*, etc. depending on how many *Drawings* are in the IGES file. Each *Drawing* must be translated into a separate Unibase file.

To view the Drawing in **NCL**, click *VIEW > MANAGE > LOAD VW*, then click on the view named *DRAWING1* (or *DRAWING2*, etc.)

All entities placed on a drawing are projected onto a two-dimensional plane and are located in *drawing space*. Thus, *Drawings* are primarily used to create a hardcopy plot for reference.

10.6 Subfigures And Subfigure Instances

The IGES Subfigure and Subfigure Instance entities are translated into **NCL/CADD** *Symbols* and *Symbol Instances*.

The entities within a **NCL/CADD** symbol cannot be used for geometry construction or tool path generation unless the symbol is first decomposed.

To decompose a symbol click *TOOLS > CADD > SYMBOLS > DECOMPOSE*, then click on the symbol you want to decompose.

10.7 Translating The IGES Solid Entity (Type 186)

A Solid Entity consists of the dependent shell entity (Type 514), the face entity (Type 510), the loop entity (Type 508), the vertex list entity (Type 502) and the edge list entity (Type 504).

The translation process converts the face entity (Type 510) to “Trimmed Surface”, the loop entity (Type 508) to “Boundary Curve” and ignores the rest of the entity types.

10.8 User Definable Maximum Parametric Record Size

In order to help solve translation problems associated with large *Parametric Surfaces, Drawings and Views*, the default parametric record size is user definable.

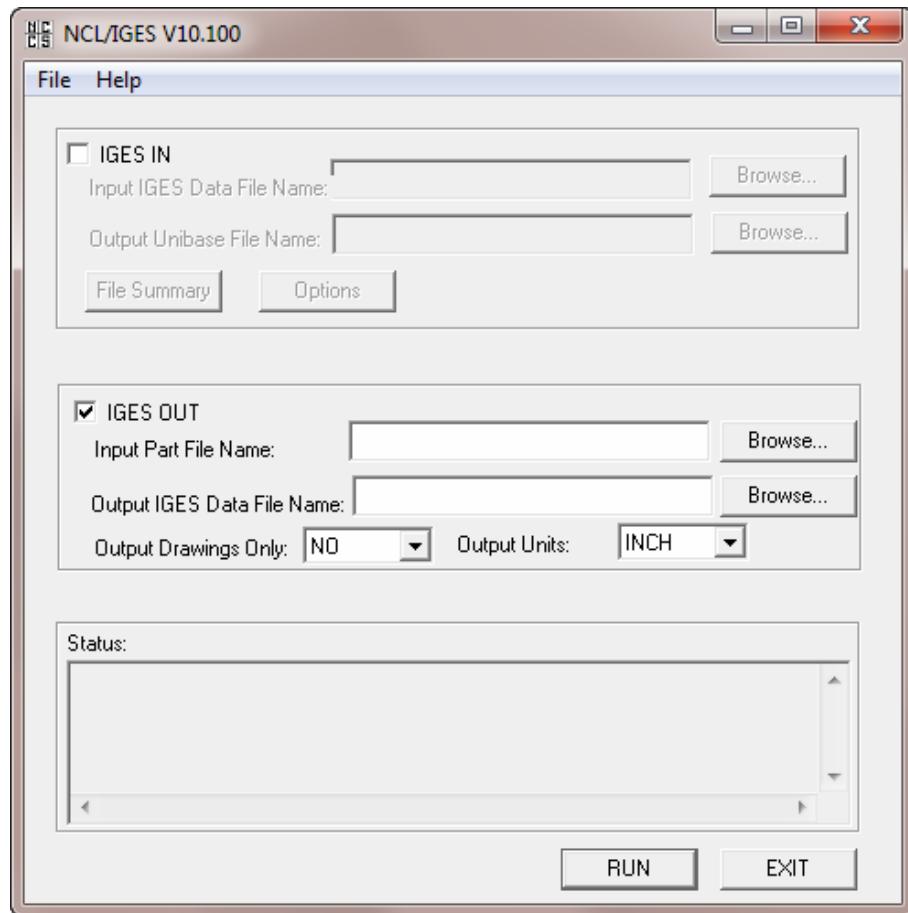
The following environment variable shown with it's default value is located in the *NCCS/NCL101/interface/ncliges.init* file.

```
MAX_PAR_REC=300000
```

If an error occurs during the processing of an IGES file referring to “parametric record size”, simply increase the value shown above and try to process the file again. *MAX_PAR_REC* environment variable can also be defined in the user's *user.init* file.

10.9 Exporting IGES Files

Start **NCL/IGES** as previously described. The form as shown on next page will appear:



10.9.1 IGES OUT

Press this button to switch from IGES IN mode to IGES OUT mode.

10.9.2 Input Part File Name:

Enter the name of the Unibase file you want to export. Use the *Browse* button to browse for a file.

10.9.3 Output IGES Data File Name:

Enter the desired name of the IGES file which will contain the converted entities. If an existing file is specified it will be overwritten. If an output file name is not specified the input file name will be used with a *.igs* extension.

10.9.4 Output Drawings Only

Set this button to *NO* to export only *3D model* data. Set this button to *YES* to export only *2D Drawing* data (if no *Drawings* exist in the Unibase file nothing will be exported).

10.9.5 Output Units

Set this button to *INCHES* if you want the IGES data to be written in inches. Set this button to *MM* if you want the IGES data to be written in millimeters.

10.9.6 RUN

Click this button to begin the translation.

An input form will appear prompting you to *Enter the identification information*. Enter any desired information that would describe the contents of the IGES file. You should always make an entry to this form as some IGES converters will not accept an IGES file that has no identification information.

Click *OK* when you are finished entering the information.

Information about the file being exported will appear in the *Status Window*. When the form becomes active again the exporting process is finished.

10.9.7 EXIT

Click this button to exit **NCL/IGES**.

10.10 Imported Entity Mapping

Imported entities are mapped from IGES to **NCL** as follows. There is no limit to the number of imported entities.

Geometric Entities:

<u>IGES</u>	<u>Type</u>	<u>Form</u>	<u>NCL</u>
Circular Arc	100	0	Circle and Circular Arc
Composite Curve	102	0	Composite Curve
Conic Arc	104	0	Ellipse or Curve
Copious Data	106	1	Separate Lines
		2	NCL Point Pattern
		3	NCL Point-Vector Pattern
		11	Polylines
		12	Polyline Curve represented by points
		13	Polyline Curve represented by point-vectors
Curve on a Parametric Surface	142	0	Surface Spline
Line	110	0	Line
Offset Surface	140	0	Offset Surface
Parametric Spline Curve	112	0	Rational B-Spline Curve
Parametric Spline Surface	114	0	Mesh Surfaces
Plane	108	0	Plane
Point	116	0	Point
Rational B-Spline Curve (up to 99th degree)	126	0	Rational B-Spline Curve
Rational B-Spline Surface (up to 99th degree)	128	0	Rational B-Spline Surface
Ruled Surface	118	0	Ruled Surface
Surface of Revolution	120	0	Surface of Revolution
Tabulated Cylinder	122	0	Tabulated Cylinder
Trimmed Parametric Surface	144	0	Trimmed Surface

Annotation:

<u>IGES</u>	<u>Type</u>	<u>Form</u>	<u>NCL</u>
Angular Dimension	202	0	Angular Dimension
Copious Data	106	20/21 31-38	Center line Crosshatch
Diameter Dimension	206	0	Diameter Dimension
General Label	210	0	Label
General Notes	212	0	Notes
General Symbol	228	0	General Symbol
Leader (Arrow)	214	0	Arrow on Curve

Linear Dimension	216	0	Linear Dimension
Radius Dimension	222	0	Radius Dimension

Property:

<u>IGES</u>	<u>Type</u>	<u>Form</u>	<u>NCL</u>
Name	406	15	NCL Entity Label

Structure Entities:

<u>IGES</u>	<u>Type</u>	<u>Form</u>	<u>NCL</u>
Subfigures	308	0	Symbols
Drawings	404	0	A View called DRAWINGn
Subfigure Instances	408	0	Symbol Instance
Groups	402	7	Individual Entities
Views	410	0	A View called IGESn
Solid	186	0	Trimmed Surfaces and Boundary Loops

Attributes:

<u>IGES</u>	<u>NCL</u>
Level	Layer
Pen	Pen
Line Font Pattern	Line Style
Solid	Solid
Dashed	Dashed
Phantom	Phantom
Center Line	Center Line
Line Weight	Line Weight

10.11 Exported Entity Mapping

Exported entities are mapped from **NCL** to IGES as follows:

<u>NCL</u>	<u>IGES</u>	<u>Type</u>	<u>Form</u>
Circle	Circular Arc	100	0
Composite Curve	Composite Curve	102	0
Conic	Conic Arc	104	0

Line	Line	110	0
Matrix	Transformation Matrix	124	0
Mesh Surface	Parametric Spline Surface	114	0
NCL Entity Label	Name	406	15
NCL curve	Rational B-Spline Curve	126	0
NCL surface	Rational B-Spline surface	128	0
Pattern	Copious Data	106	2
Plane	Plane	108	0
Point	Point	116	0
Point-Vector	Copious Data	106	3
Rational B-Spline Curve	Rational B-Spline Curve	126	0
Rational B-Spline Surface	Rational B-Spline Surface	128	0
Trimmed Surface	Trimmed Parametric Surface	144	0
Vector	Copious data	106	3

10.12 Miscellaneous

- **NCL** entity type SOLID is not output to the generated iges file.
- **NCL** entity type ANOTE is output as drafting entity to the generated iges file.

11 NCL/STEP

The name STEP is an acronym which stands for the Standard for the Exchange of Product Data. This standard has been developed to allow the exchange of geometric and graphic data between dissimilar CAD/CAM systems.

The **NCL/STEP** converter currently only imports ASCII STEP files. Imported files are converted to **NCL** Unibase files.

NCL/STEP is implemented as a subset of ISO 10303-21 AP214 supporting those entities that allow a direct conversion into **NCL** entity types.

11.1 Startup

To start **NCL/STEP**:

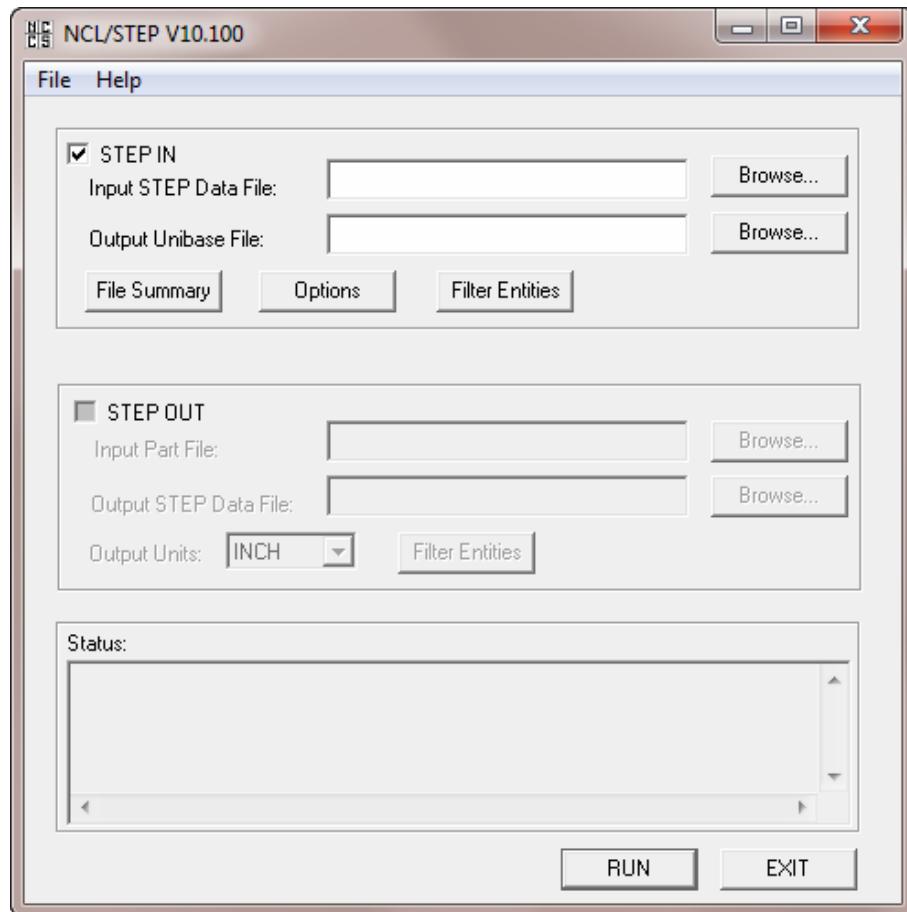
From within **NCL**, click *FILE > STEP*

NCL/STEP can also be started as followed:

Double click the STEP shortcut icon in the **NCCS** folder on the Desktop.

11.2 Importing STEP Files

Start **NCL/STEP** as previously described. The form on the next page will appear:



11.2.1 STEP IN

This is checked by default.

11.2.2 Input STEP Data File:

Enter the name of the STEP file you wish to import. Use the *Browse* button to browse for a file.

11.2.3 Output Unibase File:

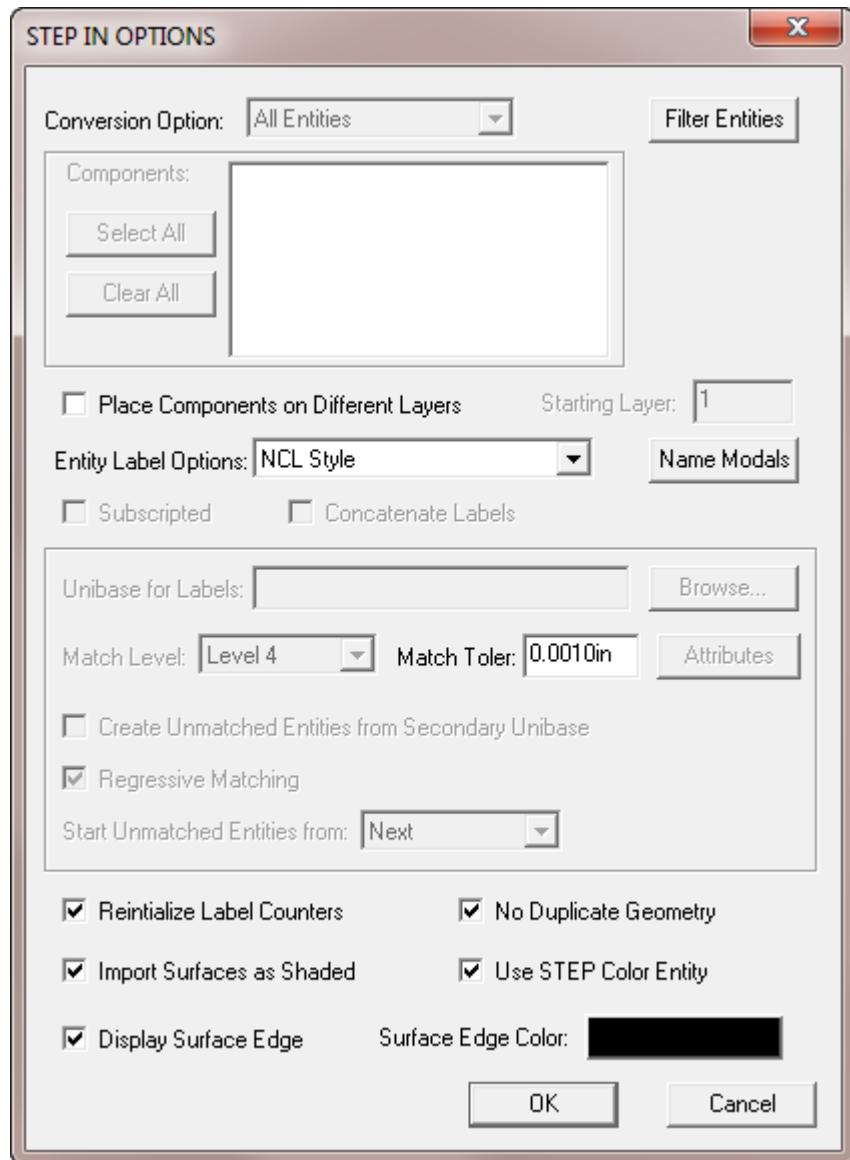
Enter the desired name of the Unibase file which will contain the converted entities. If an existing file is specified it will be overwritten. If an output file is not specified the input file name will be used with a *.u* extension.

11.2.4 File Summary

Click this button to obtain a summary of the entities contained in the STEP file. The summary data will be displayed in the Status Window. This data will also be written to a file named *input.lst*, where *input* is the name of the input STEP file.

11.2.5 Options

Click this button to set various conversion options. The following “STEP IN OPTIONS” form will appear:



Conversion Option

All Entities

Select this option to convert all STEP entities regardless of which component the entities are on.

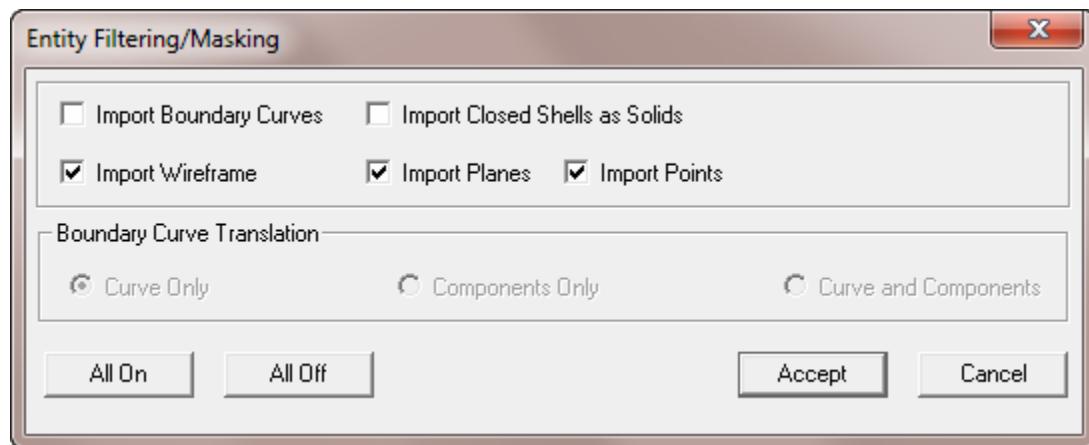
By Components

Select this option to convert entities by component. When this option is selected the Components list box will be activated and a list of components that exist in the input STEP file will be displayed.

Click on the components you wish to convert. Click *Select All* to select all components. Click *Clear All* to clear all previously selected components.

Filter Entities

Click this button to filter out specific entity types from the conversion. The form shown below will appear.



Import Boundary Curves

Check this button will cause **NCL/STEP** to translate the imported surfaces curves.

Import Closed Shells as Solids

Check this button will cause **NCL/STEP** to create composite solids from closed shells. This allows the user to manipulate closed solids imported from a STEP file as a single entity within **NCL**.

Import Wireframe

Check this button will cause **NCL/STEP** to translate the imported wireframe geometry.

Import Planes

Check this button will cause **NCL/STEP** to translate the imported plane entities.

Import Points

Check this button will cause **NCL/STEP** to translate the imported point entities.

Boundary Curve Translation

Curve Only

The surface boundary curve is translated, but the components are not translated independently.

Components Only

The components of the surface boundary curve are translated but the surface boundary curve is not.

Curve and Components

The surface boundary curve and its components are translated.

All On

Press this button to import boundary curves.

All Off

Press this button not to import boundary curves.

ACCEPT

Click this button to accept the filter settings and exit the form.

CANCEL

Click this button to exit the form without saving the filter settings.

Place Components on Different Layers

Click this item to place each component on a different layer and all entities of the same component on the same layer.

Starting Layer

Specify the starting layer number.

Entity Label Option

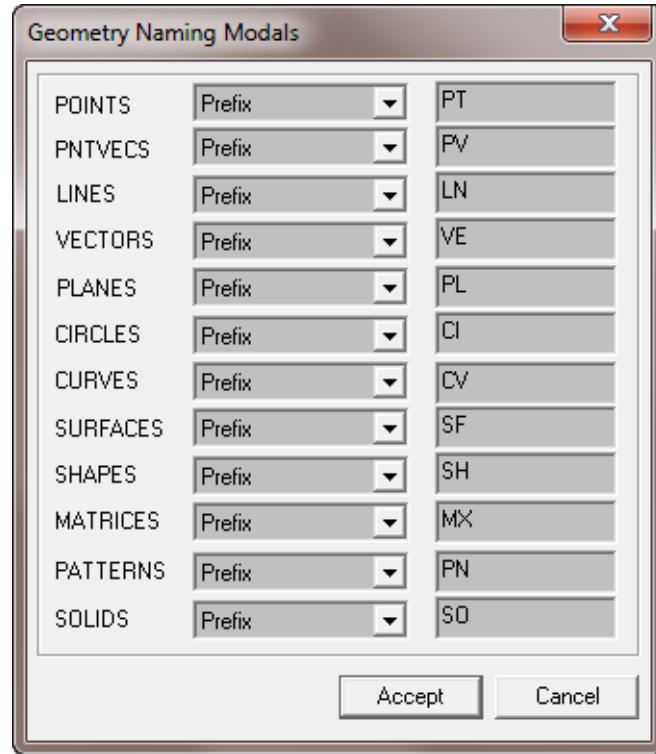
The entity label options are used to determine how the converted entities will be labeled (named). Click this button to produce the form shown on next page.



NCL Style

Click this option if you want **NCL/STEP** to automatically assign a name to the imported entities according to the “Name Modals” setup. This is the default option.

Click the “**Name Modals**” button to open the form as shown on next page.



Choose the desired naming convention (prefix or subscript) by clicking the toggle button next to the entity type.

When Prefix is used entities will be named using the character prefix (up to 20 alpha characters) shown in the box to the right of each entity type and a number, starting at 1. The default setting of this form uses the conventional **NCL** autonaming convention.

For example:

Points	PT1, PT2,...PTn
Lines	LN1,LN2,...LNn
Circles	CI1,CI2,...CIn
Planes	PL1,PL2,...PLn
Curves	CV1,CV2,...CVn
Surfaces	SF1,SF2,...SFn
Patterns	PN1,PN2,...PNn

When Subscript is used entities will be named using the characters (up to 20 alphanumeric characters) shown in the box to the right of each entity type and a subscript number, starting at 1.

For example:

<u>Entity Type</u>	<u>Right Column Entry</u>	<u>Entity Names</u>
Points	PTT	PTT(1),PTT(2),...PTT(n)
Lines	LNN	LNN(1),LNN(2),...LNN(n)
Circles	CII	CII(1),CII(2),...CII(n)
Planes	PLL	PLL(1),PLL(2),...PLL(n)
Curves	CVV	CVV(1),CVV(2),...CVV(n)
Surfaces	SFF	SFF(1),SFF(2),...SFF(n)
Paterns	PNN	PNN(1),PNN(2),...PNN(n)

Click the ACCEPT button to accept the current settings and exit the form.

Click the CANCEL button to exit the form without saving any changes.

STEP File Components

Click this option will use the names assigned to the STEP components to label the entities.

Subscripted

This specifies labels as subscripted names.

Concatenate Labels

This specifies labels created by appending the labels of each individual face (trimmed surface) to the components label.

Following shows what a components label could look like with the various options changed.

Subscripted, No Concatenate:	component(*)
Subscripted, Concatenate:	component_faces(*)
No-Subscripted, Concatenate:	component_face*
No-Subscripted, No Concatenate:	component*

where “*” is an integer represents the subscripted or the non-subscripted value of the generated labels.

STEP File Faces

This specifies using the labels of the individual faces to label the geometry.

Subscripted

This specifies labels as subscripted name.

STEP File Records

This specifies using the prefix *TSF* with a subscript value equal to the record number within the STEP file. When a trimmed surface cannot be created from the STEP record, then the base surface (*BSF*) and boundary curves (*BCV*) will be created instead, with the subscript still being equal to the input face record number. This option is useful in determining the faces that cannot be converted to trimmed surfaces.

From Existing Unibase

Click this option if a STEP file which contains a model that is theoretically the same as an existing unibase with some minor differences. In this case, **NCL/STEP** will compare the entities in the input STEP file with entities in the specified unibase. If the compared entities are exactly the same, then the names used in the existing unibase will be used for the same entities in the STEP file, otherwise a new name will be used.

There are four levels of matching with this option. All entities with an exact match will be translated first. Once all matching entities are translated, **NCL/STEP** will compare the unmatched entities searching for the closest match. Depending on the entity type except “SOLID”, there are up to four levels to determine the closest match: Entity type “SOLID” only allow exact match.

Point:

- Level 1 - Searches all remaining unmatched points for the closest point.
- Level 2:4 - Not used.

Lines:

- Level 1 - The closest line that has the same length and direction.
- Level 2 - The closest line that has the same direction.

- Level 3 - The line with the closest start point.
- Level 4 - Not used.

Circles:

- Level 1 - Circle with the same center point, radius and central angle but with reversed normal vector.
- Level 2 - Circle with the same center point, central angle and normal vector but with different radius.
- Level 3 - Circle with the same radius, central angle and normal vector.
- Level 4 - Closest circle with the same central angle and normal vector.

Planes:

- Level 1 - Plane with the same normal vector and the closest point.
- Level 2:4 - Not used.

Polylines and Patterns:

- Level 1 - Polyline or pattern with all points translated.
- Level 2 - Polyline or pattern with the same number of points.
- Level 3:4 - Not used.

Point-vectors:

- Level 1 - Point-vector with the same length, direction and closest point.
- Level 2 - Point-vector with the same direction and the closest point.
- Level 3:4 - Not used

Curves:

- Level 1 - Closest curve that is an exact match but has been translated.
- Level 2 - Closest curve with same sized bounding box and with end points translated along a vector to a different position.
- Level 3 - Closest curve with end points translated along a vector to a different position.
- Level 4 - Not used.

Rational B-spline Surfaces and Mesh Surfaces:

Planar Surface:

- Level 1 - Planar surface with same normal, same distance and the closest boundary.
- Level 2 - Planar surface with same normal and with the closest boundary.
- Level 3:4 - Not used.

Spherical Surfaces:

- Level 1 - Spherical surface with same center and radius.
- Level 2 - Spherical surface with same center point, but different radius.
- Level 3 - Spherical surface with the closest center point.
- Level 4 - Not used.

Cylindrical Surfaces:

- Level 1 - Closest cylindrical surface with the same radius, axis vector and height.
- Level 2 - Cylindrical surface with the same start point, radius and axis vector but with a different height.
- Level 3 - Cylindrical surface with the same start point and axis vector but with a different height and radius.
- Level 4 - Cylindrical surface with the same axis vector, closest radius.

Cone Surfaces:

- Level 1 - Closest cone surface with the same radius, angle, axis vector and height.
- Level 2 - Cone surface with the same start point, radius, angle and axis vector but with a different height.
- Level 3 - Cone surface with the same start point, angle and axis vector but with a different height and radius.
- Level 4 - Not used.

Freeform Surface:

- Level 1 - Same surface but reversed in U and/or V.
- Level 2 - Closest surface with same shape.
- Level 3 - Surface with same corner points.
- Level 4 - Surface with same corner points to within 5 times tolerance

Trimmed Surfaces:

Base surface goes through the same qualifications as untrimmed surfaces. If the base surface match for the level being tested, then

- Level 1 - Surface with the same number of trimming curves and a bounding box that intersects the bounding box of the surface being matched.
- Level 2 - Surface with a bounding box that intersects the bounding box of the surface being matched.
- Level 3:4 - Not used.

All the remaining unmatched entities will be labeled with the standard prefix for that type of entity, and the number will start one higher than the highest number used for that type of entity. The label of any entity not used in the existing unibase will be displayed in the “*Status*” area of the main form. This information will also be placed into the “.lst” file.

Unibase for Labels

This feature is only activated when the “Using Existing Unibase” option is selected. Enter the file name of the secondary unibase or use the *Browse* button to browse for a secondary unibase file

Match Level

This feature is only activated when the “Using Existing Unibase” option is selected. There are five choices for this toggle button, they are: Exact Match, Level 1, Level 2, Level 3, and Level 4.

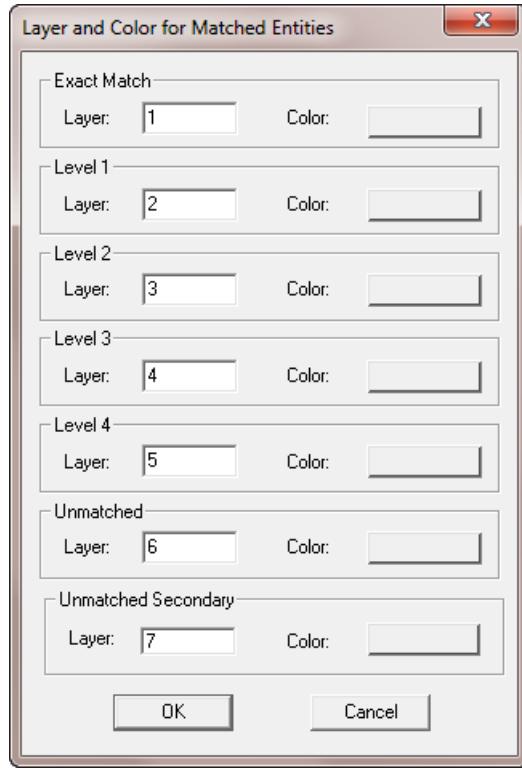
Matching Toler

This feature is only activated when the “Using Existing Unibase” or “No Duplicate Geometry” option is selected. This specifies the tolerance under which two entities are considered to be matched or duplicated. This value will be used as is in “Exact” matching or “No Duplicate Geometry”. This value will be expanded by different internal specified factors for the other levels of matching.

Attribute

This feature is only activated when the “Using Existing Unibase” option is selected. Click this button to open the form as shown on next page.

This form is used to assign the matching/unmatching entities put in different layers and colors according to the matching levels



Create Unmatched Entities from Secondary Unibase

This feature is only activated when the “Using Existing Unibase” option is selected. Check this item to create unmatched entities from the secondary unibase, i.e. the existing unibase.

Rgregative Matching

This feature is only activated when the “Using Existing Unibase” option is selected. Check this item to initialize regressive matching. Regressive matching matches each STEP entity with both the available (unmatch) unibase entities and the other entities of the same type that were previously matched on this level. If this option is not checked STEP would search an entity in the secondary unibase that matches the entity from the STEP file per the rules base on the current level of matching. As soon as a match is found, the new entity is assigned the label of the corresponding entity from the secondary unibase.

Start Unmatched Entities from

This feature is only activated when the “Using Existing Unibase” option is selected. There are two choices for this toggle button, they are: Next or Secondary. This controls how the unmatched entities labels number in the new unibase started. The unmatched entities label number can be started from the next highest entity count or the entity after the highest matched entity in the secondary unibase.

Re-initialize Label Counters

Check this item to re-initialize the automatic label generation counters when translating multiple files in a single STEP run.

No Duplicate Geometry

Check this item will allow only one copy of duplicate entities to be translated.

Import surfaces as shaded

When this item is checked, imported surfaces will be flagged as shaded. When the resulting Unibase is loaded into **NCL** and *View > Modals > Shading* is set to *On*, the imported surfaces will be shaded. Otherwise the imported surfaces must be set to *shaded* while in **NCL**.

Use STEP Color Entity

When this item is checked, imported entities will be assigned with color specified by the Color Entity, otherwise the default **NCL** colors will be assigned to the imported entities. If the color specified in the Color Entity is not supported by **NCL**, the closest matched **NCL** color will be used.

Display Surface Edge

When this item is checked, imported surfaces will be rendered with their edges displayed.

Surface Edge Color

Defines the color to display surface edges with. 'Default' uses the same color as the surface is displayed in, while the other choices select an actual color.

OK

Click this button to accept the changes and close the **NCL/STEP In** Options form.

Cancel

Click this button to cancel all the changes and close the **NCL/STEP In** Options form.

11.2.6 RUN

Click this button to begin the conversion. STEP header information, percentage of file processed, and translation results will be displayed in the status window. The header and result data is also written to the *output.lst* file, where *output* is the name of the output Unibase file.

11.2.7 EXIT

Click this button to exit **NCL/STEP**.

11.3 Viewing The Translated File

From within **NCL** click *FILE > LOADU* and click on the **NCL/STEP** created Unibase. The Unibase file can also be loaded using the *UBFN* or *LOADU* command within a **NCL** part program file.

11.4 User Definable Maximum Parametric Record Size

In order to help solve translation problems associated with large *Parametric Surfaces*, the default parametric record size is user definable.

The following environment variable shown with it's default value is located in the *NCCS/NCL101/interface/nclstep.init* file.

MAX_PAR_REC=1000000

If an error occurs during the processing of an STEP file referring to "parametric record size", simply increase the value shown above and try to process the file

again. *MAX_PAR_REC* environment variable can also be defined in the user's *user.init* file.

11.5 Exporting STEP Files

Currently this is not supported and will be available in future.

12 LATHE MODULE

The **NCL** Lathe Module is implemented as an optional add-on module to the **NCL** processor. In order to use any of its features you must have a valid SPK (Software Product Key) for the Lathe Module.

12.1 Defining A Shape

The NSHAPE statement allows the definition of the entire shape of the part to be defined as a single geometric entity. This is done by defining the part shape first as a series of lines, circles and curves and then combining these entities together in a logical sequence. Postprocessor commands are also allowed within the NSHAPE definition. However, these postprocessor commands will be acted upon only when the shape is used in a [LATHE/FINISH](#) statement.

A shape is a two dimensional entity. Since tool motion along the shape will always be in a right to left (tailstock to headstock) direction, the lines, circles and curves should be input to the shape in that same direction. The valid syntax for the NSHAPE statement is as follows:

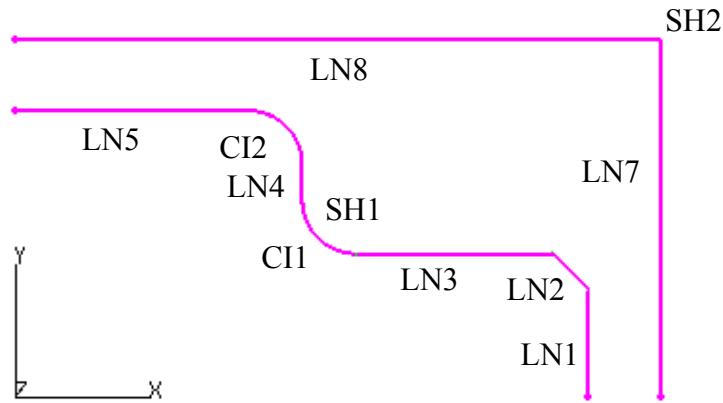
```
NSHAPE/line ,...,line  
          circle    circle  
          curve    curve
```

Example NSHAPE statements:

```
NSHAPE/LN1,CI1,LN2,CI2,CI3,LN3  
NSHAPE/LN1,LN2,LN3,FEDRAT/IPM,30,LN4,LN5
```

The illustration on next page shows the resulting shapes generated from the following two statements:

```
SH1=NSHAPE/LN1,LN2,LN3,CI1,LN4,CI2,LN5  
SH2=NSHAPE/LN7,LN8
```



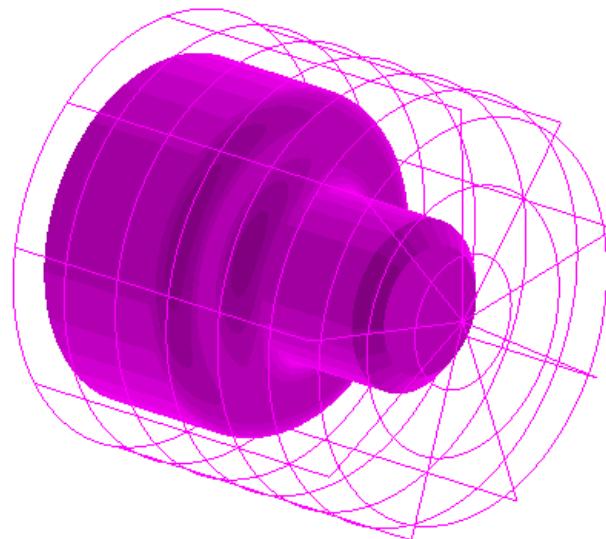
During an interactive session you can use the following menus to modify the attributes of subsequently defined or existing shapes:

TOOLS > LATHE > MODALS

or

TOOLS > LATHE > MOD ATTRIB

Following is a picture showing shapes displayed in the 3-D shaded and 3-D wire frame modes.



12.2 Rough Cutting The Shape

The LATHE/ ROUGH statement allows the automatic roughing of a SHAPE. Two shapes are input to the LATHE/ ROUGH statement. The first shape describes a blank stock and the second describes the shape of the part. The LATHE/ ROUGH statement then generates area clearance cycles to remove excess stock from the material shape. A cycle consists of a move to depth, a cutting pass, retraction from the material and finally a return for the next pass. The valid syntax for the LATHE/ ROUGH statement is:

```
LATHE/ROUGH, shp-id1[, CLDIST, scalar-1], shp-id2      $  
          [, STOCK, scalar-2[, scalar-3]] [, pst-cmd],      $  
          DEPTH, scalar-4[, pst-cmd], CUTANG, scalar-5      $  
          [, pst-cmd], RETRCT, scalar-6, scalar-7      $  
          [, pst-cmd] [, RETURN[, rtn-lctr[, pst-cmd]]]
```

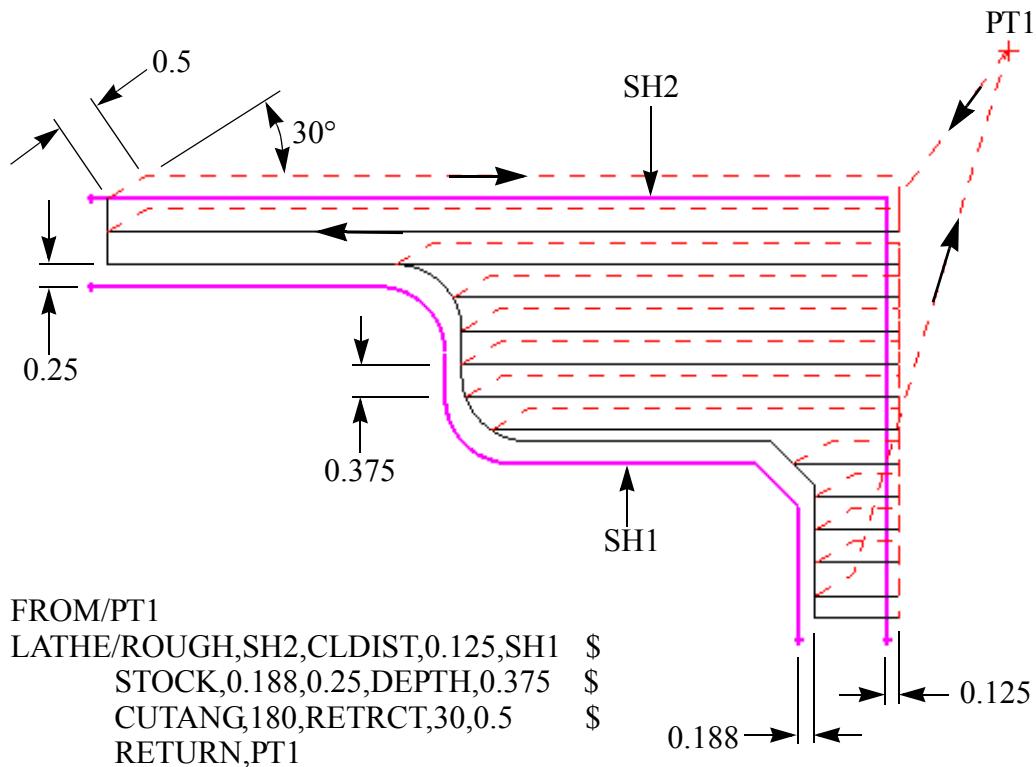
Where:

shp-id1	Specifies the NSHAPE that defines the material shape.
CLDIST	Defines the optional clearance envelope specified by scalar-1 about the blank stock. If this is omitted, the clearance distance will be 0.
shp-id2	Specifies the NSHAPE that defines the part shape.
STOCK	Defines the amount of material to be left along the part shape at the completion of the roughing operation. If omitted then the amount left for the finishing operation will be zero.
scalar-2	Specifies the amount of material to be left in the x direction.
scalar-3	Specifies the amount of material to be left in the y direction. If scalar-3 is omitted, then the value of scalar-2 will be used.
pst-cmd	Indicates an optional postprocessor statement.
DEPTH	Is the maximum depth of each cutting pass specified by scalar-4. This parameter is required.

CUTANG	Is the angle of the cut vector as measured from the positive axis of rotation. Currently the only valid value for scalar-5 is 180. This parameter is required.
RETRCT	Defines the angle specified by scalar-6 at which the tool will disengage from the work at each pass.
scalar-7	Specifies the distance that the tool will travel along the angle.
RETURN	Specifies the action to be taken upon completion of the final roughing pass.
rtn-lctr	May be OFF, XAXIS, YAXIS or a point. OFF causes the tool make no motion after the roughing operation, i.e. after the last retract. XAXIS causes the tool to return to the start point along the X-axis only. There will be no return motion along the Y-axis. YAXIS causes the tool to return to the start point along the Y-axis only. There will be no return motion along the X-axis. If a point is specified, the tool will return to that point. If RETURN is specified without rtn-lctr specified, the tool will return to the start point. If both RETURN and rtn-lctr are omitted, the condition is equivalent to "RETURN,OFF".

12 LATHE MODULE

Example LATHE/ ROUGH statements:



12.3 Finish Cutting The Shape

The LATHE/ FINISH statement causes the finish turning operation to occur on a given NSHAPE. A finish pass starts at the tailstock end of the part shape and travels along the shape to the headstock end. The valid construct for the LATHE/ FINISH statement is:

```
LATHE/FINISH, shp-id1 [, STOCK, scalar-1 [, scalar-2]]      $  
          [, INVERS] [, pst-cmd], ENGAGE, scalar-3, scalar-4    $  
          [, pst-cmd] [, RETRCT, scalar-5, scalar-6]           $  
          [, pst-cmd] [, RETURN[, rtn-lctr]] [, pst-cmd]
```

Where:

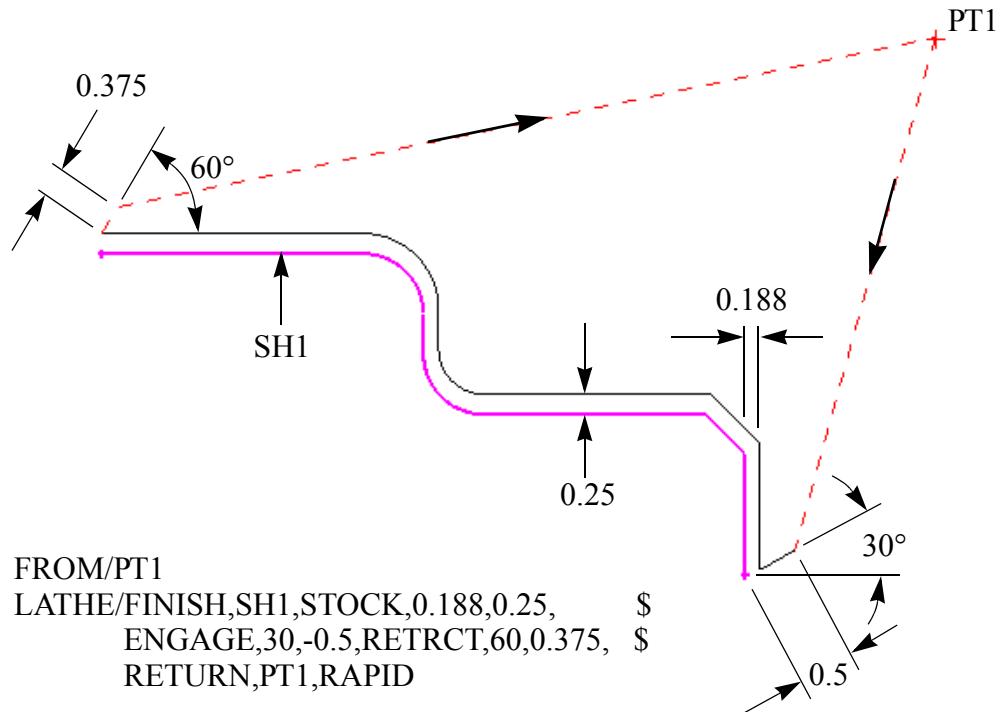
shp-id1	Specifies the part shape.
STOCK	Defines the amount of material to be left along the part shape at the completion of the finishing operation.
scalar-1	Specifies the amount of material to be left in the x direction.
scalar-2	Specifies the amount of material to be left in the y direction. If scalar-2 is omitted, then the value of scalar-1 will be used. If both are omitted then the amount left for the finishing operation will be zero.
INVERS	Specifies that the direction of the pass will be in the positive x direction from the headstock to the tailstock.
pst-cmd	Indicates an optional postprocessor statement.
ENGAGE	Specifies the approach of the tool to the shape. Scalar-3 specifies the angle of the engaging motion vector measured from the positive axis of rotation. Scalar-4 specifies the magnitude of the engaging motion vector. Caution: Since Scalar-4 is the magnitude of the engaging motion vector that the value is not always positive. Normally, this value should be a negative number.

12 LATHE MODULE

RETRCT	Defines the disengagement of the tool from the shape at the end of the cutting sequence. Scalar-5 specifies the angle of the disengagement vector measured from the positive axis of rotation. Scalar-6 specifies the length of the disengagement vector.
RETURN	Specifies the action to be taken upon completion of the cutting pass or, if selected, the retract motion.
Rtn-lctr	May be OFF, XAXIS, YAXIS or a point. OFF causes the tool make no motion after the finishing operation, i.e. after the last retract. XAXIS causes the tool to return to the start point along the X-axis only. There will be no return motion along the Y-axis. YAXIS causes the tool to return to the start point along the Y-axis only. There will be no return motion along the X-axis. If a point is specified, the tool will return to that point. If RETURN is specified without rtn-lctr specified, the tool will return to the start point. If both RETURN and rtn-lctr are omitted, the condition is equivalent to "RETURN,OFF".

Note: The postprocessor word "RAPID" must be appended to the end of the LATHE/FINISH statement, if the return motion required to be a rapid move. Otherwise, normal feedrate motion will be generated.

Example LATHE/ FINISH Statements



13 TOOL LIBRARY

13 NCL/TOOLIB

NCL/TOOLIB allows you to create and maintain a library of tools.
NCL/TOOLIB allows the following information to be stored for each tool:

- Tool number
- Tool description
- Cutter definition (mathematical definition used for tool motion calculation)
- Cutter display definition (used for graphic display)
- Cutter display parameters (moving or instanced, partial segments or full segments)
- **NCL/CADD** symbol to be used as the displayed cutter or tool holder.
- **NCL/CADD** drawing file to be associated with this tool (used to display a sketch of the tool)
- User defined command to be output each time this tool is loaded

13.1 Startup

To start **NCL/TOOLIB**:

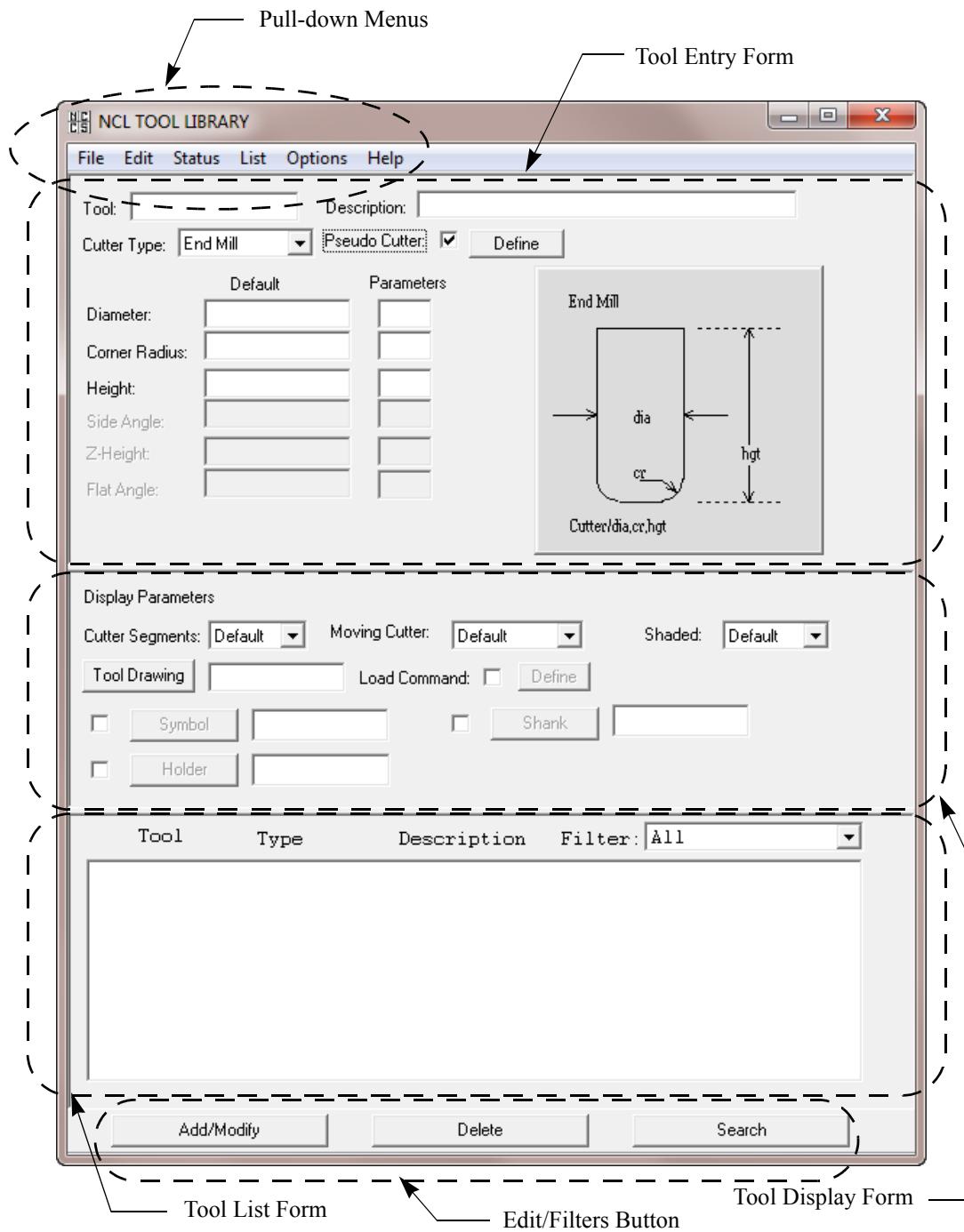
From within **NCL**, click TOOLS >CAM > TOOL LIBRARY

NCL/TOOLIB can also be started in the following way:

Double click the “TOOLIB” short cut icon in the “NCCS” folder of the desktop window.

13.2 Using **NCL/TOOLIB**

Startup **NCL/TOOLIB** without using any runtime options. The following form will appear:



13 TOOL LIBRARY

This form is composed of five sections: Pull-down Menus, Tool Entry Form, Tool Display Form, Tool List Form, and Edit/Filter-Buttons.

Pull-Down Menus:

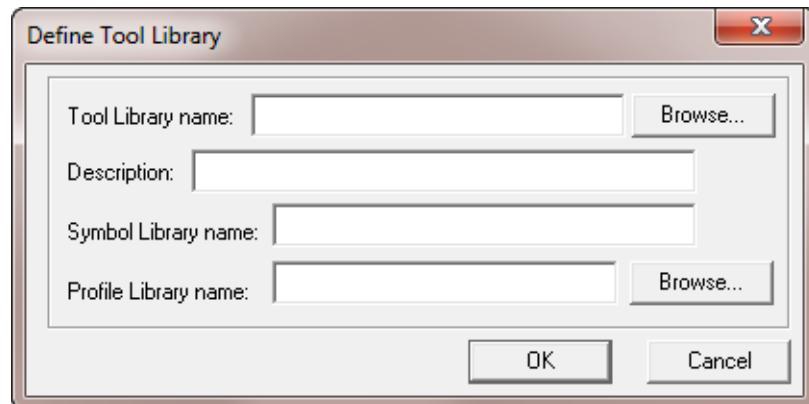
This section is composed of five items: File, Edit, Status, List, Options, Help. They are described as follows:

File:

There are five choices (New, Open, Properties, Save, Exit) under this menu:

- **New:**

Click this option to create a new tool library. The “Define Tool Library:” Form shown below will appear. If the currently loaded library has been modified, the user will be given the option to save the currently loaded library before a new one will be created. There *must be at least one alpha character* in the tool library name.



Click on the “BROWSE” button to open a file browser or enter a name for the new tool library.

Enter a description for the tool library.

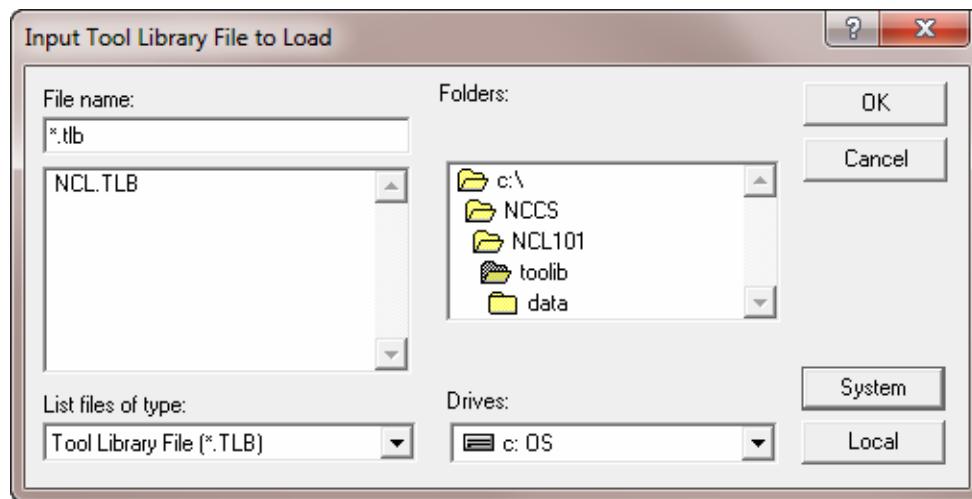
Enter the name of the **NCL/CADD** symbol library to be used with this tool library.

Enter the name of the profile library to be used with this tool library.

Click *ACCEPT* to accept the changes. Click *Cancel* to exit this form and not accept the changes.

- **Open:**

Click this option to load an existing tool library. The browser shown below will appear. If the currently loaded library has been modified, the user will be given the option to save the currently loaded library before a new one will be loaded.



Click *System* to locate a library in the **NCL/TOOLIB** system directory.

The system directory path is defined by the environment variable *NCL_TOOL* in the file:

C:/NCCS\NCL101\interface\ncl.init

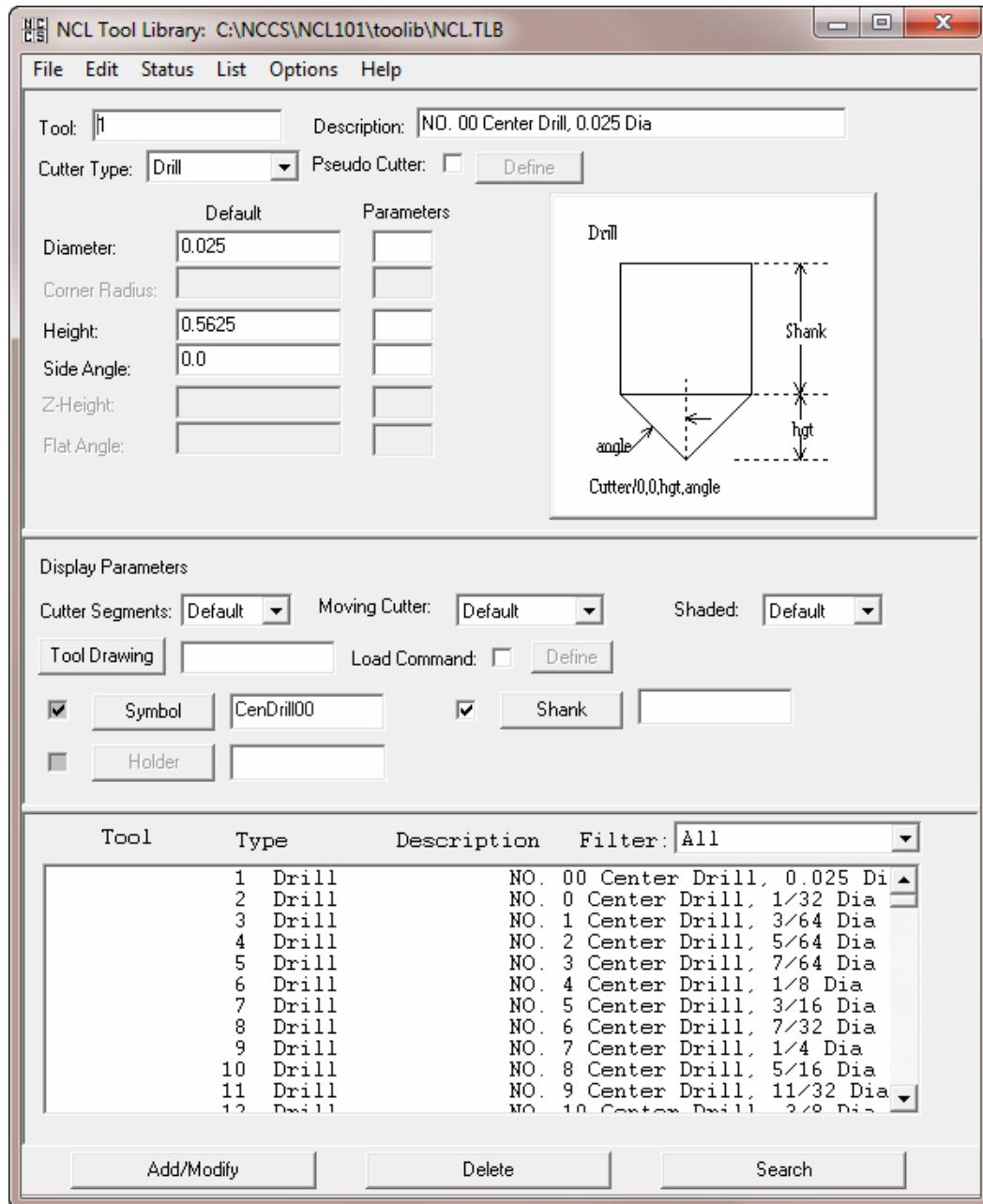
The default path is:

C:/NCCS\NCL101\toolbar

Click *Local* to locate a library in the local directory structure.

13 TOOL LIBRARY

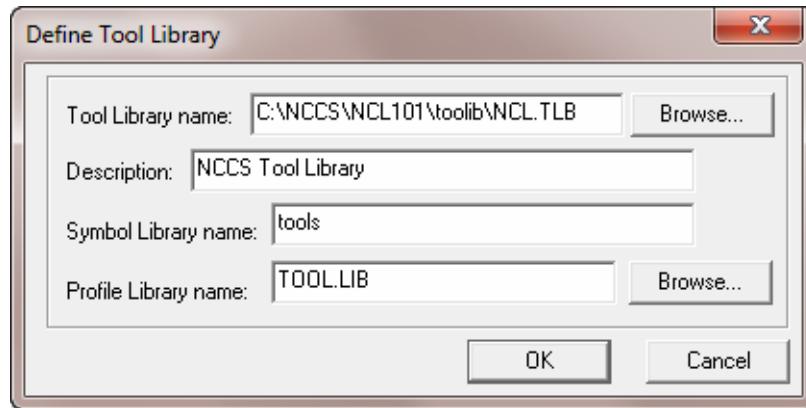
Click on the tool library file that the user wishes to load. Upon loading the library a list of tools will be displayed at the bottom section of the **NCL Tool Library Form**. For example:



Highlight a tool to display the information for that tool in the *Tool Entry Form* and the *Tool Display Form*.

- **Properties**

Click this option to change the description, the tool library name or the name of the **NCL/CADD** symbol library to use with this library. The following form will appear:



Enter the desired name for the tool library.

Enter a description for the tool library.

Enter the name of the **NCL/CADD** symbol library to be used with this tool library.

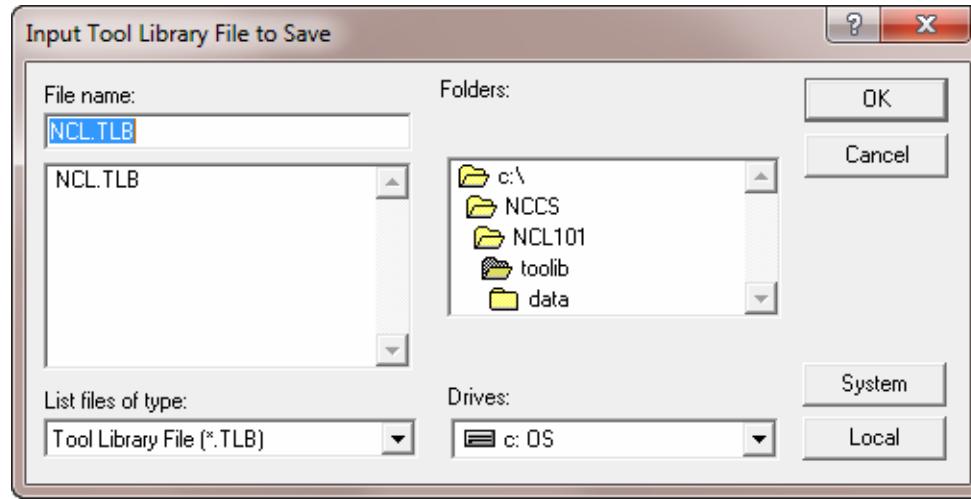
Enter the name of the profile library to be used with this tool library.

Click *OK* to accept the changes. Click *Cancel* to exit this form and not accept the changes.

- **Save**

Click this option to save the newly created tool library or the currently opened tool library. The browser shown on next page will appear. There *must be at least one alpha character* in the tool library name.

13 TOOL LIBRARY



If no file name extension is specified, the tool library file will be saved as a binary file which is platform dependent. Specify the file name extension “tla” to save the tool library file in ASCII format, i.e. a text file. This file is platform independent. The advantage of saving the tool library file in ASCII is the file can be transferred from one platform to a different platform. The disadvantage is it will take longer time to open an ASCII tool library file.

Click **System** to save the library in the **NCL/TOOLIB** system directory.

The system directory path is defined by the environment variable **NCL_TOOL** in the file:

C:/NCCS\NCL101\interface\ncl.init

The default path is:

C:/NCCS\NCL101\toolib

Click **Local** to save the library in the local directory structure.

- **Exit**

Click this option to exit **NCL/TOOLIB**. This will also remove all the internal generated temporary files used for this session.

Edit

There are three choices (Add/Modify, Delete, Search) under this menu:

- **Add/Modify**

Use this option to add/modify a tool to the current tool library. The information currently appearing in the *TOOL ENTRY Form* is used as the input data. This option is the same as the “*Add/Modify*” button at the bottom of the **NCL Tool Library Form**.

If the tool number being added already exists in the library then the information about the tool will be updated to reflect the information currently in the *TOOL ENTRY Form*.

Add/Modify must be performed after the data in the *TOOL ENTRY Form* is modified, otherwise the current tool library will not be updated.

- **Delete**

Use this option to delete a tool (the one highlighted in the *Tool List Form*) from the current tool library file. This option is same as the “*Delete*” button at the bottom of the **NCL Tool Library Form**.

- **Search**

Use this option to search for a particular tool in the tool library. The user can search by tool number or enter a string of text and the tool description field will be searched. This option is the same as the “*Search*” button at the bottom of the **NCL Tool Library Form**.

This option will open a search form as shown below.

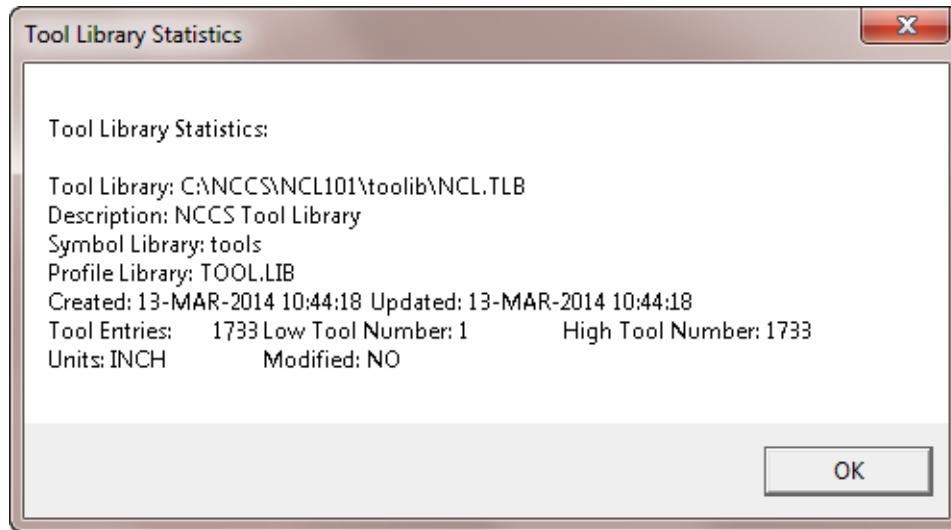


13 TOOL LIBRARY

Enter a text string that needs to be searched and click the *Find* button to begin the search. Click *Cancel* to exit the search process.

Status

Click this button to view statistical information about the current library. For example:



List

There are three choices under this menu: Brief, Full and Modify List.

- Brief**

Click this option to create a brief listing (tool number and description) of each tool in the library.

- Full**

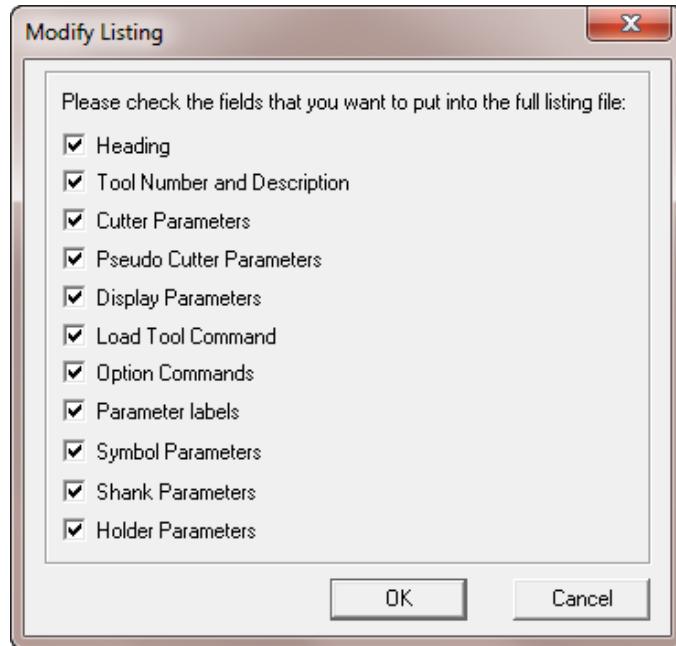
Click this option to create a full listing of each tool in the library.

With either option a file browser will appear allowing the user to enter or select a file name.

- Modify List**

Use this option to modify the format of the Full List File.

This option will open a modify list form as shown on next page.



Click the *OK* button to accept the change and the *Cancel* button to reject the change. Click either button will close the form.

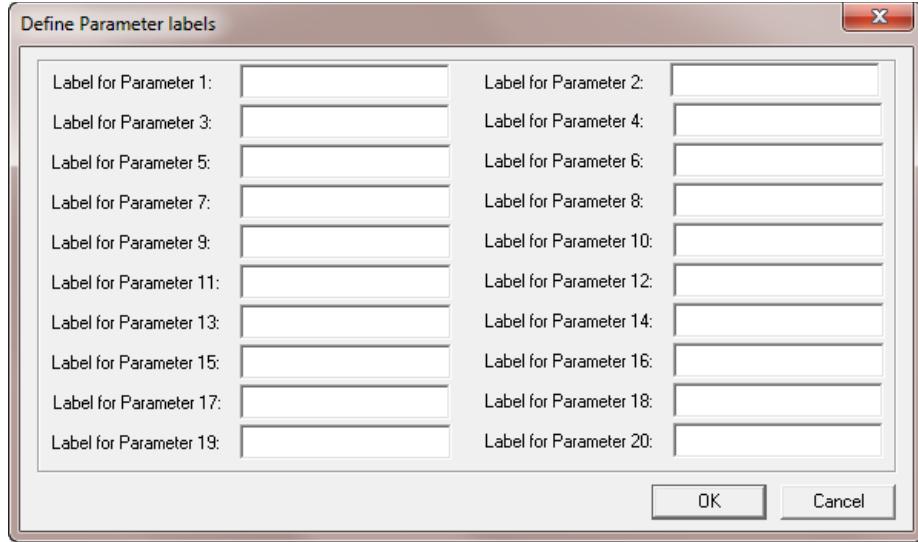
Options

There are two choices under this menu: Parameter Labels, Optional Commands.

- **Parameter Labels**

Use this option to associate label names with the override Parameters. The override parameters allow the user to override the default value for the associated field when a tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu. See both the Pseudo Cutter and the Cutter Parameter descriptions in the *TOOL DISPLAY FORM* for details.

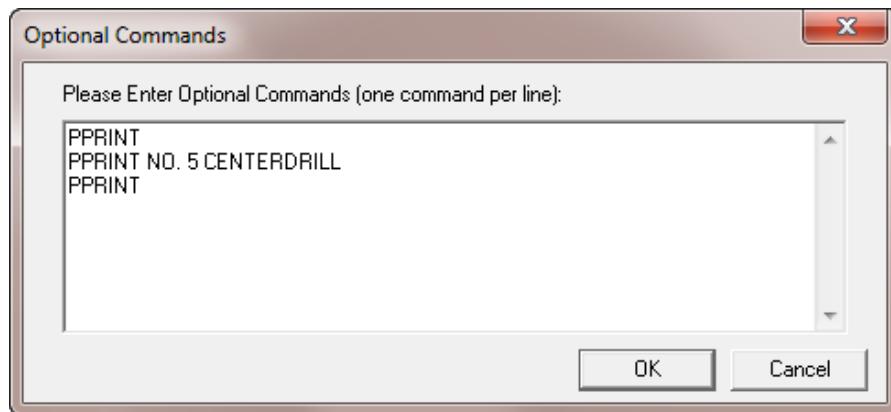
13 TOOL LIBRARY



- **Optional Commands**

Use this option to output optional commands when a specified tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

A text entry window will open as shown below. Just type in the optional commands in this window. Click the *OK* button to accept the entries or the *Cancel* to reject the entries. Click either button will close this window.



Help

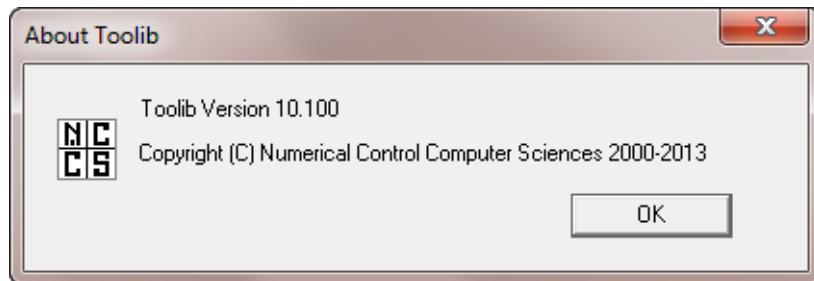
There are two choices under this menu: Help, About.

- **Help**

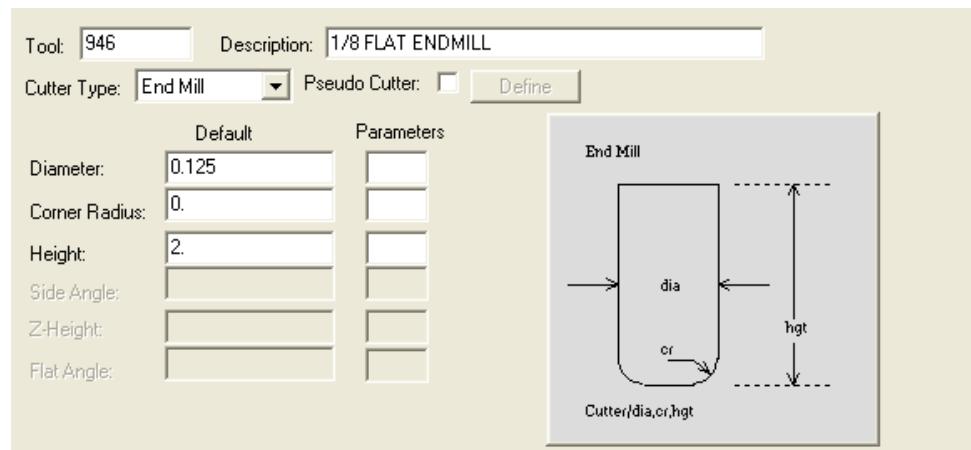
Click on this menu will bring up this document.

- **About**

Click on this menu will bring up the following form which shows the version number and the copyright notice.



The TOOL ENTRY Form



The following gives an description of each field in this form:

Tool

Enter a tool number (must be a positive integer and maximum of 15 digits).

Description

Enter a description for the tool.

13 TOOL LIBRARY

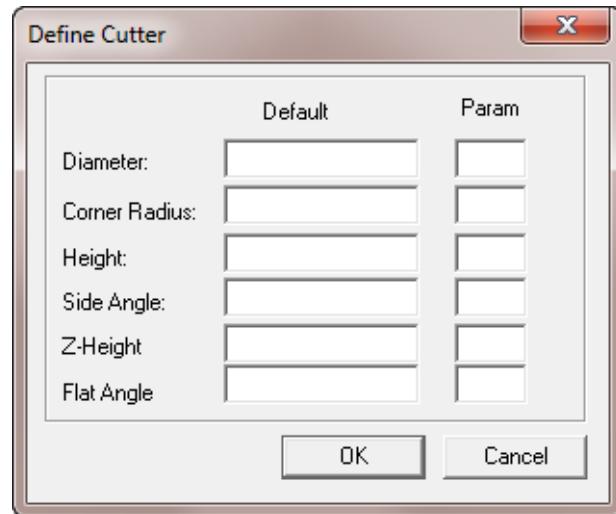
Cutter Type

Toggle this button to toggle between different types of cutter. This will activate the allowable cutter parameters fields and change the cutter picture display accordingly.

Pseudo Cutter

This option is used to define a pseudo cutter. During tool motion, this pseudo cutter will be used for calculation instead of the default cutter if defined.

Check the square button to define a pseudo cutter if it has not been defined. Or click the “Define” button to modify the pseudo cutter if it has been defined. Both action will open the form as shown below.



Default

Use this column to enter the mathematical description of the cutter for tool path calculation.

Param

Each field in the Default columns have a *Param* field associated with them. The *Param* fields allow the user to override the default cutter entries (diameter, corner radius, etc.) when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

To allow an entry to be overridden, a parameter number in the *Param* field associated with the entry must be specified. **NCL/TOOLIB** supports up to 20 override parameters, thus the number entered must be a number between 1 and 20.

For example, if the user wants the ability to override the diameter of the *Actual* cutter, then the user could enter a 1 in the *Param* field which is directly to the right of the *Diameter* field.

When this tool is loaded the user will be given the option to override “parameter #01”. See the section entitled *Accessing tools from the library* for more information about overriding default entries.

Cutter Parameters

Default

Use this column to enter the mathematical description of the cutter for motion calculation and/or display purpose. If a pseudo cutter is not defined, the data in this column will be used to calculate the tool path and for display purpose when this tool is being used. Otherwise, the pseudo cutter will be used for motion calculation, and the parameters defined here will be used for display purpose.

Param

Each field in the Default columns have a *Param* field associated with them. The *Param* fields allow the user to override the default cutter entries (diameter, corner radius, etc.) when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

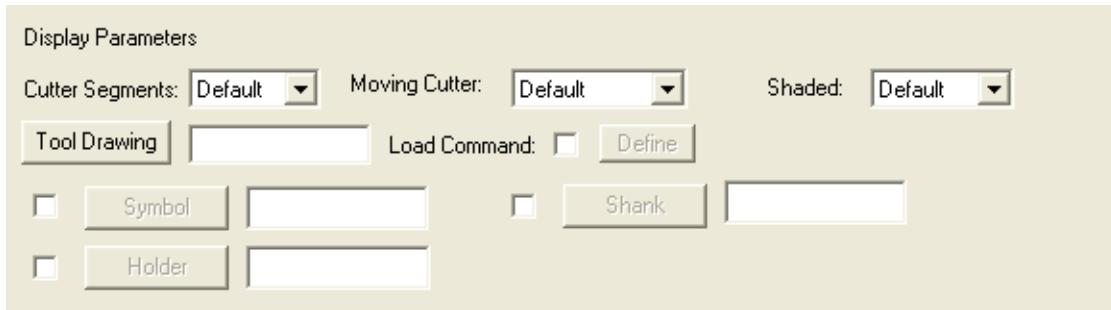
To allow an entry to be overridden, a parameter number in the *Param* field associated with the entry must be specified. **NCL/TOOLIB** supports up to 20 override parameters, thus the number entered must be a number between 1 and 20.

For example, if the user wants the ability to override the diameter of the *Actual* cutter, then the user could enter a 1 in the *Param* field which is directly to the right of the *Diameter* field.

When this tool is loaded the user will be given the option to override “parameter #01”. See the section entitled *Accessing tools from the library* for more information about overriding default entries.

13 TOOL LIBRARY

The Display Parameters Form



Cutter Segments

This button determines how the wireframe image of the tool will be displayed.

Default specifies to use the setting which is currently in effect when this tool is loaded.

Part specifies that only the outside profile of the tool will be displayed.

All specifies that the tool will be displayed using multiple 3D segments.

Moving Cutter

This button determines how tool motion will be animated when using this tool.

Default specifies to use the setting which is currently in effect when this tool is loaded.

On specifies that the tool be displayed as a dynamically moving object.

Off specifies that an instance of the tool will be drawn according to the current setting of *PLAYBACK > MODALS > CUTTER STEPS*.

Shaded

This button determines how the shaded image of the tool will be displayed.

“*Default*” specifies to use the setting which is currently in effect when this tool is loaded.

“*On*” specifies display tool as shaded. Moving tool will have jumpy motion.

“*Off*” specifies display tool as wireframe. This is the fastest display mode.

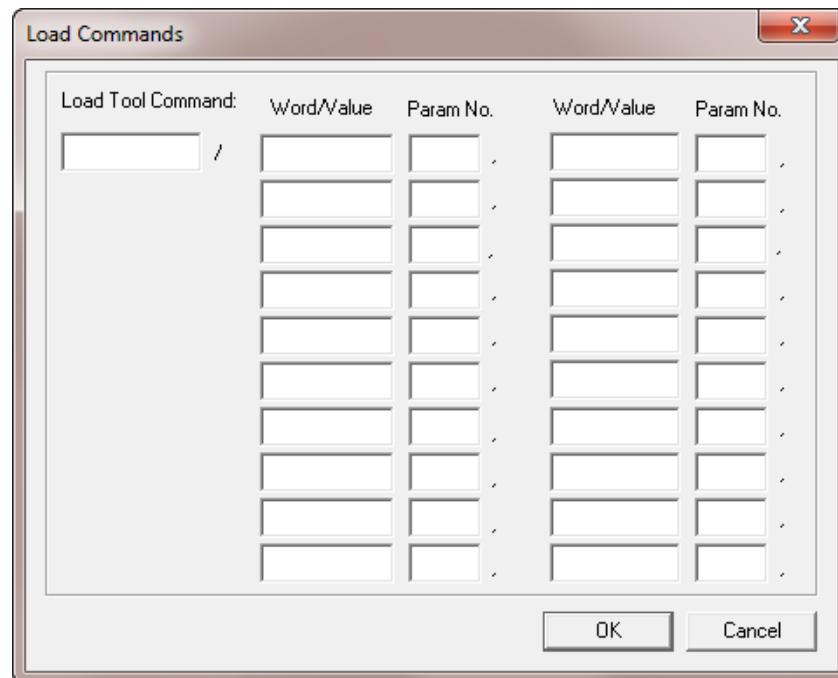
Tool Drawing

Enter the name of a **NCL/CADD** drawing file that contains a drawing of the tool, or click the “Tool Drawing” button to open a file browser to locate the drawing file and load the name into the tool library data base. The drawing can be viewed from within **NCL** by clicking the *View Tool* button on the *Cutter Form* which is opened by the *TOOLS > CAM > CUTTER* menu.

Load Command

This option allows the user to output a user specified command each time this tool is loaded.

Check the square button to define this user specified command if it has not been defined. Or click the “Define” button to modify the command if it has been defined. Both actions will open the form as shown below.



13 TOOL LIBRARY

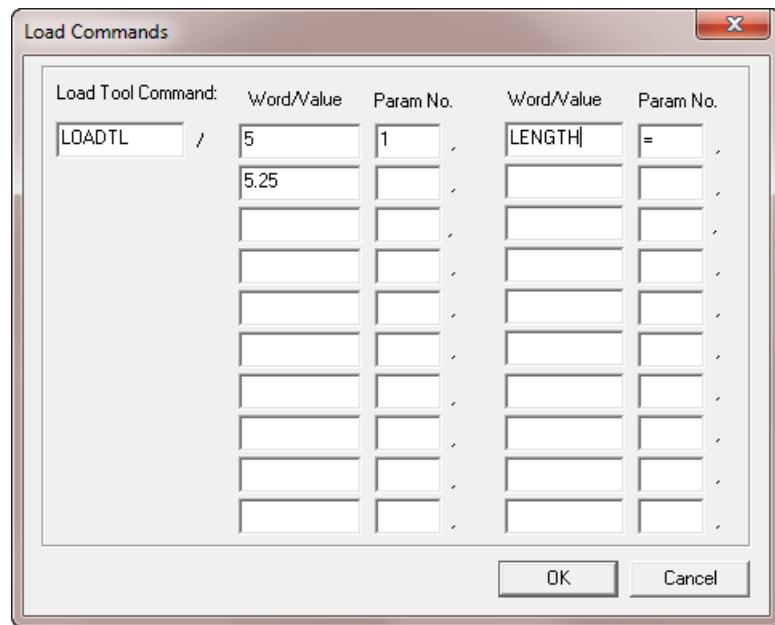
This form composes of fields which are used to define the user specified command. The fields are separated into three types, the Major word fields, Minor word fields, and override parameter fields.

“*Load Tool Command*” column is an user specified command which can be any major word with up to 20 minor words and/or values.

The “*Word/Value*” columns specify the optional minor words or values associated with the major word.

“*Param No*”. columns specify the override parameter which can also be associated with each minor word. The override parameters allow the user to override the default value for the associated field when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

An example is shown below.



The entries above would cause the following statement to be processed each time this tool was loaded from the library:

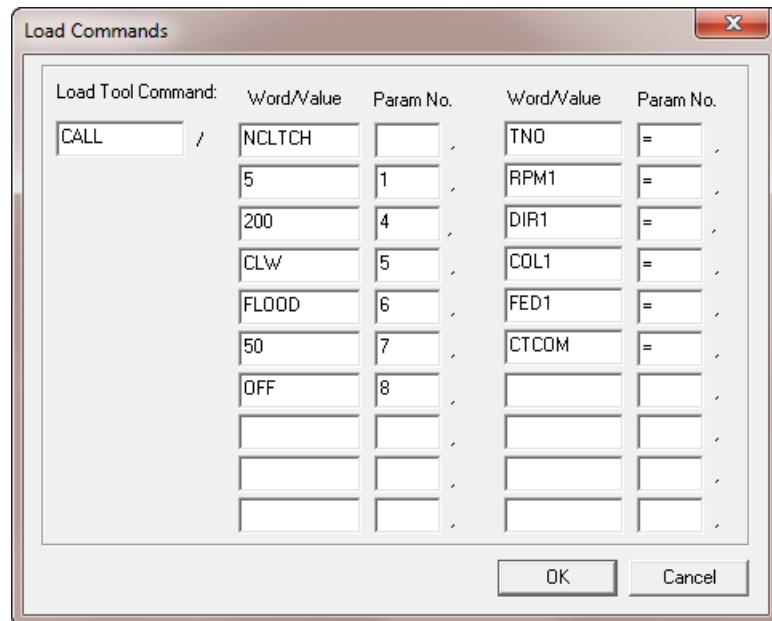
```
LOADTL/5, LENGTH, 5.25
```

The user would have the option of overriding the default tool number (5) and set length (5.25) when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

The user specified command can also be a call to an **NCL** macro. In this case the Major word field would contain the word *CALL*. The first Minor word field would contain the name of the macro. The remaining minor word fields would contain the macro variable names and their default values. The override parameter fields can be used to allow the user to override the associated default value of the macro variable when this tool is loaded. See the section entitled *Accessing tools from the library* for more information about overriding default entries.

Up to 9 macro variable names with default values can be specified.

Shown below is an example of how an entry should be made when calling a macro:



The entries above would cause the following statement to be processed each time this tool was loaded from the library

```
CALL/TCH,TNO=5,RPM1=6000,FED1=75,OFNO=54,$
      CCOM=ON
```

13 TOOL LIBRARY

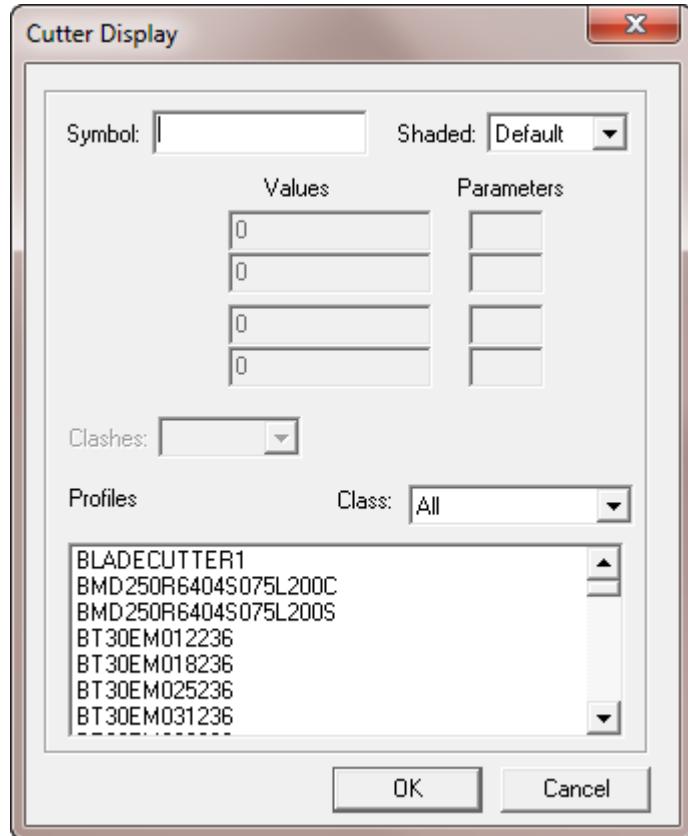
The user would have the option of overriding the default values for the *TNO*, *RPMI*, and *FEDI* variables. The *OFNO* variable would always be given a value of 54. The *CCOM* variable would always be given a value of *ON*.

Macro variable must always be given a default value.

Symbol

This option allows the user to specify a **NCL/CADD** symbol or a profile name in the tool description file to be used as the displayed tool. Check the square button to activate this option.

Enter the name of a **NCL/CADD** symbol, a profile name in the cutter point-list file to be used as the displayed tool. Or click the “Symbol” button to open the Cutter Display Form as shown below.



- **Symbols**

Enter the name of the **NCL/CADD** symbol or the profile name to be used as the cutter image. The name can be entered directly or it can be entered by highlight the profile name listed in the Profiles window at bottom of the Cutter Display form.

- **Shaded**

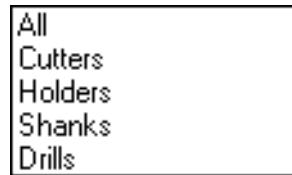
Use this button to specify how the cutter image of the tool to be displayed. “*Default*” specifies to use the setting which is currently in effect when this tool is loaded. “*On*” specifies display cutter as shaded. Moving cutter will has jumpy motion. “*Off*” specifies display cutter as wireframe. This is the fastest display mode.

- **Profiles**

This display a list of all the profiles specified in the cutter profile file. Picks a profile by highlight it on this form. After the profile has been highlighted, the profile name will be displayed in the Symbol entry prompt.

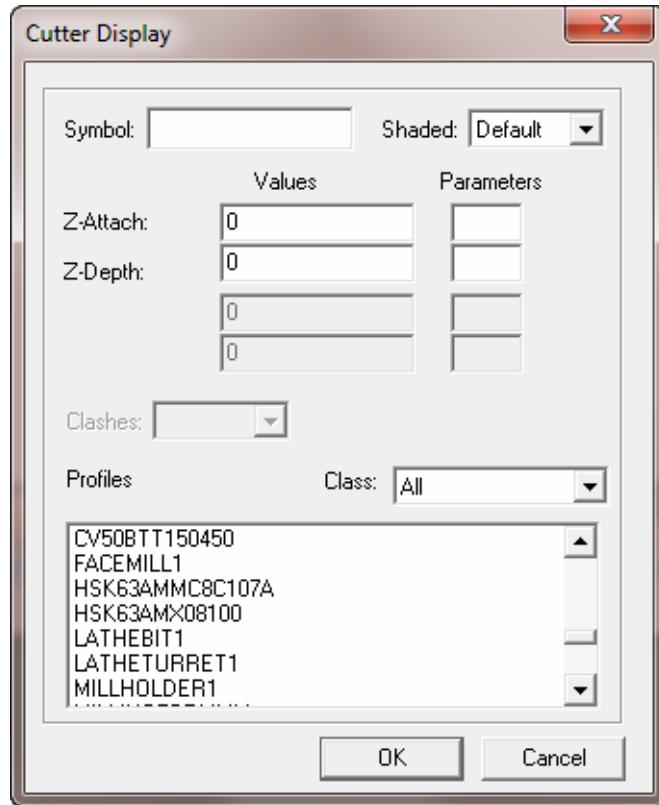
- **Class**

Click this button will open the menu as shown below.



Use this button to display a specific type of profile in the *Profile Window*. Highlight the type and the label of this button will be modified to display the type of profile. Also, the *Profile window* will be updated to only display this particular type of profile.

The Cutter Display form will be changed to as shown on next page if the specified cutter is a lathe insert and a symbol or profile name is specified.



- **Z-Attach**

Defines the starting Z-axis position of the insert. The default starting position is at the bottom of the insert cutter or top of the shank. The specified value corresponds to the “Z-Attach” parameter of the “*OFFSET,Z-Attach,Z-Depth*” parameters of the *CUTTER/LATHE* statement for a lathe style cutter. Enters the specified values in the “Values” column. Specifies the override parameter in the “Parameters” column allows the user to override the default “Z-Attach” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Z-Depth**

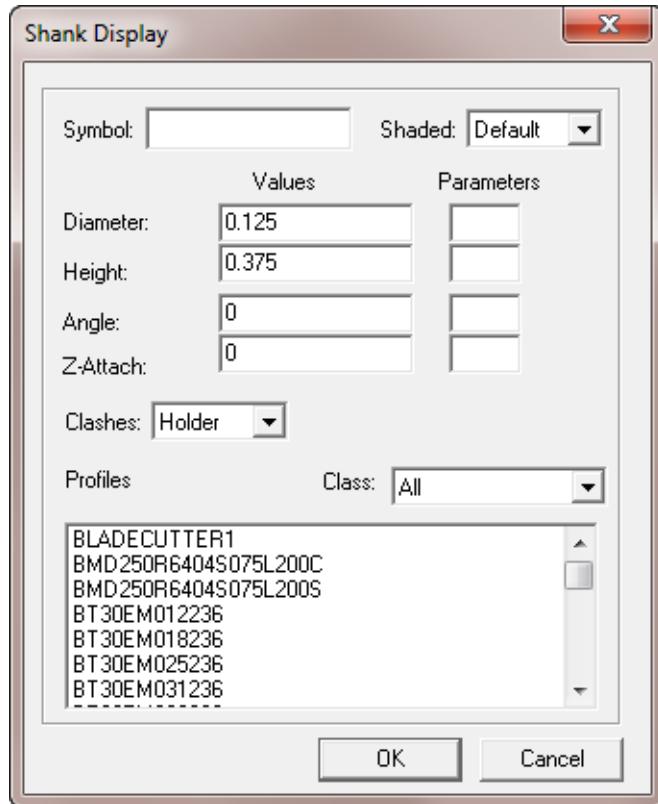
Defines the default depth of the insert cutter. The default depth is the defined cutter height from the *CUTTER/LATHE* statement. The specified value corresponds to the “Z-Depth” parameter of the “*OFFSET,Z-Attach,Z-Depth*” parameters of the *CUTTER/LATHE* statement for a lathe style cutter. Enters the specified values in the “Values” column. Specifies the override parameter in the “Parameters”

column allows the user to override the default “Z-Depth” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

Shank

This option allows the user to specify a **NCL/CADD** symbol or a profile name in the tool description file to be used as the displayed shank. Check the square button to activate this option.

Enter the name of a **NCL/CADD** symbol, a profile name in the cutter point-list file to be used as the displayed shank. Or click the “Symbol” button to open the Cutter Display Form as shown below.



- **Symbols**

Enter the name of the **NCL/CADD** symbol or the profile name to be used as the shank image. The name can be entered directly or it can be entered by highlight the profile name listed in the Profiles window at bottom of the Cutter Display form.

13 TOOL LIBRARY

- **Shaded**

Use this button to specify how the shank image of the tool to be displayed. “*Default*” specifies to use the setting which is currently in effect when this tool is loaded. “*On*” specifies display cutter as shaded. Moving cutter will has jumpy motion. “*Off*” specifies display cutter as wireframe. This is the fastest display mode.

- **Diameter**

Defines the diameter of a Mill style shank. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Diameter*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Height**

Defines the height of a Mill style shank. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Height*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

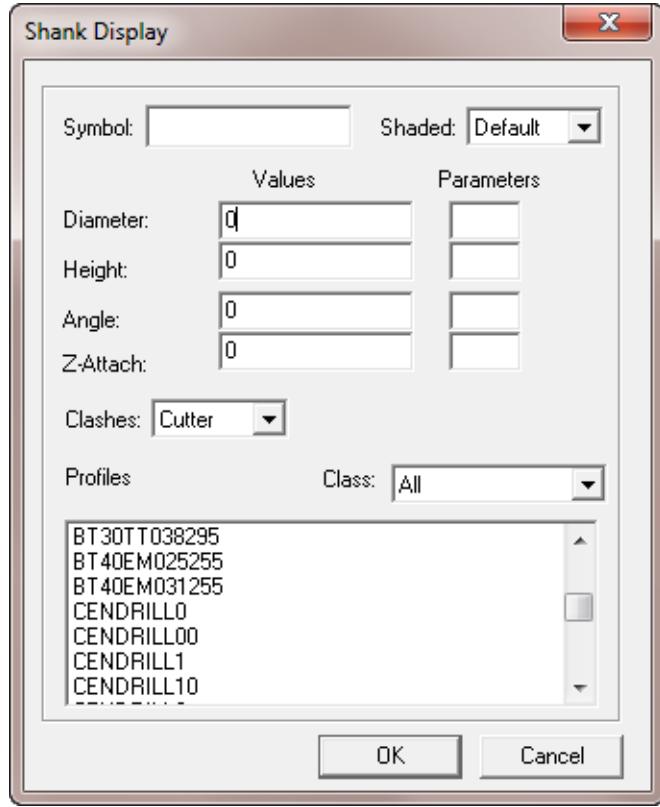
- **Angle**

Defines the side angle of a Mill style shank. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Angle*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Z-Attach**

Defines the attach point of a Mill style shank to the cutter. A value of 0 will attach the shank at the top of the cutter. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Z-Attach*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

The Shank Display form will displayed as shown on next page if a Lathe style cutter is specified.



- **Width**

Defines the width in X of a lathe insert shank. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Width*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Length**

Defines the length in Y of the lathe insert shank. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Length*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Z-Depth**

Defines the depth in Z of the lathe insert shank. Enters the specified values in the “*Values*” column. Specifies the override parameter in the

13 TOOL LIBRARY

“Parameters” column allows the user to override the default “Z-Depth” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Y-Offset**

Defines the Y-offset for the attach point of the lathe insert shank. A value of 0 will attach the shank at the center of the insert’s inscribed circle. Enters the specified values in the “Values” column. Specifies the override parameter in the “Parameters” column allows the user to override the default “Y-Offset” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Clashes**

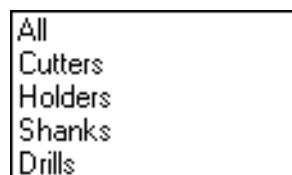
This controls how the shank is treated during the **NCL/IPV** simulation session. Specifying “*Cutter*” causes the tool shank to be treated as a part of the cutting portion of the tool as defined by the cutter parameters. Clashes will only be reported if the tool shank violates a fixture or during Rapid moves. “*Holder*” specifies that the tool shank should be classified as a part of the tool holder, causing clashes to be reported whenever it violates a fixture or a stock.

- **Profiles**

This display a list of all the profiles specified in the cutter profile files. Picks a profile by highlight it on this form. After the profile has been highlighted, the profile name will be displayed in the Symbol entry prompt.

- **Class**

Click this button will open the menu as shown below.

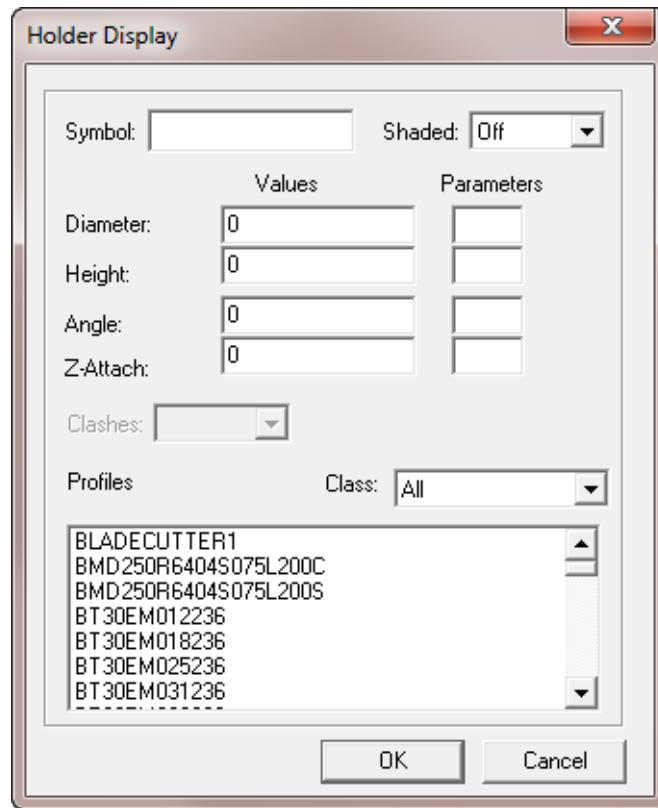


Use this button to display a specific type of profile in the *Profile Window*. Highlight the type and the label of this button will be modified to display the type of profile. Also, the *Profile window* will be updated to only display this particular type of profile.

Holder

This option allows the user to specify a **NCL/CADD** symbol or a profile name in the tool description file to be used as the displayed holder. Check the square button to activate this option.

Enter the name of a **NCL/CADD** symbol, a profile name in the cutter point-list file to be used as the image of the holder. Or click the “Symbol” button to open the Cutter Display Form as shown below.



- **Symbols**

Enter the name of the **NCL/CADD** symbol or the profile name to be used for the holder image. The name can be entered directly or it can be entered by highlight the profile name listed in the Profiles window at bottom of the Cutter Display form.

- **Shaded**

Use this button to specify how the holder image of the tool to be displayed. “*Default*” specifies to use the setting which is currently in

13 TOOL LIBRARY

effect when this tool is loaded. “*On*” specifies display holder as shaded. Moving tool will have jumpy motion. “*Off*” specifies display holder as wireframe. This is the fastest display mode.

- **Diameter**

Defines the diameter of a Mill style holder. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Diameter*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Height**

Defines the height of a Mill style holder. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Height*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

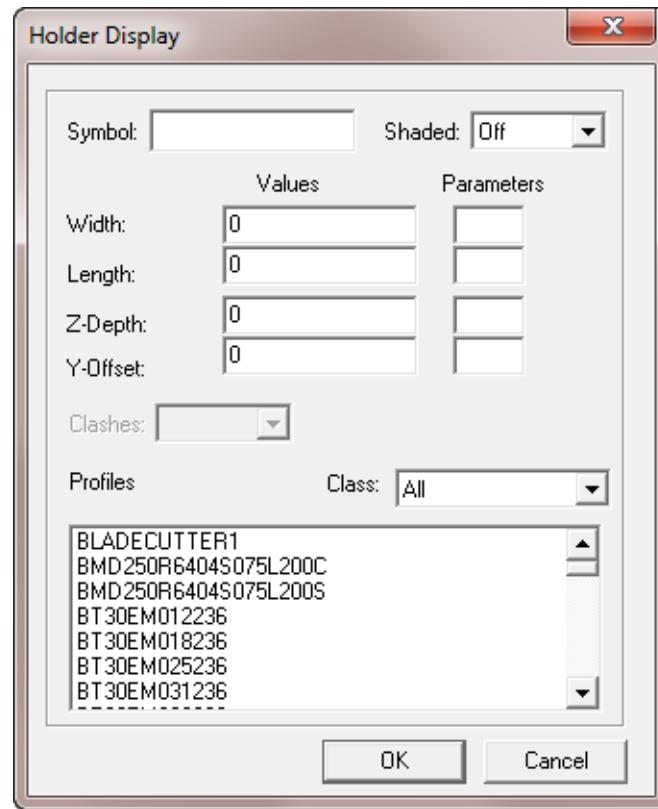
- **Angle**

Defines the side angle of a Mill style holder. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Angle*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Z-Attach**

Defines the attach point of a Mill style holder to the cutter. A value of 0 will attach the shank at the top of the cutter. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Z-Attach*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

The Holder Display form will be displayed as shown on next page if a Lathe style cutter is specified.



- **Width**

Defines the width in X of a lathe insert holder. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “*Width*” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Length**

Defines the length in Y of the lathe insert holder. Enters the specified values in the “*Values*” column. Specifies the override parameter in the “*Parameters*” column allows the user to override the default “I” value when this tool is loaded by either the *CUTTER/TOOL* statement or the *TOOLS > CAM > CUTTER* menu.

- **Z-Depth**

Defines the depth in Z of the lathe insert holder. Enters the specified values in the “*Values*” column. Specifies the override parameter in the

13 TOOL LIBRARY

“Parameters” column allows the user to override the default “Z-Depth” value when this tool is loaded by either the CUTTER/TOOL statement or the TOOLS > CAM > CUTTER menu.

- **Y-Offset**

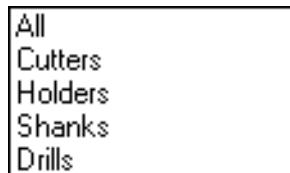
Defines the Y-offset for the attach point of the lathe insert holder. A value of 0 will attach the holder at the center of the holder. Enters the specified values in the “Values” column. Specifies the override parameter in the “Parameters” column allows the user to override the default “Y-Offset” value when this tool is loaded by either the CUTTER/TOOL statement or the TOOLS > CAM > CUTTER menu.

- **Profiles**

This display a list of all the profiles specified in the cutter profile files. Picks a profile by highlighting it on this form. After the profile has been highlighted, the profile name will be displayed in the Symbol entry prompt.

- **Class**

Click this button will open the menu as shown below.



Use this button to display a specific type of profile in the *Profile Window*. Highlight the type and the label of this button will be modified to display the type of profile. Also, the *Profile window* will be updated to only display this particular type of profile.

The Tool List Form

Tool	Type	Description	Filter:
1	Drill	NO. 00 Center Drill, 0.025 Dia	<input type="button" value="^"/>
2	Drill	NO. 0 Center Drill, 1/32 Dia	<input type="button" value="^"/>
3	Drill	NO. 1 Center Drill, 3/64 Dia	<input type="button" value="^"/>
4	Drill	NO. 2 Center Drill, 5/64 Dia	<input type="button" value="^"/>
5	Drill	NO. 3 Center Drill, 7/64 Dia	<input type="button" value="^"/>
6	Drill	NO. 4 Center Drill, 1/8 Dia	<input type="button" value="^"/>
7	Drill	NO. 5 Center Drill, 3/16 Dia	<input type="button" value="^"/>
8	Drill	NO. 6 Center Drill, 7/32 Dia	<input type="button" value="^"/>
9	Drill	NO. 7 Center Drill, 1/4 Dia	<input type="button" value="^"/>
10	Drill	NO. 8 Center Drill, 5/16 Dia	<input type="button" value="^"/>
11	Drill	NO. 9 Center Drill, 11/32 Dia	<input type="button" value="^"/>
12	Drill	NO. 10 Center Drill, 3/8 Dia	<input type="button" value="^"/>
13	Drill	1/8 SPOTDRILL	<input type="button" value="^"/>
14	Drill	1/4 SCREWDRILL	<input type="button" value="^"/>

All the defined tools will be displayed in this form after a tool library is loaded. Picks a tool by highlighting it on this form. After the tool has been highlighted, all the information of the tool will be displayed in the rest of the **NCL Tool Library Form**.

The first column of this form will display tool number corresponding to this tool. The second column displays the type of tool. The last column displays the description of the corresponding tool.

The Edit/Filter Buttons



This is composed of four buttons and they are described below.

Add/Modify:

Click this button to add/modify a tool to the current tool library. The information currently appearing in the *TOOL ENTRY Form* is used as the input data. This option is same as the “ADD/MODIFY” button at the bottom of this form.

If the tool number being added already exists in the library then the information about the tool will be updated to reflect the information currently in the *TOOL ENTRY Form*.

Add/Modify must be performed after the data in the *TOOL ENTRY Form* is modified, otherwise the current tool library will not be updated.

13 TOOL LIBRARY



This button has the same function as the option *Add/Modify* under the *Pull-Down Menu*.

Delete

Click this button to delete a tool (the one highlighted in the *Tool List Form*) from the current tool library file.

This button has the same function as the option *Delete* under the *Pull-Down Menu*.

Search

Click this button to search for a particular tool in the tool library. The user can search by tool number or enter a string of text and the tool description field will be searched.

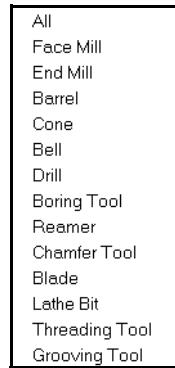
This option will open a search form as shown on next page:

Enter a text string that needs to be searched and click the *Find* button to begin the search. Click *Cancel* to exit the search process.

This button has the same function as the option *Search* under the *Pull-Down Menu*.

Filter

Click this button will open the menu as shown below.



Use this button to display a specific type of tool in the *Tool List Form*. Highlight the type and the label of this button will be modified to display the type of tool. Also, the *Tool List Form* will be updated to only display this particular type of tool.

13.3 Accessing Tools From The Library

13.3.1 Accessing The Library Using The CUTTER/TOOL Statement

Tools are loaded from the library by using the following statement:

CUTTER/TOOL, [lib,]tool[, params]

Where:

lib Is the name (with at least one alpha character) of the tool library from which to load the tool. If *lib* is not specified, then the last specified tool library will be used. The system looks first to the local directory and then to the system directory.

The system directory path is defined by the environment variable *NCL_TOOL* in the file:

C:/NCCS\NCL101\interface\ncl.init

The default path is:

C:/NCCS\NCL101\toolib

13 TOOL LIBRARY

tool	Tool number to load.
params	Optional override parameter list which can be used to override the default value of the <i>TOOL ENTRY</i> fields which have an override parameter associated with them. The order in which the parameters are given correspond with the override parameter number given in the <i>TOOL ENTRY</i> form. That is, the first parameter in the list corresponds with override parameter 1, the second corresponds with parameter 2, and so on. The word <i>SAME</i> can be used to maintain the default value of a field when a list of override parameters is being specified.

The following are some examples of the *CUTTER/TOOL* command:

`CUTTER/TOOL, NCCS, 1`

Tool number 1 from the library “NCCS” will be loaded.

`CUTTER/TOOL, NCCS, 5, 10`

Tool number 5 from the library “NCCS” will be loaded. The default value of the entry associated with override parameter 1 will be overridden and given the value of 10.

`CUTTER/TOOL, NCCS, 7, SAME, SAME, 4000`

Tool number 7 from the library “NCCS” will be loaded. The default value of the entries associated with override parameters 1 and 2 will not be overridden. The default value of the entry associated with override parameter 3 will be overridden and given the value of 4000.

13.3.2 Accessing The Library Using The CUTTER/READ Statement

Tools are loaded from the library by using the following statement:

`CUTTER/READ, [lib,]tool [,params]`

This statement works the same as the CUTTER/TOOL statement with the following modifications and additions.

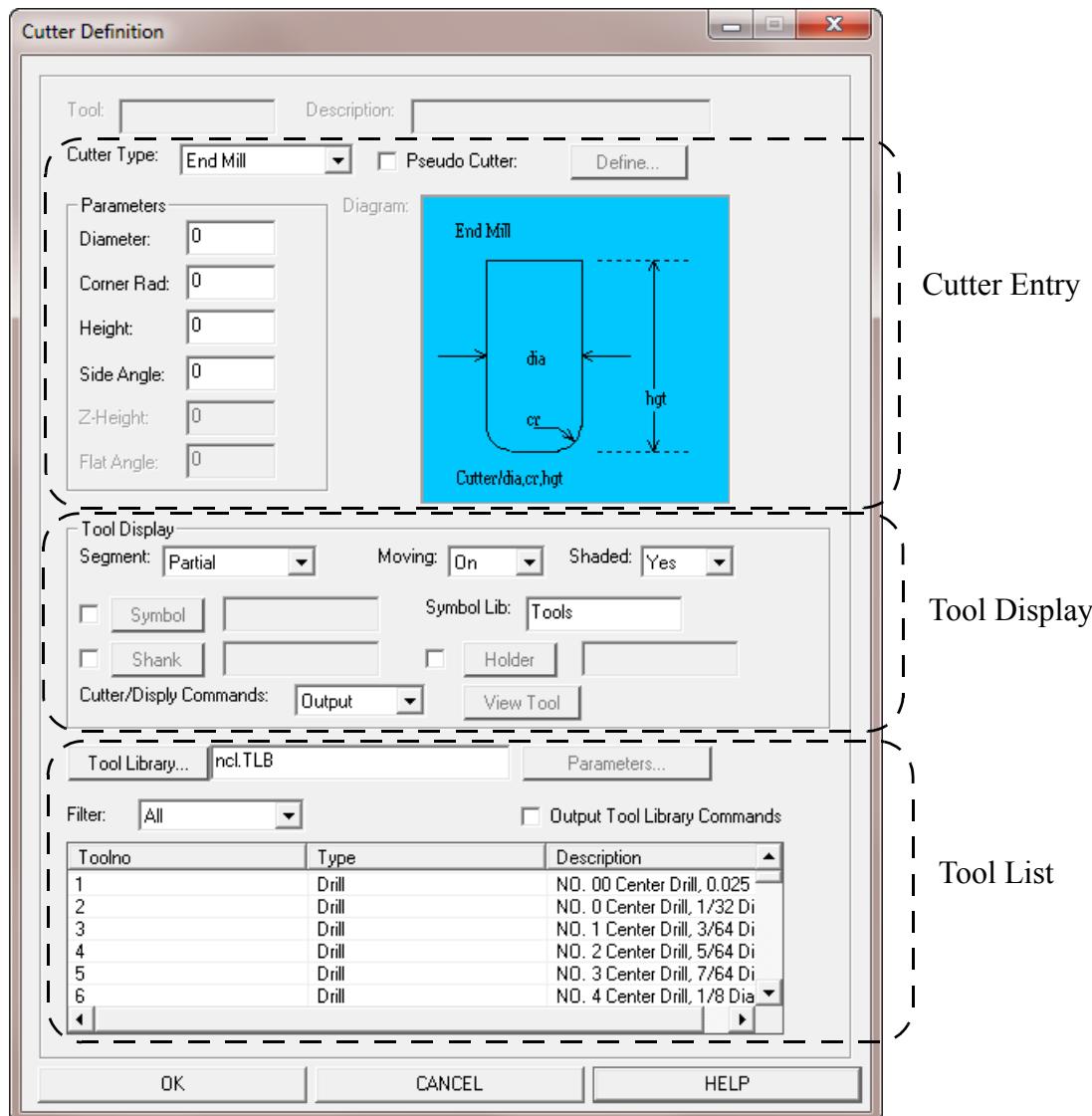
- The input CUTTER/READ statement will be output as a comment statement, i.e.

`$$ CUTTER/READ...`

- The tool library commands normally issued internally with the CUTTER/TOOL command are actually written to the source file and will be saved with the part program.
- A CUTTER/PROFIL command will be issued, when a Cutter Profile library is specified in the tool library, with the CUTTER/TOOL commands to denote which profile library is currently set as the default.

13.3.3 Accessing Tools Using The Interactive Interface

Tool library functions can be accessed from within **NCL** by clicking **TOOLS > CAM > CUTTER**. The following form will appear with a default tool library loaded:



13 TOOL LIBRARY

This form composed of three sections: Cutter Entry, Tool Display, Tool List.

Cutter Entry

The section is for the definition of the standard cutter parameters. Use this section to define the cutter which includes the actual mathematical definition and the display parameters of the cutter. If a pseudo cutter is not defined, the values entered in the “Parameter Columns” will be used for tool path calculation and for display purpose. If a pseudo cutter is defined, the pseudo cutter will be used for tool path calculation and the values defined in the “Parameter Columns” will be used for display purpose.

Tool Display

This section is to define the display parameters of the cutter. The function in this section is virtually the same as described in the *DISPLAY PARAMETER FORM* of the tool library with the following differences.

Symbol Lib

This field specifies the default symbol library to use when loading symbols to use as the cutter or holder display. **NCL** will first look for this library in the local directory and then in the system directory.

Note: Once the tool library is accessed, the symbol library displayed here will not be used if a different symbol library name is specified inside the tool library. Only the one specified inside the tool library will be used.

Cutter/Display Commands

- Toggle between *Output* and *Reference*.
- When *Reference* is specified, then the parameters will be set without the corresponding commands being output to the part program.
- Setting this field to *Output* will cause all of the CUTTER/DISPLAY commands to be output to the part program.

View Tool

This button will only be enabled when a tool from the Tool Library is selected as the active cutter and a drawing has been associated with this

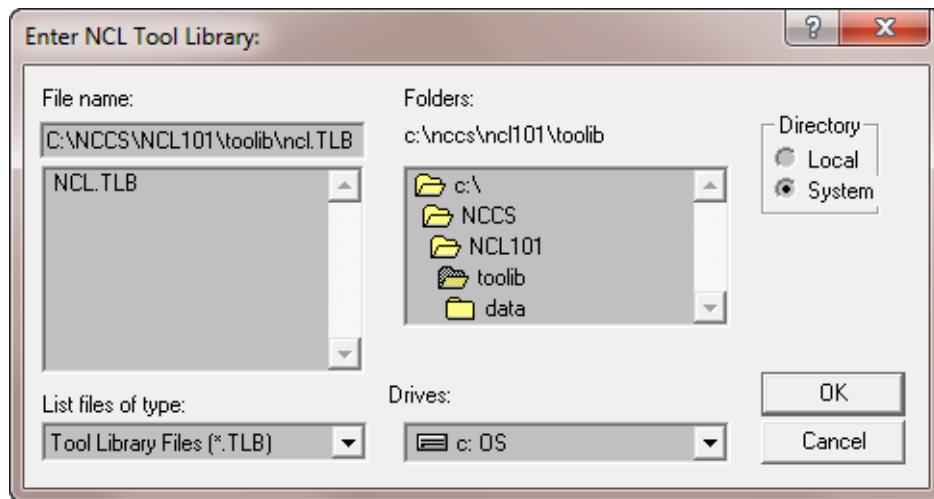
tool. Pressing the View Tool button will display this drawing in a pocket window.

This section can also used to override the *DISPLAY PARAMETER FORM* setting in this area for a selected tool in the library.

Tool List

Tool Library

Click the “*Tool Library*” button to load a different tool library. A file browser as shown below will appear. Before loading a tool the user must first load a library. The user will be given a list of existing libraries in both the local and system directories. Double click the name of the library on wants to load.



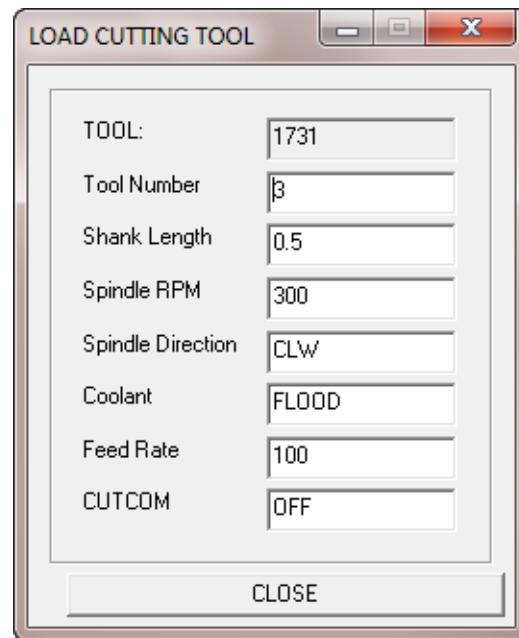
The tool library files can be located in either the local or the system directories. If a tool library that is not located in one of these two directories, an error message will be generated.

Caveat: If there is a tool library (located in a directory other than the local/system directory) with the same name as a tool library located in either the local or the system directory, selecting this tool library and clicking “OK” will not load this copy. The local or the system copy will be loaded instead and there will be no error generated.

13 TOOL LIBRARY

Parameters

The Parameters button will only be enabled when a tool is selected from the Tool List that contains overriding parameters. Click the “*Parameter*” button to open a “Load Cutting Tool” Form. This form will allow the user to accept or override the default values. See example below:

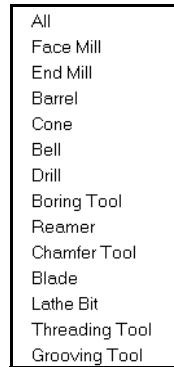


Click *CLOSE* to accept the default values shown or change the desired default values and then click *CLOSE*.

After a tool is highlighted and it is decided not to use any tool from the library, just click on the highlight tool again to deselect it.

Filter

If there is a lot of tools to choose from, the *Filter* button can be used to list just one type of tool. The menu as shown on next page will appear after the *Filter* button was clicked. Just picked the one that need to be appeared on the list.



Highlight the tool from the tool list that need to be loaded, e.g. tool number 9. Upon click the *OK* button, the following command will be written to the part program file and the cutter entries associated with this tool will be activated.

CUTTER/TOOL, NCCS, 9

Output Tool Library Commands

Check this box to enable output of the tool library commands associated with the selected tool. Checking this box will cause a **CUTTER/READ** statement to be issued instead of a **CUTTER/TOOL** statement.

13.4 Working Units Of *NCL/TOOLIB*

NCL/TOOLIB stores the working units in the **NCL** tool library and will convert certain fields in the tool record when changing the working units. All applicable fields in the tool record will be converted. Any fields in the user defined command which needs to be converted must be numeric with the first character containing the “&” symbol (i.e. &1.0).

NCL/TOOLIB supports an initialization file which can be used to specify the working unit. This “toolib.ini” file can be either in the current directory or the following directory:

C:/NCCS/NCL101/toolib\data

The “toolib.ini” file in the current directory will overwrite the one in the following directory:

C:/NCCS/NCL101/toolib\data

13 TOOL LIBRARY

You can either specify “-units:inch” to default to inch or “-units:mm” to default to metric in this file.

14 NCL/IPV COMMANDS

NCL/IPV (NCL In-Process-Verification) is **NCL**'s integrated numerical control simulation and verification module. **NCL/IPV** is a fast, accurate and fun way to simulate the material removal process of NC tool paths. The user can instantly verify tool paths at any time during a part programing session. **NCL/IPV** can also be used to verify existing toolpaths from **NCL** or other CAM systems.

NCL/IPV significantly reduces the need for costly tape prove-outs on expensive CNC equipment and insures accurate, safe, and productive NC programs.

14.1 PPRINT IPV Command

“PPRINT IPV” commands will only be processed during simulation, unlike STOCK and FIXTUR commands that are processed when they are encountered during **NCL** processing.

“id” specifies the ID number of the fixture/stock being created in the following commands.

“id-list” specifies a list of existing fixtures/stocks separated by commas. A “THRU” clause can also be specified in the “id-list” such as “3,THRU,10”.

“Color” specifications can be either the color name or a number corresponding to the color index in all commands. Some “color” settings allow other designators, such as “DEFAULT” or “AUTO”. These will be described for each parameter where additional “color” settings are allowed.

If the statement is longer than 72 characters (the characters “PPRINT IPV” included), the command can be continued onto the next PPRINT statement by specifying a tilde (~) character at the end of the command and specifying the remaining parameters on the following PPRINT statement(s) without the characters “IPV” specified.

If an error is encountered processing a PPRINT IPV command during **NCL/IPV** simulation an error of the same type as post-processor errors will be generated and optionally be displayed in the **NCL/IPV** Clash Detection for as well as output to the log file.

The “PPRINT IPV MODALS” commands that affect the default tool settings will be applied to all tools no matter where the command is encountered in the simulation file. It behaves similar to pressing the Rescan button in the **NCL/IPV**

Edit Tool List form. For this reason it is recommended that these commands be specified only once at the beginning of the program. The following modals are affected by this logic.

PPRINT IPV MODALS COLORS CUT ccol CUTTER ccol SHANK scol ~
PPRINT HOLDER hcol

PPRINT IPV MODALS TOOL TOLER tol MAXANG ang TRANS tval ~
PPRINT EDGES ecol MIN_HEIGHT nhgt MAX_HEIGHT xhgt ~
PPRINT MIN_DIAMETER ndia SHANK smod RAPID rap

14.2 Creating/Loading Stock Or Fixture

Before any verification can be done, a stock or fixture model must be created or loaded into **NCL/IPV**. The creation of stock(s) or fixture(s) can be accomplished by using the **NCL/IPV** interactive interface icon menus or by using the following commands in the part program.

14.2.1 PPRINT IPV FIXTUR BOX id x1,y1,z1,x2,y2,z2 STOCK

This command creates a box shaped stock/fixture through two points in space.

Where:

id = An integer specifies the ID number of the stock/fixture to be created with this command.

x1,y1,z1 = Defines the first corner of the stock/fixture box to be created.

x2,y2,z2 = Defines the opposite corners of the stock/fixture box to be created.

14.2.2 PPRINT IPV FIXTUR CONE id x,y,z,i,j,k,r1,r2,h STOCK

This command creates a cone shaped stock/fixture through a point, along a vector with height and the upper and lower radius specified.

Where:

14 NCL/IPV COMMANDS

id = An integer specifies the ID number of the stock/fixture to be created with this command.

x,y,z = Defines the base center of the cone shape stock or fixture.

i,j,k = Defines the cone axis.

r1, r2 = Numerical values define the radius of the cone stock or fixture to be created at each end.

h = A numerical value defines the height of the cone shaped stock/fixture. A negative value means the axis direction will be opposite in direction to the specified “vector”. A positive value means the axis direction will be in the same direction of the specified “vector”.

14.2.3 PPRINT IPV FIXTUR CYLNDR id x,y,z,i,j,k,r,h STOCK

This command creates a cylindrical shaped stock/fixture with its axis going through one point with radius and height specified

Where:

id = An integer specifies the ID number of the stock/fixture to be created with this command.

x,y,z = Defines the center point of one end of the cylindrical shape stock or fixture.

i,j,k = Defines the cylindrical axis.

r = A numerical value defines the radius of the cylindrical stock or fixture to be created.

h = A numerical value defines the height of the cylindrical shaped stock/fixture. A negative value means the axis direction will be opposite in direction to the specified “vector”. A positive value means the axis direction will be in the same direction of the specified “vector”.

**14.2.4 PPRINT IPV FIXTUR SPHERE id x,y,z,r
STOCK**

This command creates a spherical shaped stock/fixture with the center point and radius of the sphere.

Where:

- id = An integer specifies the ID number of the stock/fixture to be created with this command.
- x,y,z = Defines the center of the spherical shape stock or fixture to be created.
- r = A numerical value defines the radius of the spherical stock or fixture to be created.

**14.2.5 PPRINT IPV FIXTUR TORUS id x,y,z,i,j,k,r1,r2
STOCK**

This command creates a torus shaped stock/fixture using a center point, axis, axial radius, and circular radius.

Where:

- id = An integer specifies the ID number of the stock/fixture to be created.
- x,y,z = Defines the center of the torus shape stock or fixture.
- i,j,k = Defines the torus axis.
- r1 = A numerical value defines the axial radius of the torus stock/fixture to be created.
- r2 = A numerical value defines the circular radius of the torus stock or fixture to be created. This value cannot be larger than “r1”.

14 NCL/IPV COMMANDS

14.2.6 PPRINT IPV FIXTUR LOAD id "file_name.stk" STOCK

This command will load an **NCL/IPV** primitives file "file_name.stk". It does not matter which command is used, either STOCK or FIXTUR, as the primitive file contains the Stock and Fixture designations. **NCL/IPV** will first look in the current directory for the file and if it is not found there, it will look in the system directory which is defined by the "NCL_INCDIR" parameter inside the ncl.init file. The stock(s)/fixture(s) loaded will have the ID number specified by "id". If there are multiple stocks/fixtures in the primitive file, then **NCL/IPV** will continue numbering them in ascending order.

14.2.7 PPRINT IPV FIXTUR STL id INCHES "file_name.stl" STOCK MM

This command will load an STL file with the name "file_name.stl". **NCL/IPV** will first look in the current directory for the file and if it is not found there, it will look in the system directory which is defined by the "NCL_INCDIR" parameter inside the ncl.init file. The stock or fixture loaded will have the ID number specified by "id". If there are multiple Stocks/Fixtures associated with this STL file, then **NCL/IPV** will continue numbering them in ascending order. "INCHES" or "MM" specifies the unit that the STL file is actually stored in.

14.2.8 PPRINT IPV FIXTUR CLONE id idn,ncopies [,~] STOCK

**PPRINT AT, x1,y1,z1,d1, x2,y2,z2,d2, x3,y3,z3,d3]
TRANSL, x,y,z
XYROT, ang
YZROT
ZXROT**

This command clones a specified stock/fixture "ncopies" times and move them at the same time.

Where:

id = An integer specifies the new ID number of the cloned entity.

idn = The ID of the stock/fixture to be cloned.

copies	= Specifies the number of copies to be cloned.
AT, matrix-parameters	= Move the cloned copies by the specified matrix
TRANSL,x,y,z	= Translates the cloned copies linearly by the specified xyz values.
XYROT,angle	= Rotates the cloned copies around the Z-axis by the specified “angle”.
YZDOT,angle	= Rotates the cloned copies around the X-axis by the specified “angle”.
ZXROT,angle	= Rotates the cloned copies around the Y-axis by the specified “angle”.

14.2.9 PPRINT IPV FIXTUR COMPOS id id-list STOCK

This command creates a composite of individual stocks/fixtures.

Where:

id	= An integer specifies the ID number of the composite stock/fixture to be created.
id-list	= A list of predefined stocks.fixtures separated by a “,” that will be combined into a single stock.fixture definition.

Once a stock/fixture is added to a composite definition, it will be removed from the list of active stocks/fixtures and can only be referenced as part of the composite stock/fixture.

14.2.10 PPRINT IPV FIXTUR DECOMP id id-list STOCK

This command decompose composite stocks/fixtures back into their individually referenced stocks/fixtures.

Where:

id	= An integer specifies the starting ID number of the extracted stocks/fixtures. If it is set to “0”, then the ID numbers of the
----	---

14 NCL/IPV COMMANDS

extracted stocks/fixtures before they were stored in a composite solid will be used.

id-list = A list of composite stocks/fixtures to decompose.

14.3 Stock/Fixture Control Commands

**14.3.1 PPRINT IPV FIXTUR MOVE 0 x1,y1,z1,d1,~
 STOCK 1
 INCR**

PPRINT x2,y2,z2,d2,x3,y3,z3,d3,id-list

This command applies the matrix defined by its canonical form (x1,y1,z1,d1,x2,y2,z2,d2,x3,y3,z3,d3) to the stock(s)/fixture(s) with the id-list denoted by "id1 [[...] , idn, THRU, idm] [...]" in the command.

“0” specifies the matrix will be applied to the original Stock/Fixture and will not be multiplied by any matrix already associated with this Stock/Fixture.

“1” or “INCR” specifies the matrix will be applied to the stocks/fixtures incrementally from the current position.

**14.3.2 PPRINT IPV FIXTUR TRANSL 0 x,y,z,id-list
 STOCK 1
 INCR**

This command translates the specified stocks/fixtures linearly.

“x,y,z” specified the translation values along the XYZ axes. “id-list” denotes the list of the stocks/fixtures to translate.

“0” specifies the translation will be applied to the original Stock/Fixture.

“1” or “INCR” specifies the translation will be applied to the stocks/fixtures incrementally from the current position.

**14.3.3 PPRINT IPV FIXTUR XYROT 0 angle,id-list
STOCK YZROT 1
ZXROT INCR**

This command rotates the stock(s)/fixture(s) about a major axis.

“XYROT” rotates about the Z-axis. “YZROT” rotates about the X-axis. “ZXROT” rotates about the Y-axis.

“0” specifies the rotation will be applied to the original Stock/Fixture.

“1” or “INCR” specifies the rotation will be applied to the stocks/fixtures incrementally from the current position.

“angle” specifies the rotation angle.

“id-list” denoted by "id1 [[[...] [, idn, THRU, idm]] [...]]" specifies the stocks/fixtures to be rotated.

**14.3.4 PPRINT IPV FIXTUR REMOVE id-list
STOCK**

This command removes the specified stock(s)/fixture(s) denoted by the “id-list” such as "id1 [[[...] [, idn, THRU, idm]] [...]]" in the command.

**14.3.5 PPRINT IPV STOCK SAVE id [“file”]
FIXTUR**

Saves stocks/fixtures as external STL files. “id” specifies the ID number of the stock or fixture to save. If “id” is set to 0 then all stocks/fixtures will be saved as STL files. “file” is optional and specifies the base filename of the saved STL file. If “file” is not specified, then the name of the motion playback file will be used. When multiple files are created a number will be appended to this base filename for each subsequent file.

Caveat: User will be asked for the stl file name if the current cl file is used to generate the sim file in an interactive session with no file name specified.

14.3.6 PPRINT IPV FIXTUR MODIFY 0 color,visible,lucency,active,~ STOCK

PPRINT toler,id-list

This command modifies the attributes of the specified stock(s)/fixture(s) with the "id-list" denoted by "id1 [[[...] [, idn, THRU, idm]] [...]]" in the command.

Where:

- 0 = A required parameter.
- color = Specifies the new color of the stock/fixture.
- visible = Specifies the visibility of the stock/fixture. It can be either "0" or "1".
- TOLER = Specifies the geometry tolerance.
- TRANS = Defines the transparency of the stock/fixture and can be in the range of 1 to 100.
- ACTIVE = Specifies whether the stock/fixture will be used in the simulation. It can be either "0" or "1".

Note: Specifying a value of -1 for any of these attributes will result in the solid maintaining its original attribute.

14.3.7 PPRINT IPV STOCK REMOVE_CHIPS 0 x1,y1,z1,i1,j1,k1,...,~ PPRINT xn,yn,zn,in,jn,kn

This command removes the excess chips from the stock directly within the part program. A list of point-vectors (x,y,z,i,j,k) can be input with this command. These point-vectors will be used to select the portion of the stock to keep. There should be one point-vector per defined stock and each of these should lie within the stock and point towards an edge of the stock to ensure that the correct portion of the stock is retained.

14.4 CUTTER/SHANK/HOLDER Commands**14.4.1 PPRINT IPV CUTTER parameters**

Defines the active cutter. “parameters” are real values used to define the cutter shape. All cutter definitions supported by **NCL** are supported by the PPRINT IPV CUTTER command.

14.4.2 PPRINT IPV CUTTER BLADE parameters

Defines a blade cutter. The “parameters” follow the same syntax as the CUTTER/BLADE command.

14.4.3 PPRINT IPV CUTTER LATHE parameters

Defines a lathe cutter. The “parameters” follow the same syntax as the CUTTER/LATHE command.

14.4.4 PPRINT IPV CUTTER DISPLAY “pt-list”

Specifies a cutter profile (pt-list) that is stored in the tool profile description file.

14.4.5 PPRINT IPV SHANK parameters

Defines a cutter shank using the standard parameters as supported by the CUTTER/DISPLAY,SHANK command.

14.4.6 PPRINT IPV SHANK DISPLAY “pt-list” parameters

Defines a cutter shank using a profile (pt-list) that is stored in the tool profile description file.

14.4.7 PPRINT IPV HOLDER parameters

Defines a cutter holder using the standard parameters as supported by the CUTTER/DISPLAY,HOLDER command.

14.4.8 PPRINT IPV HOLDER DISPLAY “pt-list” parameters

Defines a cutter holder using a profile (pt-list) that is stored in the tool profile description file.

14.5 NCL/IPV Model Commands

14.5.1 PPRINT IPV MODALS AUTO_HIDE [mode] [TRANS tval] ~ PPRINT [EDGES emode]

This command specifies whether **NCL/IPV** will dynamically set the translucency and edge display of any solid that obscures an important solid. “mode” can be set to “YES” or “NO” and enables or disables this feature. “TRANS” defines the translucency of the obscuring solid. “EDGES” can be set to “YES” or “NO” and determines if the edges of an obscuring solid should be displayed.

14.5.2 PPRINT IPV MODALS COLORS [CUT ccol] [CUTTER ccol] ~ PPRINT [SHANK scol] [HOLDER hcol] [FIXTUR_CUT fcol] ~ PPRINT [HOLDER_CUT hccol] [RAPID_CUT rcol] ~ PPRINT [AUTO_COLOR acol] [USE_STOCK smod] ~ PPRINT [USE_FIXTUR fmod]

“CUT” defines the default cut color. “ccol” can be set to “AUTO”, which automatically changes the cut color for each tool, or to a valid color designator. “CUTTER” defines the default display color of all tools. “ccol” can be set to “DEFAULT”, which uses the cut color as the cutter color, or to a valid color designator. “SHANK” defines the default shank color. “scol” can be set to “DEFAULT”, which uses the cutter color, or to a valid color designator. “HOLDER” defines the default holder color. “hcol” can be set to “DEFAULT”, which uses the cutter color, or to a valid color designator.

“FIXTUR_CUT” defines the color to use when a fixture is cut. “fcol” can be any valid color designator. “HOLDER_CUT” defines the color to use when the tool holder cuts a solid. “hccol” can be any valid color designator. “RAPID_CUT” defines the color to use when a solid is cut while in RAPID mode. “rcol” can be any valid color designator.

“AUTO_COLOR” defines the initial color when the cut color is automatically changed when a new tool is loaded. “acol” can be any valid color designator. “USE_STOCK” determines if colors assigned to stocks should be include in the list of automatic cut colors to use. “smod” can be set to “YES” or “NO”. “USE_FIXTUR” is the same as “USE_STOCK” except that it applies to fixtures.

14.5.3 PPRINT IPV MODALS MACHINE type

Defines the machine type to use for simulation. “type” can be “MILL”, “LATHE”, “MILLTURN” or “STRINGER”.

14.5.4 PPRINT IPV MODALS STACK [mode] [FIXTUR fstate] [size]

Enables/Disables the ability to interactively Undo/Redo cuts in **NCL/IPV**. “mode” can be set to “ON” or “OFF” and enables/disables the **NCL/IPV** Undo/Redo stack. “FIXTUR” determines if fixture cuts are stored in the Undo/Redo stack. “fstate” can either be “YES” or “NO”. “size” defines the maximum number of cuts that can be stored in the Undo/Redo stack. A value of 0 creates an unlimited stack size.

14.5.5 PPRINT IPV MODALS STOCK [COLOR col] ~ FIXTUR

PPRINT [VISIBLE vmod] [TRANS tval] [TOLER tol] ~
PPRINT [IMPORTANT imod] [EDGES ecol] [STL smod] ~
PPRINT [STL_STOP spmod] [STL_DEACT sdmod] ~
PPRINT [STL_SKIP_ERROR skmod]

Defines the default settings for stocks and fixtures. “STOCK” defines the settings for stocks. “FIXTUR” defines the default settings for fixtures.

“COLOR” defines the default color. “col” is any valid color designator. “VISIBLE” defines the initial visibility of stocks. “vmod” can be set to “ON” or “OFF”. “TRANS” defines the default translucency. “tval” can be in the range of 1-100. “IMPORTANT” determines if the solid is important. Important solids can cause solids that obscure an important solid to become translucent. “imod” can be set to “YES” or “NO”. “EDGES” defines the default edge display setting for solids. “ecol” can be “OFF” to disable edges display, “DEFAULT” to use the solid color for the edge color, or any valid color designator.

14 NCL/IPV COMMANDS

“STL” determines how STL models will be output. “smod” can be “ASCII” or “BINARY”. “STL_STOP” determines if **NCL/IPV** will output an error message when a bad STL file is loaded. “spmod” can be set to “YES” or “NO”. “STL_DEACT” determines whether an STL model will be automatically deactivated if it is bad. “sdmod” can be set to “YES” or “NO”. “STL_SKIP_ERROR” will cause **NCL/IPV** to skip the error checking when loading an STL file when “skmod” is set to “YES”. Setting “skmod” to “NO” will cause **NCL/IPV** to perform standard error checking on STL models.

14.5.6 PPRINT IPV MODALS TOOL [TOLER tol] [MAXANG ang] ~ PPRINT [TRANS tval] [EDGES ecol] [MIN_HEIGHT nhgt] ~ PPRINT [MAX_HEIGHT xhgt] [MIN_DIAMETER ndia] ~ PPRINT [FROM_NEXT fmod] [SHANK smod] [RAPID rap]

Defines the default tool modals. “TOLER tol” defines the default cutting tolerance for defined tools. “MAXANG ang” defines the maximum angular change that the tool axis can make in a single move.

“TRANS” defines the default translucency of the defined tools. “tval” can be in the range of 1-100. “EDGES” defines the default edge display setting for the defined tools. “ecol” can be “OFF” to disable edges display, “DEFAULT” to use the tool color for the edge color, or any valid color designator.

“MAX_HEIGHT” defines the maximum height for defined tools, “MIN_HEIGHT” defines the minimum height for defined tools, and “MIN_DIAMETER” defines the minimum tool diameter. “FROM_NEXT” determines if the move immediately following a tool change should be considered a positioning or cutting move.

“fmod” can be set to “YES” for positioning moves or “NO” for cutting moves. “SHANK” determines if the tool shank should be treated as part of the cutter or as a tool holder. “smod” can be set to “CUTTER” or “HOLDER”. “RAPID” defines the default RAPID rate for defined tools.

14.6 Miscellaneous PPRINT IPV Commands

14.6.1 PPRINT IPV DNTCUT

This command causes **NCL/IPV** to treat the next motion statement as a *FROM* (positioning) move.

This command is useful when positioning to an opposite side of a rotary table in one example, since the programmed move will result in a straight linear move to the next position, usually right through the part.

This command is ignored if a **PostWorks** simulation file is being used for motion playback, since the simulation file will show the actually moves created by rotating the table instead of moving the linear axes.

This *DNTCUT* command is only valid for the next move and will be automatically cancelled after the move.

14.6.2 PPRINT IPV OFFSET “file.ofs”

This command is used to load an external Work Coordinate Offset Systems (WCS) file during machine simulation.

“file.ofs” is the name of the external WCS file.

A WCS system is activated by the use of the CUSTOM/ADJUST and TOOLNO/ADJUST commands. The following command syntaxes are supported by **NCL/IPV**.

CUTCOM / ADJUST,ON	
TOOLNO	OFF
	n [, PLUS]
	MINUS

“ADJUST” is required for proper syntax. “ON” will enable WCS offsets for Register 0. “OFF” disables WCS offsets of this type (CUSTOM, TOOLNO). “n” specifies the register to enable offsets for. “PLUS” enables offsets in the positive direction (default) and “MINUS” enables offsets in the negative direction.

The CUTCOM and TOOLNO WCS offsets are stored in separate registers and can be active at the same time.

In order for the WCS offsets to have any effect on the machine axes positions, a file that contains the type of offset, register number, and axis offsets for this WCS register must be defined. This external file has the following format.

Fixture reg label1 ofs1 [label2 ofs2 ... labeln ofsn]
Tool

14 NCL/IPV COMMANDS

The file contains multiple lines using this format that define the axes offset values for the WCS registers. FIXTURE defines a CUTCOM style offset register and TOOL defines a TOOLNO style offset register. “reg” defines the register value that the offsets are being defined for. For example, if an H15 code enables WCS offsets 15, then specifies 15 for “reg”. “label” specifies the label of the axis which will be offset and “ofs” is the offset value to apply to this axis when the “reg” WCS offsets have been enabled. “label” must match the label of one of the axes in the machine definition file. Each register can be specified for multiple axes.

The external WCS offset file must first be loaded before any WCS offsets will be activated.

14.6.3 PPRINT IPV OFFSET label ofs [label2 ofs2 [...] labeln ofsn]

This command is used to directly define axis offset values without referencing an external WCS offset file.

The offset values are treated exactly the same as if they came from the WCS offset file.

14.6.4 PPRINT IPV POSITN label pos [label2 pos2 [...] labeln posn]

This command is used to position individual axes on the machine without having to program a motion block.

“label” is the label of the axis to position as defined in the **NCL/IPV** machine model file. “pos” is the new position for this axis. Multiple axes can be specified in a single command and all of the axes will move at the same time.

14.6.5 PPRINT IPV PRINT_SCREEN type size [“file”]

Prints the current **NCL/IPV** view to an image file. “type” specifies the image type of the file to create and can be “BMP”, “GIF”, “JPG”, or “PS” (PostScript). “size” specifies the size of the paper to format the print image for and can be “AH”, “AV”, “B”, “C”, “D”, “E”, “F”, “A0”, “A1”, “A2”, “A3”, “A4”, “USER1”, “USER2”, “USER3”, or “USER4”.

“file” is optional and specifies the name of the file to create. If “file” is not specified, then the name of the motion playback file followed by an underscore and an number (_n) denoting the number of print files created during the current run, and with an extension matching the file type (BMP,GIF,JPG,PS) will be used.

14.6.6 PPRINT IPV SESSION EXPORT [“file”]

Exports the active **NCL/IPV** session to an external file. “file” is the name of the external file to save the session to and is typically named with a file type of “.ipv”. If “file” is not specified, then the name of the motion playback file will be used with the file extension of “.ipv”.

14.6.7 PPRINT IPV SESSION IMPORT “file”

Imports an external **NCL/IPV** session. file is the name of the external session file to import and is typically named with a file type of “.ipv”.

14.6.8 PPRINT IPV SPINDLE n [, n1, n2, ...]

This command defines which spindle to load the next tool into in “Machine Simulation” mode. “n” is the number corresponding to the “ToolSpindle n” command within a machine component definition file. Up to 10 spindles can be defined concurrently in the range of 0 to 9. This command does not have any effect if “Machine Simulation” mode is not active.

14.6.9 PRINT IPV STOCK RESET_CUTCOLOR [id-list]

Resets the color of cut faces on the list of solid to the current color of the solid. Usually when changing the color of a stock the cut faces do not change color to match the stock color. If “id-list” is not specified, then all stocks/fixtures are modified.

**14.6.10 PPRINT IPV TOOL [CUT_COLOR ccol] ~
PPRINT [CUTTER_COLOR ctcol] [CUTTER_EDGES cecol] ~
PPRINT [CUTTER_TRANS ctval] HOLDER_COLOR hcol] ~
PPRINT [[HOLDER_EDGES hecol] [HOLDER_TRANS htval] ~
PPRINT [MAXANG ang] [RAPID rap] [SHANK smod] ~
PPRINT [SHANK_COLOR scol] [SHANK_EDGES secol] ~
PPRINT [SHANK_TRANS stval] [TOLER tol]**

The PPRINT IPV TOOL command specifies the attributes for the active tool, therefore this command must be specified after the tool is defined.

14 NCL/IPV COMMANDS

“CUT_COLOR” defines the cut color. “ccol” can be set to “AUTO”, which automatically changes the cut color for each tool, or to a valid color designator. “CUTTER_COLOR” defines the display color of the cutter. “ctcol” can be set to “DEFAULT”, which uses the cut color as the cutter color, or to a valid color designator. “CUTTER_EDGES” defines the edge display setting for the cutter. “cecol” can be “OFF” to disable edges display, “DEFAULT” to use the cutter color for the edge color, or any valid color designator. “CUTTER_TRANS” defines the default translucency of the defined cutter. “ctval” can be in the range of 1-100.

“HOLDER_COLOR” defines the display color of the tool holder. “hcol” can be set to “DEFAULT” or a valid color designator. “HOLDER_EDGES” defines the edge display setting for the holder. “hecol” can be “OFF”, “DEFAULT,” or any valid color designator. “HOLDER_TRANS” defines the default translucency of the defined holder. “htval” can be in the range of 1-100.

“MAXANG ang” defines the maximum angular change that the tool axis can make in a single move. “RAPID” defines the default RAPID rate for the active tool.

“SHANK” determines if the tool shank should be treated as part of the cutter or as a tool holder. “smod” can be set to “CUTTER” or “HOLDER”. “SHANK_COLOR” defines the display color of the tool shank. “scol” can be set to “DEFAULT” or a valid color designator. “SHANK_EDGES” defines the edge display setting for the shank. “secol” can be “OFF”, “DEFAULT”, or any valid color designator. “SHANK_TRANS” defines the default translucency of the defined holder. “stval” can be in the range of 1-100.

“TOLER tol” defines the default cutting tolerance for the active tool.

14.6.11 PPRINT IPV TOOLPN [label] [x,y,z,i,j,k,u,v,w]

This command is used to define the specific tooling pin so that individual stocks and fixtures can be assigned to it on the machine.

Where:

label = specifies the machine tooling pin name that all stocks defined after this command will be attached to. The “label” can be up to 18 characters.

x, y, z = Origin of the tooling pin in relationship to the part.

i, j, k = Axis of the tooling pin cylinder in relationship to the part.

u, v, w = Vector that points from the center of the tooling pin towards its flat side.

The part tooling pin parameters specified in this command (x,yz, i,j,k, u,v,w) will be stored with the machine tooling pin and any stocks/fixtures attached to this machine tooling pin will use these values as the part tooling pin.

If “label” is not specified, then the last designated tooling pin will be used. “label” must be specified if the tooling pin parameters are not specified. In this case the default tooling pin for subsequent stocks/fixtures will be set without modifying its part tooling pin parameters.

14.6.12 PPRINT IPV VIEW FIT

Does an extreme zoom on the **NCL/IPV** window.

APPENDIX A: ERROR MESSAGES

This appendix contains more detailed information about the Error Messages that may occur while running **NCL**. Each Error Message is listed by the number you will see on the screen when it appears. Next to the number is the Error Message as it will appear on the screen. Below the message is an explanation of what it means.

To obtain an explanation of the Error Message in **NCL** use the *HELP command followed by: ERRxxx. "xxx" is the error message number shown on the screen with the Error Message at the time the error occurs. For example to display the full explanation for error number 16 enter:

```
*HELP/ ERR016
```

and the text will be displayed in a scrolling window in the graphic area of the screen. **The *HELP command does not work with any error number bigger than 416.**

ERR001: GEOMETRY TYPE EXPECTED

An **NCL** geometry type word is required at this point in the statement. Refer to the geometry creation and lathe module sections for valid geometry types.

ERR002: POINT, NUMBER OR VECTOR EXPECTED

A point variable name, a nested point definition, a number, a vector variable name or a nested vector is expected at this point in the statement.

ERR003: REAL OR INTEGER EXPECTED

A real number is a number that contains a decimal point. An integer is a number that does not contain any decimal point.

ERR004: END OF STATEMENT EXPECTED

There are one or more characters found after the last expected character in the statement.

ERR005: STATEMENT NOT YET IMPLEMENTED

This error message indicates that a feature has been tried that is not yet fully implemented in **NCL** yet. The feature is planned for future releases of **NCL**.

ERR006: ‘=’ EXPECTED

NCL expected an equal sign at this point in the statement.

ERR007: NUMBER OR SCALAR EXPECTED

A numeric value or a variable name that has been previously assigned a numeric value is expected at this point in the statement.

ERR008: IDENTIFIER PREVIOUSLY DEFINED

The indicated identifier name has already been used in another statement. An identifier name may not be re-used if it has been assigned to a geometry item. An identifier may be re-used if it has been assigned to a scalar (numeric) value or if it has been assigned to a geometry item and CANON has been set on either by the CANON/ ON statement or a *SET/ CANON control command.

ERR009: IDENTIFIER NOT PREVIOUSLY DEFINED

The indicated identifier has not been assigned to either a geometry item or a scalar (numeric) value.

ERR10: ‘TRANS’ EXPECTED

The vocabulary word TRANS was expected in the statement.

ERR11: VECTOR EXPECTED

An identifier that had been previously assigned to a vector geometry item or a nested vector definition is expected at this point in the statement.

ERR12: INVALID GEOMETRY FORMAT

The statement in error does not conform to one of the acceptable formats for defining a geometric item. Check on proper available formats by looking in the section of the help file that deals with the type of geometry that you are trying to define. For example, if you are trying to define a point you may use a *HELP/ POINT command to scan all the permissible ways to define a point.

ERR13: STATEMENT MAY NOT BEGIN WITH THIS

The statement has a character or word that is not a permissible item to start an **NCL** statement with. Generally an **NCL** statement may only start with words that

are valid identifiers for geometry or a major word such as GOFWD, TLRGT, DRAFT, VECTOR, PARTNO, etc.

ERR14: MAXIMUM NUMBER OF THIS TYPE GENERATED

The maximum number of automatically named items for this type of geometry has been exceeded. The maximum number is 9999. This geometry may still be defined but the programmer must assign a specific name to it other than the type assigned by the automatic name generation facility.

For example, if a program has point definitions with names assigned to them of PT1 through PT9999 the next time a point definition tries to use the automatic name generation facility it will receive this error, but the programmer can assign a name to the point definition such as PT1A and the point definition will be processed correctly.

ERR15: GEOMETRIC ELEMENT LIMIT EXCEEDED

In defining a geometric entity, too many elements were used in the definition statement.

ERR16: VECTOR MUST FOLLOW POINT

In a curve definition statement, the defining geometry items may only be point identifiers with optional vector identifiers associated with any point identifier to indicate slope direction at that point. This error occurs if this syntax rule is not followed.

ERR17: LINE EXPECTED

NCL expects to find a line identifier name or a nested line definition at this point.

ERR18: DIRECTION MODIFIER EXPECTED

NCL expects to find a direction modifier word at this point. Valid direction modifier words are:

XLARGE	or	XL
XSMALL	or	XS
YLARGE	or	YL
YSMALL	or	YS
ZLARGE	or	ZL
ZSMALL	or	ZS

Direction modifiers clarify which of two or more possible choices is wanted in situations where there is a choice.

ERR19: PLANE EXPECTED

NCL expects to find a plane identifier name or a nested plane definition at this point.

ERR20: POINT EXPECTED

NCL expects to find a point identifier name or a nested point definition at this point.

ERR21: CURVE EXPECTED

NCL expects to find a curve identifier name or a nested curve definition at this point.

ERR22: '/' EXPECTED

NCL expects a slash character at this point in the statement. Generally, a slash character is required after the first major word of the statement such as in a geometry definition statement like POINT/ or a motion statement like [GORGT/](#).

ERR23: DUPLICATE NAME GENERATION**ERR24: POINT, VECTOR OR 'RADIUS' EXPECTED**

NCL expects either a point identifier name, nested point definition, vector identifier name, nested vector definition, scalar identifier name or numeric value at this point in the statement.

ERR25: INVALID CONTROL OPTION SPECIFICATION

A word or item was found in a control command that is not valid for that type of control command. Check the particular command syntax rules in the control command section for valid options.

ERR26: INVALID CONTROL COMMAND

A word or command was found after an "*" character at the beginning of a statement that is not a valid **NCL** control command. Check the control command section for a list of valid control commands.

ERR27: MOTION GENERATION VERB EXPECTED

NCL found a word or item in the statement at this point other than a valid motion generation verb. Valid motion generation verbs are:

FROM, GOTO, GODLTA, GOFWD, GOBACK, GORGT, GOLFT,
GOUP, GODOWN, RMILL, POCKET, FMILL, GOFWDA or
PROFIL.

ERR28: INVALID DRIVE SURFACE

The type of geometry that has been given as the drive surface, that is the surface that controls the contact point of the side of the cutter, is not a valid type. Valid geometry types for a drive surface are:

LINE, PLANE, CIRCLE, CURVE and SURF

If a cone or bell shaped cutter is being used, and TLLFT or TLRGT is specified, the only valid drive surface types are PLANE and SURF.

ERR29: CHECK SURFACE MODIFIER EXPECTED

NCL expected to see a word at this point in the statement indicating how the cutter is to stop in relation to the check surface. Valid check surface modifiers are:

TO, ON and PAST

ERR30: INVALID CHECK SURFACE

The type of geometry that has been given as the check surface, that is the surface that controls the stopping point of cutter, is not a valid type. Valid geometry types for a check surface are:

POINT, LINE, PLANE, CIRCLE, CURVE and SURF

If a cone or bell shaped cutter is being used you cannot check TO, PAST or TANTO wireframe geometry (only PLANEs and SURFs) unless CONTCT/ON has been specified.

ERR31: LINE MUST HAVE REAL LENGTH

The statement is trying to define a line that has a length of zero. This can occur if you try to define a line with two points that have the same coordinates.

ERR32: PLANE NORMAL MUST HAVE REAL LENGTH

A plane's canonical form is made up of a vector that is normal to the plane and an offset value along that vector indicating the distance the plane is from the origin (0, 0, 0). An attempt was made to define a plane with either zero or very small values for the X, Y and Z coordinates of the normal vector.

ERR33: POINT, NUMBER OR SCALAR EXPECTED

A point variable name, nested point definition, number or a variable name that has been previously assigned a numeric value is expected at this point.

ERR34: INVALID NUMBER OF ARGUMENTS

Either not enough or too many arguments (words or characters) have been given for **NCL** to process this statement.

ERR35: NUMBER OR VOCABULARY WORD EXPECTED

NCL expected a numeric value or a valid **NCL** vocabulary word at this point. Refer to the particular section that covers this statement type for more details.

ERR36: INVALID PART SURFACE

The type of geometry that has been given as the part surface, that is the surface that controls the contact point of the bottom of the cutter, is not a valid type. Valid geometry types for a part surface are:

PLANE and SURF

ERR37: INVALID SEQUENCE RANGE SPECIFIED

A "SHOW/ SOURCE" statement has an invalid item specified after the word "SOURCE". Valid items are: "S", "*", a line number, "FN=part program file name" or a blank.

ERR38: TLAXIS MAY ONLY BE NORMAL TO PART SURF

The tool axis must be normal (perpendicular) to the part surface during the processing of this statement.

ERR39: GENERATED POINT HAS NOT BEEN DECLARED

A point identification name that was generated as a result of a THRU clause has not been defined.

ERR40: GENERATED CURVE HAS NOT BEEN DECLARED

A curve identification name that was generated as a result of a THRU clause has not been defined.

ERR41: THRU NEEDS A GENERATED OR RESERVED ID

The **NCL** vocabulary word THRU has been used after a word or variable name that is not an automatically generated or subscripted name.

A valid subscripted variable name is one that has been used to define a subscripted array such as SSP(1) or HL(20).

ERR42: A GENERATED IDENTIFIER MUST FOLLOW THRU

The **NCL** vocabulary word THRU has been used with a word or variable name following it that is not a generated or subscripted type name. See error 41 above for valid names information.

ERR43: MUST USE 'FIT' FOR MORE THAN 50 POINTS

A curve definition with more than 50 points must use the curve FIT definition syntax. See the section on curves definition for more information.

ERR44: LINE APPEARS VERTICAL

In this case, a line must have a real length when projected onto the XY plane.

ERR45: MOVE COMPUTES TO BE ZERO (?)

The motion statement generates a move to the same location the tool is at the beginning of the motion statement.

ERR46: SHOULD BE LINE, NOT CURVE

The curve definition defines a straight line curve.

ERR47: NOT ENOUGH POINTS

An insufficient number of points were input for this construct.

ERR48: POINTS TOO CLOSE TOGETHER

The points that are given to define a circle are too close together to be able to create a circle definition.

ERR49: INPUT VECTOR NO GOOD

The vector is not suitable for the current situation.

ERR50: SLOPES DID NOT CONVERGE IN 15 TRIES

The input points to a curve definition are too radical to define a reasonable curve.

ERR51: CONDITIONS ARE TOO SEVERE

The geometric elements that are being used to define the surface create a condition that would result in too sharp or too severe of an angle in the surface to be defined. Look carefully at the defining geometry.

ERR52: MATRIX IS NOT 1/1 ROTATION BOX

A matrix has been defined that does not have mutually perpendicular X, Y and Z axes. The matrix definition has been processed and the matrix defined. This is only a warning message.

ERR53: INTEGER EXPECTED

NCL was expecting to find an integer value or a scalar identifier with an integer value. An integer may not contain any decimal point.

ERR54: MACRO NAME EXPECTED

NCL was expecting to find a variable name that had been assigned to a macro statement.

ERR55: NO MACRO STATEMENT IS IN EFFECT

NCL has found a TERMAC statement but it was not processing a MACRO.

ERR56: 'ON' OR 'OFF' EXPECTED

The word ON or OFF was expected at this point in the statement.

ERR57: ';' EXPECTED

NCL was expecting to find a comma at this point in the statement. Check the syntax for this statement type.

ERR58: ARITHMETIC OPERATOR EXPECTED

An arithmetic operator was expected at this point. Valid arithmetic operators are:
+, -, *, / and **.

ERR59: EXPRESSION TOO LONG, PLEASE SEGMENT IT

The expression contains more than 20 arithmetic operators in it. Break it into two or more smaller expressions.

ERR60: INCORRECT PARENTHESIS COUNT

NCL found there were more or less right parentheses than there were left parentheses in the statement. The number of right parentheses must be equal to the number of left parentheses.

ERR61: INVALID SYNTAX FORMAT

This statement does not follow the syntax rules for this type of statement. Refer to the portion of the manual or help file that covers this particular statement for proper syntax rules.

ERR62: MAXIMUM TOKEN SIZE IS 20 CHARACTERS**ERR63: DIVIDE BY ZERO NOT ALLOWED**

A condition has occurred that would cause **NCL** to divide by zero.

ERR64: CANNOT TAKE SQRT OF NEGATIVE NUMBER

A square root can only be taken of a positive number. Check the value of the variable if a variable is being used.

ERR65: PT, VE, CV, SF OR INTEGER EXPECTED

NCL requires that one of the geometry types given in the error message be used at this point in the statement. Check the syntax rules for this particular statement type for more details.

ERR66: PARAMETER PREVIOUSLY USED IN THIS CALL

The parameter indicated has already appeared in this CALL statement.

ERR67: PARAMETER ALREADY USED BY CALLING MACRO

A parameter name was used in this CALL that is already being used by a MACRO that called this MACRO or is currently in the nest of MACROs being executed.

ERR68: COMMA OR DEFAULT VALUE EXPECTED

In a MACRO parameter list either a comma, valid parameter default value of a scalar, geometry name or an **NCL** vocabulary word was expected.

ERR69: PARAMETER NAME EXPECTED

A valid parameter name associated with the called MACRO is expected at this point.

ERR70: PARAMETER NOT IN MACRO STATEMENT

The specified parameter name was not found in the MACRO statement that is being called.

ERR71: PARAMETER VALUE EXPECTED

A valid parameter value of a scalar, geometry name or an **NCL** vocabulary word was expected.

ERR72: MACRO PARAMETER(S) NOT GIVEN VALUE(S)

One or more MACRO parameters were not assigned a value either by the macro statement or the call statement.

ERR73: MACRO NAME MAY NOT BE SUBSCRIPTED

MACROS may not be assigned a subscripted variable name.

ERR74: MACRO STMT MAY NOT BE NESTED IN MACRO

A MACRO statement was found inside a MACRO/ TERMAC range. A MACRO may be called from inside another MACRO but not defined there.

ERR75: IDENTIFIER LENGTH MAY NOT EXCEED 6 CHARS

The input token defines an identifier name that is longer than 6 characters.

ERR76: THIS LABEL HAS BEEN PREVIOUSLY DECLARED

A label may only be used once within a loop or a macro.

ERR77: NESTED LOOP NOT ALLOWED

A LOOPST statement has been encountered within a current looping region or within a MACRO.

ERR78: NO LOOPST STATEMENT IS IN EFFECT

A LOOPND statement was encountered without first seeing a LOOPST.

ERR79: LABEL ONLY ALLOWED IN LOOP OR MACRO

A label was encountered outside of a LOOP or MACRO.

ERR80: STATEMENT ONLY ALLOWED IN MACRO OR LOOP

A JUMPTO or an IF statement was encountered outside of a LOOP or MACRO.

ERR81: LABEL NOT DECLARED IN CURRENT LOOP

A label was referenced that has not been declared within the current LOOP or MACRO.

ERR82: LABEL EXPECTED

A valid label is expected at this point in the statement.

ERR83: ARITHMETIC EXPRESSION EXPECTED

An arithmetic expression with a value of plus, minus or zero is required in an IF statement.

ERR84: STATEMENT NOT ALLOWED IN LOOP OR MACRO

This statement is not allowed within a LOOP or MACRO.

ERR85: SUBSCRIPT OUT OF RANGE

The subscript for the identifier is not within the range specified by the RESERV statement for that identifier.

ERR86: SUBSCRIPTED IDENTIFIER NOT ALLOWED

A subscripted identifier is not allowed at this point of the program.

ERR87: IDENTIFIER EXPECTED

A valid identifier is expected at this point.

ERR88: IDENTIFIER HAS NOT BEEN ‘RESERV’ED

A RESERV statement must be executed for an identifier before a subscript may be used with that identifier.

ERR89: IDENTIFIER CURRENTLY HAS DIFFERENT TYPE

An identifier type cannot be changed. For example, if B3 is defined to be a POINT, it cannot be redefined to be a line.

ERR90: CAN’T CALL CURRENTLY EXECUTING MACRO

An attempt was made to CALL a MACRO which is currently executing.

ERR91: TOO MANY POINTS IN CURVE DEFINITION

A CURVE FIT definition was processed that contained too many input points.

ERR92: SUBSCRIPT MAY NOT BE REAL NUMBER

A subscript must be an integer number, that is a number without a decimal point or a scalar that was defined as a number without a decimal point.

ERR93: MAXIMUM OF 50 ITEMS ALLOWED

An attempt was made to input more than 50 items to define a surface.

ERR94: MATRIX EXPECTED

A MATRIX was expected at this point in the statement.

ERR95: ONLY CURVES MAY BE USED WITH THRU

Only curves may be used with a THRU in a surface definition statement.

ERR96: *PAUSE MAY NOT BE ENTERED FROM CONSOLE

A PAUSE command may only be executed from the part program file.

ERR97: INSERT NOT ALLOWED DURING MACRO CALL

When executing a MACRO, INSERT mode is not allowed.

ERR98: NO DATA BASE ASSIGNED WITH WRITE ACCESS

An attempt was made to issue a PUT of geometric data to a data base when no data base file was currently opened in a write mode.

ERR99: NEW GEOMETRY REQUIRES MORE ROOM THAN OLD**ERR100: INVALID CHARACTER IN DATA BASE FILE NAME**

An invalid character was entered as part of a data base name. Valid characters are: A-Z, 0-9 and ".".

ERR101: OLD FILE ASSIGNED TO THIS NO. WAS CLOSED

An attempt to show data about an item on a data base that was closed has been made.

ERR102: DATA BASE FILE NOT OPEN

An attempt to access a data base file that was not open was made.

ERR103: DATA BASE ACCESS NUMBER NOT 1

A data base file access number other than 1, 2 or 3 was used. See the DBFN or DBSHOW statement section for more details.

ERR104: I/O ERROR ON DATA BASE FILE CLOSE

An error condition occurred while trying to close a data base file. See your operating system documentation for detailed information on the I/O error code meaning.

ERR105: I/O ERROR ON DATA BASE FILE OPEN

An error condition occurred while trying to open a data base file. See your operating system documentation for detailed information on the I/O error code meaning.

ERR106: DATA BASE WRITE ACCESS PASSWORD EXPECTED

A write password that is assigned to this data base file was expected. See the person responsible for creating and maintaining your data base files for the correct password. Passwords may be 1-6 characters long.

ERR107: FILE NAMED IS NOT AN NCL DATA BASE FILE

The file name that was given was opened and found to not be an **NCL** formatted data base file. See the person responsible for creating and maintaining your data base files for further information.

ERR108: INVALID DATA BASE WRITE ACCESS PASSWORD

The password that you specified in the DBFN statement didn't match the password assigned to the named data base file. See the person responsible for creating and maintaining your data base files for further information.

ERR109: ONLY ONE DECIMAL POINT ALLOWED

More than one decimal point was found in a real number.

ERR110: DATA BASE GEOMETRY NAME EXPECTED

A valid geometry identification name was expected.

ERR111: ILLEGAL LABEL FOR CURRENT LOOPING REGION

The label was not declared in the current LOOP or MACRO.

ERR112: MUST BE A POSITIVE VALUE

The number or scalar computed to a negative value when a positive value was required.

ERR113: IDENTIFIER NOT FOUND ON DATA BASE(S)

The item you asked for was not found on any currently opened data base(s).

ERR114: INVALID RANGE SPECIFIED FOR 'THRU'

The range of a THRU is incorrect. This could be caused by the items not being of the same type; for example, PT1,THRU, P(3) is not correct.

ERR115: TOO MANY INPUT ARGUMENTS, 11 MAXIMUM

The number of input arguments exceeded 11.

ERR116: LABEL NOT ALLOWED ON CONTINUED LINE

A label may only occur as the first item of a logical statement.

ERR117: INITIAL MOVE DIRECTION INDEFINITE

A forward sense has not been established. This may be done by an initial tool movement or an INDIRV or INDIRP statement.

ERR118: ILLEGAL GEOMETRY TYPE FOR DS OR CS

The only valid geometry types for a check surface or a drive surface are: LINE, CIRCLE, PLANE, CURVE and SURF.

ERR119: INVALID CUTTER DEFINITION

The input arguments do not define a valid cutter; for example, the corner radius may be larger than the tool radius.

ERR120: TASK IS UNKNOWN TO MOTION SECTION

The statement is asking to do an operation that **NCL**'s motion generation section is not able to process.

ERR121: DIRECTION NUMBERS TOO SMALL

The input numbers must be large enough to define the vector. For example, VE/ 0, 0, .000001 is not large enough.

ERR122: ILLEGAL GEOMETRY TYPE FOR PART SURFACE

A part surface may only be a PLANE or SURF.

ERR123: TLAXIS I, J, K VALUES TOO SMALL

The input numbers must be large enough to define the vector. For example, TA/ 0, 0,.000001 is not large enough.

ERR124: FWD SENSE HAS BEEN LOST. CANNOT RECOVER

This error can be caused by a number of different conditions. As the motion section of **NCL** is computing the tool path along the part surface/ drive surface pair, it keeps track of its forward sense (direction) while looking at the check surface. If the forward sense is not clear at any of the iterations, the motion section stops and puts out this error. A common reason for this to occur is if the tool at any time becomes parallel to the part surface or perpendicular to the drive surface. This error can also occur if the tool is moving in a manner such that it cannot clearly see the check surface. If none of these reasons have caused the error, check the surfaces by displaying them and make sure that they are correct.

ERR125: FWD SENSE HAS REVERSED. CANNOT RECOVER

This error can be caused by a number of different conditions. As the motion section of **NCL** is computing the tool path along the part surface/ drive surface pair, it keeps track of its forward sense (direction) while looking at the check surface. If the forward sense suddenly reverses on any of the iterations, the motion section stops and puts out this error. A common reason for this to occur is if the tool at any time becomes parallel to the part surface or perpendicular to the drive surface. This error can also occur if the tool is moving in a manner such that it cannot clearly see the check surface. If none of these reasons have caused the error, check the surfaces by displaying them and make sure that they are correct.

ERR126: TLAXIS/ MODE NOT IMPLEMENTED

The TLAXIS specified is in development, but not yet implemented.

ERR127: TOOL IS OUT OF BOUNDS OF GEOMETRY

The cutter is not in the correct position to perform the requested operation. Check its position.

ERR128: SURF SOLUTION FAILED. INVALID PATCH?

During the specified operation, **NCL** detected an error in building or using the surface. Check the surface by displaying it.

ERR129: TOOL TO DRIVE SURFACE APPLICATION FAILED

During the requested motion, the motion section was unable to properly apply the tool to the drive surface. Check the relationship between the drive surface and part surface and the position of the cutter.

ERR130: INVALID SURF DEFINING GEOMETRY TYPE(S)

An invalid combination of geometry types was included in a SURF definition. Check the manual for the valid combinations.

ERR131: PATCH BREAKUP FAILED IN SURFACE, OVER 20

The surface section of **NCL** had a problem in building the specified surface. This is usually because the input geometry items define a very severe set of conditions. Check the input.

ERR132: CRVPNT - POINT IS NOT ON CURVE

A point specified to be on a curve is not on the curve.

ERR133: SURFACE TYPE CANNOT BE MIXED WITH OTHERS

This error message is not used at this time.

ERR134: TOOL TO CURVE APPLY FAILURE

A point on the CURVE was attempting to be generated but could not be for some undetermined reason.

ERR135: OUT OF BOUNDS OF THIS CURVE

A point on the CURVE was attempting to be generated but could not be because it was not within the boundaries of the CURVE.

ERR136: TANGENT TO CIRCLE NOT FOUND

This error message is not currently being used at this time.

ERR137: OUT OF BOUNDS OF THIS CIRCLE

A point on the CIRCLE was attempting to be generated but could not be because it was not within the boundaries of the CIRCLE.

ERR138: TOOL TO SURFACE APPLY FAILURE

The motion section of **NCL** is having trouble applying the cutter to the surface. A typical cause for this is when attempting to drive the cutter on the extensions of a SURFace. The solution in this case is to move the cutter so that it is in-bounds of the surface.

ERR139: CURVPV DID NOT CONVERGE IN 12 TRIES

The **NCL** motion routines cannot properly locate the input curve. Sometimes this is because it is trying to locate the tool on an extension of the CURVE. This can be avoided by either positioning the tool on the real CURVE instead of the extension, or reducing the MAXDP for that move.

ERR140: PASS STARTPT DID NOT RESOLVE

The **NCL** motion routines cannot properly locate a valid starting point for this motion statement.

ERR141: NUMPTS EXCEEDED

The current NUMPTS was exceeded during the cutting pass. This is usually due to an error in check surface location.

ERR142: TOOL TO PS/ DS NO GOOD AT START OF PASS

The cutter position in relationship to the drive surface and the part surface is not acceptable. Check the position of the cutter and the relationship of the drive surface to the part surface.

ERR143: GIVEN GEOMETRY GENERATES IMPOSSIBLE SURF

The combination of input geometry defines an unacceptable SURFace. Check the input geometry items.

ERR144: VECTOR OR POINT INVALID AS SURF BOUNDARY

A VECTOR or a POINT cannot be used to define the boundary of this SURFace. A POINT may be used to define one boundary of a simple ruled SURFace. A VECTOR may not be used to define the boundary of any SURFace. A VECTOR or POINT may always be used to indicate the slope at any boundary.

ERR145: ERROR IN OPENING NAMED FILE

An I/O error occurred while trying to open this file.

ERR146: NO INDEX IN EFFECT FOR THIS SCALAR

There is no "INDEX/" statement in effect that uses the number given in this statement. Check to make sure that an "INDEX/ x, NOMORE" statement has not already been issued.

ERR147: 'NOMORE' WORD EXPECTED

The vocabulary word NOMORE was expected at this position in the statement.

ERR148: SCRUB. PS NORMAL OPPOSES TOOL IJK

During the requested SCRUB operation, the cutter became parallel with the surface to be scrubbed. Check the SURFace and the requested tool axis.

ERR149: MOVE ABORTED, NO CLFILE DATA GENERATED

The requested move was aborted by using the "esc" key. The cutter is repositioned at the start of the pass and no CLfile data was generated.

ERR150: INDEX VALUE PREVIOUSLY USED

The number given has been used by a previously processed "INDEX/" statement.

ERR151: MORE THAN 5 DIGITS IN INDEX NUMBER

A number was used as an INDEX number that was greater than 99999.

ERR152: INDEX REQUESTED HAS NO ENDING POINT

The INDEX number given refers to an "INDEX-INDEX/ NOMORE" region that has not been defined yet. The "INDEX/ NOMORE" statement hasn't been processed yet.

ERR153: NOMORE ALREADY ISSUED FOR THIS INDEX NO.

An "INDEX/ x, NOMORE" statement for this INDEX number has already been processed.

ERR154: 'TRANSL', 'SAME' OR 'MODIFY' EXPECTED

One of the vocabulary words mentioned was expected at this spot in the statement.

ERR155: INVALID INDEX NUMBER

A value was given for the INDEX number that was not valid. Valid INDEX numbers may be between 1 and 99999.

ERR156: MORE THAN 20 S-VALUES GENERATED

In defining a curve or surface, **NCL** exceeded its internal limit for that item. This is usually because the geometry item being defined has too severe of conditions.

ERR157: 'NOMORE' OR END OF STATEMENT EXPECTED

The vocabulary word NOMORE or nothing was expected at this spot in the input statement.

ERR158: 'TLAXIS' NOT ALLOWED IN THREE-AXIS VERSION

The specified tool axis cannot be used in the 3-axis version of **NCL**.

ERR159: CIRCLE EXPECTED

A CIRCLE was expected at this spot in the statement.

ERR160: NO HELP INDEX ENTRY FOUND

The input HELP item is not in the **NCL** HELP directory. Enter "*HELP" to see which items are valid.

ERR161: CIRCLE IS TIPPED FROM Z-PLANE

The specified circle is not in the XY plane and that is required for this statement.

ERR162: POINT MUST BE OUTSIDE CIRCLE

The point specified was found to be inside the circle and that is invalid for this construct.

ERR163: IMPOSSIBLE GEOMETRY CASE

The generated geometry item is invalid. Check the input geometry items.

ERR164: COORDINATE PLANE SELECTOR EXPECTED

See matrix definition syntax rules for valid options.

ERR165: CIRCLE MUST HAVE POSITIVE RADIUS

The specified circle has a radius that is negative or zero.

ERR166: LINE APPEARS ON END

The line specified is perpendicular to the XY plane and therefore is invalid for the requested definition.

ERR167: LINES ARE PARALLEL

The specified lines do not intersect.

ERR168: CIRCLES ARE CONCENTRIC

The circles specified do not intersect.

ERR169: 'MODE' EXPECTED

The vocabulary word MODE was expected at this point in the statement.

ERR170: '0' OR '1' EXPECTED

A "0" or "1" indicating the type of circular interpolation CLfile output format that is desired. A "0" indicates the type that is used by NCCS postprocessors and a "1"

indicates the type that is used by the Westinghouse or Automation Intelligence Incorporated postprocessors.

ERR171: TANTO REQUIRES TLAXIS/ SAME OR PLANE PS**ERR172: INVALID TYPE FOR TANTO ENDING**

The only valid check surfaces that may be used in a TANTO mode are lines, circles and planes.

ERR173: DS, CS COMBO NO GOOD FOR TANTO ENDING

TANTO was specified as the check surface modifier, however, the geometry of the drive surface and check surface is such that a TANTO condition is impossible.

ERR174: TAPE FOOTAGE EXCEEDS 'TAPEFT' AMOUNT:

The calculated amount of punched tape output since the last BREAK statement has exceeded the amount given in the "SET/ TAPEFT" command.

ERR175: XY, YZ, OR ZX EXPECTED

The valid input at this spot in the input statement is "XY", "YZ" or "ZX".

ERR176: INVALID PLOT OPTION

The specified option is invalid. Check the manual for all valid plot options.

ERR177: INVALID POCKET INPUT PARAMETER

One of the parameters given in the POCKET statement is not valid. Refer to the POCKET section of the manual or help file for more details.

ERR178: DRAFT MINOR WORD EXPECTED

The DRAFT command requires one or more valid minor words. See the section on the DRAFT command for a complete list of valid minor words.

ERR179: SCALAR BETWEEN -360 AND +360 EXPECTED

The only valid values at this point are values between -360 and +360 degrees.

ERR180: LETTER SIZE EXPECTED (1 TO 1000% NORMAL)

A value indicating the size of lettering on a graphic output display must be between 1 and 1000. The number indicates the percentage of the normal lettering size to be used.

ERR181: INVALID PEN NUMBER

Valid pen numbers are:

0-7 for HP7220, Tektronix 4105
0-15 for Tektronix 4107 and 4109
0-255 for Tektronix 4115

Pen numbers are ignored on VT125 terminals.

ERR182: INVALID USE OF VOCABULARY WORD

The specified vocabulary word is invalid at this spot in the input statement.

ERR183: ** ERROR LIMIT EXCEEDED *******

The error limit has been exceeded. See the "SET/ ELIMIT" option.

ERR184: ** WARNING LIMIT EXCEEDED *******

The warning limit has been exceeded. See the "SET/ WLIMIT" option.

ERR185: 4 POINTS REQUIRED FOR SCRUB BOUNDARY

Only four POINTs are allowed to define the area of a SURFace to be scrubbed.

ERR186: SURF EXPECTED

A SURFace is expected at this spot in the input statement.

ERR187: ‘CROSS’ EXPECTED

The vocabulary word CROSS is expected at this spot in the statement.

ERR188: ‘INTOF’ EXPECTED

The vocabulary word INTOF is expected at this spot in the statement.

ERR189: PT, VE, LN, PL OR CI EXPECTED

One of the specified geometry types is expected at this spot in the statement.

ERR190: PLOTTING SUPPRESSED DURING BATCH RUN

The *SET/ PLOT command is ignored when running in BATCH mode.

ERR191: SUBSCRIPT NAME FOR DEBUG EXPECTED

This is an error message that is used by the developers of **NCL** and should not be encountered by an **NCL** user.

ERR192: LINE OR CIRCLE EXPECTED

A LINE or a CIRCLE is expected at this spot in the input statement.

ERR193: 'TANTO' EXPECTED

The vocabulary word TANTO is expected at this spot in the statement.

ERR194: 'LARGE' OR 'SMALL' EXPECTED

The vocabulary word LARGE or SMALL is expected at this spot in the statement.

ERR195: TLAXIS MAY ONLY BE ATANGL TO PART SURF**ERR196: TLAXIS MAY ONLY BE TANTO DRIVE SURFACE****ERR197: 'RADIUS' EXPECTED**

The vocabulary word RADIUS is expected at this spot in the statement.

ERR198: 'PERPTO', 'FAN' OR 'MODIFY' EXPECTED

The vocabulary word PERPTO, FAN or MODIFY is expected at this spot in the statement.

ERR199: TEMP ERR

This is an error message that is used by the developers of **NCL** and should not be encountered by an **NCL** user.

ERR200: ‘MODIFY’ OR END OF STATEMENT EXPECTED

Either the vocabulary word MODIFY or the end of the statement is expected at this point in the statement.

ERR201: CURVE, CIRCLE, LINE OR INTEGER EXPECTED

A CURVE, CIRCLE, LINE, integer name or nested definition is expected at this point in the statement.

ERR202: PLANES WILL NOT BE DISPLAYED

This is an obsolete error message.

ERR203: GEOMETRY TYPE OR ‘ALL’ EXPECTED

A generic geometry type or the vocabulary word ALL is expected at this point in the statement. Refer to the geometry creation and lathe module section for valid geometry types.

ERR204: END OF STATEMENT OR ‘FULL’ EXPECTED

The vocabulary word FULL or the end of the statement is expected at this point in the statement.

ERR205: 9 NESTED ITEMS MAXIMUM FOR EACH TYPE

The maximum of nine nested definitions for any one geometry type was exceeded in this statement.

ERR206: ZSURF/ N DURING REFSYS NOT ALLOWED

A ZSURF with a value is not allowed to be in effect while a REFSYS is in effect. Use ZSURF/ pl when REFSYS is in effect.

ERR207: TLAXIS MUST BE SAME AND PS MUST BE PLANE

When pocketing, the TLAXIS must be in a fixed position. The only valid pocket bottom is a plane.

ERR208: POCKET BOTTOM ANGLED TOO MUCH TO TLAXIS

See POCKET command explanation for more details.

ERR209: NESTED LINE TOO LONG; PLEASE BREAK IT UP

A statement was found to exceed the limit of 255 total characters. Break the statement into multiple statements.

ERR210: NO COPY DATA GENERATED WITHOUT CL FILE

The COPY command requires a CLfile to be being created in order to COPY generated data. Re-run the program and specify a CLfile to be created for the COPY command to generate copied/ transformed data.

ERR211: NEGATIVE BASE CAN'T USE FRACTIONAL EXP.

A negative number cannot be raised to a fractional power.

ERR212: DISPLAY OF SURFACE ABORTED BY USER

The user interrupted the display of a SURFace on the hard copy plotter or a graphics display terminal. No harm was done to the geometric data or motion data. This is an informational message only.

ERR213: ILLOGICAL DIRECTION MODIFIER USED

A direction modifier was specified that is not logical when related to the rest of the statement.

ERR214: CIRCLES DO NOT INTERSECT

The CIRCLEs specified do not intersect at any point.

ERR215: ON, OFF OR POSITIVE INTEGER EXPECTED

The vocabulary word ON, OFF or a positive integer value is expected at this point in the statement.

ERR216: CIRCLE OR END OF STATEMENT EXPECTED

A CIRCLE identifier or the end of the statement was expected at this point in the statement.

ERR217: INTEGER GREATER THAN ZERO EXPECTED

A value larger than zero is required at this point in the statement.

ERR218: ILLEGAL GRAPHICS TERMINAL ID NUMBER

A non-octal or otherwise illegal terminal number was given in the statement.

ERR219: 'ALL' EXPECTED

The vocabulary word ALL was expected at this point in the statement.

ERR220: 'NORMAL' OR 'GRAPH' EXPECTED

The vocabulary word NORMAL or GRAPH was expected at this point in the statement.

ERR221: LETORG IGNORED - OUT OF RANGE

A position was given that specified lettering would start off of the screen or plotter. The "LETORG" command is not acted upon and the next LETTER statement is ignored.

ERR222: INVALID LINE TYPE PARAMETER

The valid values for LINTYP are 0-7. If you are using a VT125 the valid values are 1-6.

ERR223: CIRCLE OR CURVE EXPECTED

A CIRCLE or CURVE identifier was expected at this point in the statement.

ERR224: POSITIVE INTEGER EXPECTED

A positive integer value is required at this point in the statement.

ERR225: 'CROSS', 'PLUS' OR 'MINUS' EXPECTED

The vocabulary word CROSS, PLUS or MINUS is expected at this point in the statement.

ERR226: SCRUB - TLAXIS MUST BE SAME OR NORMAL, PS

The tool axis must be in a fixed position or normal to the part surface during a SCRUB operation.

ERR227: REFSYS NOT ALLOWED DURING THIS STATEMENT

REFSYS must not be in effect during certain operations such as FMILL. Issue a "REFSYS/ NOMORE" command to correct the problem.

ERR228: BALL END CUTTER OR TLAXIS NORMAL REQ'D

The cutter must be defined as a ball end mill during a FMILL operation. This means that the cutter must have a corner radius equal to exactly 1/2 the cutter diameter.

ERR229: TERMNO OR END OF STATEMENT EXPECTED

The vocabulary word TERMNO or the end of the statement is expected at this point in the statement.

ERR230: TERMNO, CUTTER OR END OF STATEMENT EXP.

The vocabulary word TERMNO, CUTTER or the end of statement is expected at this point in the statement.

ERR231: LINE, CIRCLE OR SHAPE EXPECTED

A LINE, CIRCLE or SHAPE variable name or nested definition is expected at this point in the statement.

ERR232: VOCABULARY WORD EXPECTED

A valid vocabulary word is expected at this point in the statement.

ERR233: INVALID VOCABULARY WORD IN SHAPE

A vocabulary word that is not allowed was found in the SHAPE statement at this point.

ERR234: LN, CI, CV, POST WORD OR MODIFIER EXPECTED

A LINE, CIRCLE, CURVE, postprocessor word or a modifier is expected at this point in the statement.

ERR235: STATEMENT ONLY ALLOWED WITH LATHE MODULE

This type of statement is only allowed when using the lathe module version of **NCL**.

ERR236: SHAPE EXPECTED

A SHAPE variable name is expected at this point in the statement.

ERR237: TOO MANY MINOR WORDS - 50 MAXIMUM

More than 50 minor words were found in a postprocessor statement.

ERR238: 'OFF', 'XAXIS' OR 'YAXIS' EXPECTED

The vocabulary word OFF, XAXIS or YAXIS is expected at this point in the statement.

ERR239: INVALID CLAUSE IN LATHE STATEMENT**ERR240: LINE NUMBER OR END OF LINE EXPECTED**

An *EDIT command was expecting either no more data which would indicate the current line should be edited or a number indicating a valid part program line that is to be edited.

ERR241: NO STATEMENT BY THAT NUMBER FOUND

An *EDIT command was given that tried to edit a statement with a line number that is not in the current part program.

ERR242: ENDING 'THRU' RANGE IDENTIFIER EXPECTED

An invalid THRU range was specified.

ERR243: DATA BASE OPERATION ABORTED BY OPERATOR

This warning indicates the operator interrupted the data base GET or PUT operation before it was completed.

ERR244: VALID GEOMETRY NAME EXPECTED

An invalid name was given in a data base GET or PUT command.

ERR245: DEPTH AND CUTANG REQ'D IN LATHE/ ROUGH

The DEPTH and CUTANG clauses are required for the LATHE/ ROUGH statement.

ERR246: CUTANG MUST BE 180

At the current time, the only valid CUTANG value is 180.

ERR247: UNDERCUTS NOT ALLOWED IN LATHE/ ROUGH

An undercut is not allowed in a LATHE/ ROUGH statement. If an undercut is desired in the finished shape, a roughing shape must be defined with no undercut for the LATHE/ ROUGH statement.

ERR248: PP COMMANDS NO GOOD IN SHAPE (TEMP)

This is an error message that is used by the developers of **NCL** and should not be encountered by an **NCL** user.

ERR249: PLANE OR POINT EXPECTED

A PLANE or POINT identifier is expected at this point in the statement.

ERR250: PLANES MUST BE PARALLEL

The two planes in the DIST statement are not parallel.

ERR251: STOCK M*N NOT ALLOWED IN LATHE/ FINISH

Only one stock value is allowed in a LATHE/ FINISH statement.

ERR252: SHAPE MAY NOT START WITH PP-COMMAND

A shape may not start or end with a post processor command.

ERR253: SHAPE MAY NOT END WITH PP-COMMAND

A shape may not start or end with a post processor command.

ERR254: PLANE IS ALMOST PARALLEL WITH TLAXIS

In a G0DLTA/ plane statement, the PLANE specified is within 5 degrees of being parallel with the tool axis.

ERR255: MOTION ABORTED DUE TO DISCONTINUITY

As the motion section of **NCL** moved the tool along the drive-surface/ part-surface toward the check-surface, the points generated changed direction abruptly. A common cause of this error is when the part-surface is flat and a large tool is driven into a surface defined as a fillet. Another cause of this error is to use a radical drive surface. The problem can often be corrected by reducing MAXDP or by increasing the value of TOLER.

ERR256: CHECK SURFACE VERSUS TOOL APPLY FAILURE

A problem was found in determining a final position for the tool in relation to the check surface. Check the motion statement to see if an impossible condition was specified.

ERR257: UNACCEPTABLE TRACUT MX IN USE WITH COPY

A matrix was specified in a COPY statement that would not produce proper results if used by **NCL**.

ERR258: DIRECTION MODIFIER PARALLEL TO LINE

A direction modifier was specified that does not uniquely identify which of two possible cases is the desired one.

ERR259: UNAUTHORIZED USE OF **NCL DETECTED**

NCL has determined that it is running on a system that it has not been authorized to run on. Check with your system administrator or **NCL** representative to correct this problem.

ERR260: THIS ERASE NOT VALID FOR TEK 4014/ 4105

The ERASE/ MOTION and ERASE/ GEO commands are not valid commands for Tektronix 4014 or 4105 type terminals. This type of terminal does not have the required hardware to allow selective erasing of graphic items.

ERR261: 'GRAPH' INVALID FOR NON TEK TERMINAL

The *SET/ PLOT command may not use the "GRAPH" keyword if you are not running **NCL** on a Tektronix type terminal. **NCL** checks to see what type of terminal it is running on when it starts up and Tektronix terminals are the only type that it can positively identify. If a graphic terminal other than a Tektronix model is being used, you must explicitly specify the terminal id. For example, if you are using a Digital VT125 graphics terminal, you must issue a *SET/ PLOT, VT125 command.

ERR262: DISK CUTTER + TANTO/ DS REQUIRES 'TLON'

Since a disk cutter is one that has no "flat" side, there is no tangent point to maintain a contact with the drive surface. Therefore, the only valid drive surface relationship is TLON.

ERR263: DISK CUTTER + FAN REQUIRES 'ON', CS

Since a disk cutter is one that has no "flat" side, there is no tangent point on the side of the cutter to FAN into the check surface. Therefore, the only valid check surface relationship is "ON".

ERR264: TOO MANY OUTPUT CURVES, TRY LARGER TOLER

When using surface fitting, the weed out routine was unable to reduce the number of curves to 20 or less. Sometimes this is because the curves are just too irregular to weed out enough to build the surface within the given program tolerance (TOLER). In most cases, increasing the TOLER will allow the surface fitting routines to weed out enough curves to reduce to 20 or less and still generate an acceptable surface.

ERR265: NESTED ITEM IN MACRO CALL MUST BE NAMED

It is invalid to have an unnamed nested item in a CALL statement.

ERR266: MACRO BY THIS NAME ALREADY EXISTS

A macro by this name has already been defined in this program and it is invalid to redefine it. Use a different name.

ERR267: THE FILE SPECIFIED IS THE WRONG FORMAT

The file specified is not a file containing Rockwell patches in the proper format.

ERR268: CURVE NOT RIGHT FOR THIS CIRCLE DEF**ERR269: PATCH SPECIFIED IS NOT IN FILE**

A patch number given in a Rockwell surface type definition could not be found in the data file named in the surface definition statement.

ERR270: THE SPECIFIED FILE WAS NOT FOUND**ERR271: AN I/ O ERROR OCCURRED DURING FILE OPEN****ERR272: DISTANCE MUST BE GREATER THAN ZERO**

In this statement the generated distance computed to be zero or negative.

ERR273: ZERO VALUE EXPECTED

A scalar value of zero is expected at this point in the statement.

ERR274: PRIMARY FEEDRAT IS 0

The value given in a FEDRAT statement should set the feed rate to zero.

ERR275: STATEMENT IS ONLY VALID WITH DATA TABLET

The statement can only be processed if there is a data tablet in use on the terminal.

ERR276: INVALID ZSURF

ZSURF may only be a PLANE which is not normal to the XY plane.

ERR277: MUST HAVE EVEN NUMBER OF ITEMS IN SF DEF

A surface definition that is not using "FIT" is one that is assumed to be made up of boundary-slope pairs. Therefore, the number of items must be even.

ERR278: STATEMENT ONLY ALLOWED WITH SURF PACKAGE

This type of statement is only valid if your copy of **NCL** includes the optional Airframe Surfaces package. This is an option that allows **NCL** to handle different formats of SURF types that are supplied by major airframe manufacturers.

ERR279: FAN REQUIRES SURF OR PLANE DS AND CS

When using the FAN attribute, the drive surface and check surface must be either SURFaces or PLANEs.

ERR280: INVALID PICK APERTURE VALUE

A value was given for a data tablet pick aperture that was outside the valid range. The valid range is from 1 to 4095.

ERR281: PLANE OR SCALAR EXPECTED

NCL expects to find a PLANE identifier or a scalar expression at this point.

ERR282: SCALAR VARIABLE EXPECTED

NCL expects to find a scalar identifier at this point.

ERR283: GENPTS - MORE THAN 500 POINTS GENERATED

The maximum number of points that can generated in one motion statement while using GENPTS is 500. Use a larger TOLER or break the move into two or more smaller moves.

ERR284: GENPTS - MULTAX REQ'D TO STORE TA VECTOR

MULTAX must be on when using GENPTS to store the tool axis vector.

ERR285: 'STRTPT' EXPECTED

NCL expects to find the vocabulary word STRTPT at this point.

ERR286: 'ENDPT' EXPECTED

NCL expects to find the vocabulary word ENDPT at this point.

ERR287: SURFACE SPECIFIED IS NOT IN FILE

The surface specified in the SURF/ MESH statement is not in the file specified.

ERR288: INCONSISTENT START AND END PATCHES

In the SURF/ MESH statement, the start patch is less than zero, the end patch is greater than the number of patches in the file or the start and end patches do not form a rectangular grid.

ERR289: I/O ERROR READING FILE

An I/O error occurred while reading the surface file. This is usually because the file is not in the correct format.

ERR290: SCRUB NOT IMPLEMENTED FOR THIS SURF TYPE

The SCRUB statement has not been implemented for Net surfaces or Rockwell Quilt surfaces.

ERR291: STRTPT OR ENDPT IS ON CIRCLE CENTER

The STRTPT or ENDPT of a circle in a REDEF statement may not be at the center of the circle.

ERR292: DIRECTION MODIFIER IS INVALID

The direction modifier given does not uniquely identify which end of the piece of geometry is required.

ERR293: DIRECTION MODIFIER AND/OR ENDPTS NO GOOD

Either the direction modifier is not unique or the ENDPT given is not valid.

ERR294: NO OTHER OCCURRENCES OF PATTERN FOUND

The FIND statement did not find any more occurrences of the current find pattern.

ERR295: PATTERN NOT FOUND IN PART PROGRAM

The find pattern specified for the FIND statement does not occur in the part program.

ERR296: INTERNAL ERROR - TOO MANY HELP ENTRIES

NCL generated an internal error while attempting to execute a *HELP statement.

ERR297: COMMA OR END OF STATEMENT EXPECTED

Either a comma or the end of the statement was expected at this point.

ERR298: CIRCLE AXIS VECTOR IS ZERO

The vector specified for the axis of the CIRCLE has either zero or very small length.

ERR299: AXIS AND LIMIT PLANE VECTORS NOT PERP

When defining a CIRCLE by its canonical form, the vectors given for the axis and the limit plane must be mutually perpendicular.

ERR300: LIMIT PLANE DOES NOT CUT CIRCLE

The limit plane specified for the CIRCLE does not cut the CIRCLE.

ERR301: LINE IS PARALLEL TO PLANE

The LINE specified is either exactly or very nearly parallel to the PLANE specified.

ERR302: INVALID WHILE USING IMPLIED CHECK SURF

This statement is invalid while within an implied check surface range.

ERR303: EDIT WORK FILE ALREADY EXISTS**ERR304: LINE DS IS PARALLEL TO TOOL AXIS**

A LINE drive surface may not be parallel to the tool axis.

ERR305: TOOL IS AT CENTER OF DS OR CS CIRCLE

The tool may not start at the center of the circle being used as the drive surface or check surface.

ERR306: INDIRV REQ'D BEFORE GO WITH CURVE CS

An INDIRV or INDIRP statement must be given immediately before a GO statement using a CURVE as a check surface.

ERR307: TLAXIS MUST BE PERP TO CIRCLE DS OR CS

The tool axis must be perpendicular to the plane of the CIRCLE being used as either a drive or check surface.

ERR308: DS MUST BE RULED SURF FOR PARELM CUTMODE

When the TLAXIS/ TANTO, DS, 1, PARELM statement is in effect, the drive surface must be a ruled surface.

ERR309: LEFT PARENTHESIS EXPECTED

A left parenthesis (round bracket) is expected at this point in the statement.

ERR310: RIGHT PARENTHESIS EXPECTED

A right parenthesis (round bracket) is expected at this point in the statement.

ERR311: COMMA EXPECTED

A comma is expected at this point in the statement.

ERR312: INDEX SCALAR TOO BIG FOR THIS GEO TYPE

The index scalar used in the CAN function is greater than the number of elements in the canonical form of the geometry type being used.

ERR313: INVALID GEOMETRY TYPE FOR CAN FUNCTION

CURVEs and SURFaces may not be used in the CAN function.

ERR314: INVALID ARGUMENT FOR TAN FUNCTION

The argument for the TAN function may not be close to 90 degrees or 90 plus a multiple of 180 degrees.

ERR315: FAILED TO PROJECT START POINT TO CURVE

The start point given could not be projected onto the CURVE. This is usually because it is near the center of curvature of a segment of the CURVE. Try moving the start point by a small amount. If it is not already on the curve, try moving it onto the CURVE.

ERR316: ‘CLW’ OR ‘CCLW’ EXPECTED

One of the minor words CLW or CCLW is expected at this point in the statement.

ERR317: VECTOR MUST HAVE REAL LENGTH

The VECTOR defined has a magnitude close to zero.

ERR318: ‘INCHES’ OR ‘MM’ EXPECTED

One of the minor words INCHES or MM is expected at this point in the statement.

ERR319: TOOL IS PARALLEL TO SURFACE

The tool is parallel to the surface. This is not permitted, for example, at the start of a FMILL.

ERR320: OPERATION ABORTED BY USER

The operation was aborted by the user by using the ESC or control-C keys.

ERR321: NOT VALID FOR THIS SURFACE TYPE

The SURFace type used in this statement is not valid. For example, QUILT or NET surfaces may not be used in a NET or OFFSET SURFace definition.

ERR322: PLANE DOES NOT CUT X-Y PLANE

The PLANE in this definition is required to cut the X-Y and it does not.

ERR323: FULL 360 DEGREE CIRCLE IS INVALID HERE

The circle used in this definition must be bounded.

ERR324: LINE, PLANE, CIRCLE OR CURVE EXPECTED

NCL expects to find a LINE, PLANE, CIRCLE or CURVE identifier at this point in the RMILL statement.

ERR325: INTEGER VALUE OF ‘1’ OR ‘2’ EXPECTED

A value of 1 or 2 are the only valid values at this point. A value of 1 indicates lace cutting if this is the RMILL cutting mode parameter or fixed step over value

follows if this is the RMILL step over mode parameter. A value of 2 indicates back and forth cutting if this is the RMILL cutting mode parameter or scallop height value follows if this is the RMILL step over mode parameter.

ERR326: SURFACE OR PLANE EXPECTED

The only valid geometry types that can be used as the part surface in an RMILL statement are SURF and PLANE.

ERR327: FEEDRATE VALUE OR 'RAPID' EXPECTED

A feed rate value, a scalar variable indicating a feed rate or the word RAPID is expected at this point in the RMILL statement.

ERR328: RMILL REQS. TA/ SAME AND BALL-END CUTTER

To do an RMILL operation, the cutter must be defined as a ball-end type and the tool axis must currently be set to TLAXIS/ SAME.

ERR329: RMILL END LIMITS MUST BE CURVES

The geometry that limits the motion of the RMILL operation must be CURVE type geometry.

ERR330: TOOL AXIS MUST BE PARALLEL TO PLANES

The tool axis must be set to be parallel to the PLANEs given to define the beginning/ending and direction of the RMILL motion.

ERR331: FAILED TO FIND PLANE-CURVE INTERSECTION

One of the limit PLANEs given in the RMILL statement does not intersect one of the boundary CURVEs given in the RMILL statement.

ERR332: CLEARPL DOES NOT 'CLEAR' THE SURFACE

The clearance PLANE given in the RMILL statement does not allow for G0DLTA motion to be generated in a positive direction up the tool axis.

ERR333: RANDOM, LINEAR, ARC OR PARLEL EXPECTED

The vocabulary word RANDOM, LINEAR, ARC or PARLEL is required at this point in the PATERN statement. See the PATERN geometry definition section for details.

ERR334: POINT OR VECTOR EXPECTED

NCL expects to find a POINT or VECTOR identifier at this point in the statement. See the PATERN geometry definition section for details.

ERR335: 'INCR' EXPECTED

The vocabulary word INCR is required at this point in the PATERN statement. See the PATERN geometry definition section for details on the various formats using the INCR clause.

ERR336: 'OMIT', 'RETAIN' OR 'AVOID' EXPECTED

The vocabulary word OMIT, RETAIN or AVOID is expected at this point in the PATERN statement. See the PATERN geometry definition section for details on the correct formats.

ERR337: CAN'T 'OMIT' AND 'RETAIN' IN SAME STMT

You cannot have an "OMIT" clause and a "RETAIN" clause in the same PATERN statement.

ERR338: CAN'T 'AVOID' POINT THAT'S BEEN OMITTED

You cannot have a point in an "AVOID" clause that also appears in an "OMIT" clause in the same PATERN statement.

ERR339: INVALID 'THRU' RANGE SPECIFIED

A "THRU" vocabulary word was found without an integer preceding it specifying the beginning POINT of the THRU range.

ERR340: POINT IS OUTSIDE THE PATERN RANGE

The PATERN specified contains a specific number of points and the scalar value given in this statement does not correspond to one of those points. For example: A PATERN may have been specified as containing 10 points. If a GOTO or POINT

definition statement using that PATERN specifies a point in the PATERN larger than 10, this error will be generated.

ERR341: INVALID NUMBER OF POINTS IN PATERN

Too few or too many points were specified in the PATERN statement. The acceptable number of points is between 2 and 32,767.

ERR342: PATERN EXPECTED

A PATERN geometry type identifier is required at this point in this PARREL type PATERN statement. The PARREL type PATERN is based on an previously defined PATERN. See the PATERN geometry definition section for details.

ERR343: ‘AT’ OR SCALAR EXPECTED

A scalar value or variable or the vocabulary word AT must be used at this point in the INCR clause of the PATERN statement.

ERR344: LINES ARE NOT PARALLEL

The LINEs that are specified in the DIST statement are not parallel to each other so there is no constant DISTance to calculate.

ERR345: INVALID COMBINATION OF GEOMETRY FOR DIST

One or both of the geometric entities given in the DIST statement are not valid.

Valid combinations are:

PLANE/PLANE, LINE/LINE, POINT/POINT, POINT/PLANE
or POINT/ SURF.

ERR346: ‘INCR’ OR END OF STATEMENT EXPECTED**ERR347: UNDO ONLY ALLOWED WITHIN DO LOOP**

The UNDO statement has been processed while a DO loop has not been in effect.

ERR348: ‘LOOPND’ NOT ALLOWED WITHIN DO LOOP

A LOOPND statement cannot be processed while a DO loop is in effect. A DO loop implies a looping region and another looping region may not be set up by

using the LOOPST/ LOOPND statements. A secondary (nested) DO loop may be set up inside another DO loop using the DO syntax.

ERR349: 'LOOPST' NOT ALLOWED WITHIN DO LOOP

A LOOPST statement cannot be processed while a DO loop is in effect. A DO loop implies a looping region and another looping region may not be set up by using the LOOPST/ LOOPND statements. A secondary (nested) DO loop may be set up inside another DO loop using the DO syntax.

ERR350: OVERLAPPING DO LOOP RANGE NOT ALLOWED

A nested DO loop must be completely contained within the outer DO loop's limits. See the DO loop statement section for more details.

ERR351: INCREMENT VALUE MAY NOT BE ZERO

The value used as the increment in a DO statement may not be zero. This could potentially lead to an endless loop.

ERR352: MACRO CALL CAN'T BE LAST LINE OF DO LOOP

A MACRO CALL statement cannot be the ending statement of a DO loop. Use a CONTIN statement to end the DO loop and place the MACRO CALL statement before the CONTIN statement.

ERR353: HEIGHT IN TA/ TT CMD TOO SMALL FOR CUTTER

The height value specified in a TLAXIS/ TANTO, DS command is too small for the type of cutter that is defined.

ERR354: FAN MODE NOT ALLOWED WITH BARREL CUTTER

The FAN tool axis mode is not allowed when using a barrel type cutter.

ERR355: AVOID LIST CAN'T BE EMPTY

The AVOID list in the PATERN statement was found to not contain any scalars. At least one point of the PATERN must be included in an AVOID list.

ERR356: TOO FEW SEGMENTS FOR CIRCLE

The parameter given in the *SET/ ADISPL command for the number of segments to display when drawing a circle was too small. The minimum is three (3).

ERR357: TOO MANY SEGMENTS FOR CIRCLE

The parameter given in the *SET/ ADISPL command for the number of segments to display when drawing a circle was too large. The maximum is 500.

ERR358: VALUE IS TOO SMALL (MINIMUM IS 2)

The parameter given in the *SET/ ADISPL command for the number of "U" or "V" segments to display when drawing a surface was too small.

ERR359: VALUE IS TOO LARGE (MAXIMUM IS 400)

The parameters given in the *SET/ ADISPL command for the number of "U" and "V" segments to display when drawing a surface were too large. The product of the "U" and "V" values may not exceed 400.

ERR360: 'LONG' OR 'SHORT' EXPECTED

The FORMAT statement syntax requires a vocabulary word of either "LONG" indicating 6 decimal places output or "SHORT" indicating 4 decimal places output for [PPRINT](#) and [INSERT](#) statements.

ERR361: 'TO' OR SCALAR VALUE EXPECTED

In a *SKIP command, the word "TO" or a line number is expected.

ERR362: REQUESTED LINE NUMBER IS LESS THAN 1

In a *SKIP command, the line number given is less than 1 which is the first line in the part program file.

ERR363: INVALID VALUE FOR DEPTH

The value of the depth parameter in the lathe statement is less than .001 or larger than 10.

ERR364: INVALID ARGUMENT FOR ARC SIN FUNCTION

The value of the argument for the arc sin function is not between the limits of -1 and +1.

ERR365: INVALID ARGUMENT FOR ARC COS FUNCTION

The value of the argument for the arc cos function is not between the limits of -1 and +1.

ERR366: VARIABLE VALUE TOO LARGE

The value of the variable in the [PPRINT](#) or [INSERT](#) statement is not between the limits of -10,000.000 and +10,000.000.

ERR367: POINT OR VECTOR EXPECTED

NCL expects to find a POINT or VECTOR identifier at this point in the statement.

ERR368: THIS FORMAT INVALID WITH 'FIT'

The vocabulary word "FIT" may not be used with this syntax structure.

ERR369: CONIC DATA MUST BE PLANAR

Data used to define a conic curve must be in a common plane.

ERR370: CONIC DEFINING DATA IS NOT GOOD

The data used in a conic geometry definition is invalid for some reason.

ERR371: SURFACE DISPLAYED BUT U & V ARE IGNORED

A DISPLAY statement contains values for the number of U and V traces to be drawn to display the specified SURFace. Only ruled, sculptured and mesh surfaces may use U and V values. These U and V values are ignored and the SURFace is displayed using the appropriate method for this SURFace type.

ERR372: CAN'T 'OBTAIN' THIS VOCABULARY WORD

The OBTAIN command may not be used to assign the value of the word specified. See the OBTAIN section of the help file for a complete list of legal values that may be used with the OBTAIN statement.

ERR373: NUMBER BETWEEN 20 AND 200 REQUIRED

When defining a CURVE as the edge of a SURFace, you must specify a number of points to generate along the CURVE that is being defined. This number must be between 20 and 200.

ERR374: PATERN OR RESERV VARIABLE EXPECTED

The name of a PATERN or a reserved (subscripted) geometry name needs to be given for the NUM function.

ERR375: MOVE COMPLETED WITH MAXDP REDUCED TO

The requested motion was completed but **NCL** automatically reduced the current MAXDP value by the amount shown in order to successfully calculate the steps of the move.

ERR376: PATERN OR RESERV VARIABLE EXPECTED

The name of a PATERN or a reserved (subscripted) geometry name needs to be given for the NUM function.

ERR377: U OR V OFFSET MUST BE BETWEEN 0 AND 1

In defining a CURVE as the edge of a SURFace, you may only give an offset value from the edge of the surface of between 0 and 1. This offset value is approximately the percentage along the surface from one edge to the opposite edge. For example: a value of .25 will produce a curve that is a quarter (25%) of the way across the surface from the edge specified in the CURVE definition.

ERR378: TOO FEW INTOF POINTS FOUND.

The geometry requested could not be calculated. Provide a point near the desired geometry.

ERR379: INCOMPLETE STATEMENT - 'IN' EXPECTED

The PRINT statement for turning the printing of input and output information ON and OFF requires the vocabulary word "IN" (for input) to be after the "ON" or "OFF" word.

ERR380: 'OPEN' OR 'CLOSE' EXPECTED

The *WINDOW control command requires either the word "OPEN" or "CLOSE" or no word at all. No word will cause a default of "open" to be used.

ERR381: TOO MANY DIGITS IN NUMBER

A numeric value was given that contained more digits than are allowed. Up to nine (9) digits plus a positive or negative sign are allowed in the integer (left of the decimal point) portion of a number. Up to twenty (20) characters are allowed for the entire number including the integer and decimal portions, decimal point and sign.

ERR382: 'WARN' OR 'NOWARN' OPTION EXPECTED

The word WARN or NOWARN was expected to be seen on the MAXDP statement at this point. See the explanation in the MAXDP section of the help file for details on the meaning of these words.

ERR383: NOT VALID FOR THIS CURVE TYPE

The specified CURVE cannot be used in this statement. Only CAM defined CURVEs are allowed.

ERR384: GEOMETRY HAS ALREADY BEEN DEFINED

The name specified in the statement has already been used.

ERR385: SURFACES PARALLEL NEAR

The two SURFaces are parallel or nearly parallel and the intersection of them cannot be found. Check the area near the given coordinates.

ERR386: NUMBER OF RAMPS VALUE EXPECTED

The number of ramping moves to enter the pocket area was expected at this point in the POCKET statement.

ERR387: 'RAMP' OR 'PLUNGE' EXPECTED

Either the vocabulary word RAMP or PLUNGE to indicate the type of entry into the pocket area was expected at this point in the POCKET statement.

ERR388: POSITIVE RAMP DISTANCE VALUE EXPECTED

A positive value is the only acceptable value when indicating how far each ramping move is to be during pocketing motion.

ERR389: CLEARANCE LEVEL SCALAR OR PLANE EXPECTED

Either a clearance level value or a PLANE is expected to indicate the plane the tool is to be moved to when locating moves above the pocket area are being made.

ERR390: NUMBER OF STEPS OR DISTANCE EXPECTED

A number of steps (levels or passes) or a distance between each step is expected to indicate how many levels of cutter motion should be calculated for the pocketing motion.

ERR391: POSITIVE RETRACT DISTANCE EXPECTED

A positive value is the only acceptable value when indicating how far away the tool should be from the top of the pocket when it begins its entry motion into the pocket.

ERR392: 'IN' OR 'OUT' SPIRAL DIRECTION EXPECTED

The vocabulary word IN or OUT is expected to indicate what direction the cutter should move in during the spiraling pocketing motion.

ERR393: 'UP' OR 'DOWN' BETWEEN SECTIONS EXPECTED

The vocabulary word UP or DOWN is expected to indicate whether the cutter should be raised to the clearance level between pocket sections.

ERR394: 'SHARP' OR 'ARC' CORNER TYPE EXPECTED

The vocabulary word SHARP or ARC is expected to indicate how motion should be generated in a pocket when going around corners that have angles of less than 130 degrees.

ERR395: POSITIVE MAXIMUM STEPOVER VALUE EXPECTED

A positive value is the only acceptable value when indicating the maximum distance that **NCL** should allow when it calculates the concentric tool paths of the pocketing motion.

ERR396: POSITIVE FINISH PASS THICK VALUE EXPECTED

A positive value is the only acceptable value when indicating the distance away the tool should be from the top of the pocket when it begins its entry motion into the pocket.

ERR397: POSITIVE GENERAL FEDRAT VALUE EXPECTED

A positive value is the only acceptable value when indicating the general feed rate to be used during pocketing motion.

ERR398: POSITION F/R SCALAR OR 'RAPID' EXPECTED

A positive value or the vocabulary word RAPID are the only acceptable values when indicating the positioning feed rate to be used during pocketing motion.

ERR399: RETRACT F/R SCALAR OR 'RAPID' EXPECTED

A positive value or the vocabulary word RAPID are the only acceptable values when indicating the feed rate to be used during retract moves during pocketing motion.

ERR400: ENTRY FEDRAT SCALAR EXPECTED

A value is expected that indicates the feed rate to be used on entry type moves into a pocket region. It may be positive or negative.

If it is negative, it is taken to be a fraction of the given general feed rate.

ERR401: TRANSITION FEDRAT SCALAR EXPECTED

A value is expected that indicates the feed rate to be used on transition type moves in a pocket region. It may be positive or negative. If it is negative, it is taken to be a fraction of the given general feed rate.

ERR402: FINISH PASS FEDRAT SCALAR EXPECTED

A value is expected that indicates the feed rate to be used on the finish pass movement in a pocket region. It may be positive or negative. If it is negative, it is taken to be a fraction of the given general feed rate.

ERR403: TOP OR CLEAR PLANE NOT PERP TO TLAXIS

The top or the bottom plane specified is not perpendicular to the tool axis.

ERR404: INVALID LOGICAL EXPRESSION

The expression in the statement is not a valid logical type expression. Refer to the section of the **NCL** reference manual for details on legal logical expressions.

ERR405: POST WORD SHOULD START IN FIRST COLUMN

This statement MUST have the major postprocessor word start in the first column.

ERR406: UNIBASE FORMAT INCORRECT FOR PLATFORM

The Unibase file that was specified is not in the correct format to be used on this hardware platform. Use the data translator option to convert the Unibase file to the proper format for this hardware.

ERR407: UNIBASE VERSION NUMBER NOT UP TO DATE

The Unibase file that was specified was created with a program that built the Unibase file in a different format than the one that is acceptable to this version of **NCL**. Use the Data Translator option to convert the Unibase file to the proper format that will be acceptable to this version of **NCL**.

ERR408: LINE, CIRCLE OR COMPOSITE CURVE EXPECTED

A LINE, CIRCLE or COMPOSITE CURVE type geometry was expected at this point in the COMPOSITE CURVE statement.

ERR409: COMPOSITE CURVE FAILED TO BE BUILT

The COMPOSITE CURVE failed to be built. Check to ensure the end points of the entities specified meet each adjoining entity.

ERR410: TOO MANY POINTS GENERATED

During the generation of the intersection of the SURFaces specified, too many points were calculated. Check to make sure the SURFaces are proper. A possible way to overcome this error is to make the TOLER value larger but the user is cautioned the resulting geometry may not be within acceptable tolerance.

ERR411: MISMATCHED LOOPING STATEMENT FOUND

While scanning the part program after a *EDT operation was complete, a DO loop, LOOPST/LOOPND or MACRO case was found that did not create a complete looping region. For example, a MACRO statement was found with no terminating TERMAC statement. The operation is completed but the warning should notify the user that some vital statements may have been deleted during the editing session.

ERR412: *EDT NOT ALLOWED WHILE IN NESTED LOOPING

The *EDT command may not be used while the program is in a nested MACRO call or DO loop.

ERR413: OTHER USER HAS DATA BASE OPEN ALREADY

The Geometry Data Base file specified in the statement is already opened by another user in a write mode. Access to the Data Base file is restricted since the other user could be updating the Data Base and the data obtained by this user might be incorrect if the other user is not finished with their update. Wait until the other user has finished using the file then reprocess this statement.

ERR414: NO DATA BASE FILE BY THIS NAME FOUND

The Geometry Data Base filename given was not found in the user's current directory. Check to make sure of the spelling and if the file is in the correct directory.

ERR415: NO LOOP REGION STACK RECORD POINTER

This is an internal **NCL** error. It indicates a failure in the software, not the user's part program. If it is received, please submit a Field Service Report (FSR) along with the part program that was running and any details about when it occurred. It indicates a failure by **NCL** to properly keep track of the statement numbers of the starting and ending limits of a looping region such as a MACRO or LOOP.

ERR416: NO LOOP REGION STACK RECORD EXTENSION

This is an internal **NCL** error. It indicates a failure in the software, not the user's part program. If it is received, please submit a Field Service Report (FSR) along with the part program that was running and any details about when it occurred.

It indicates a failure by **NCL** to properly keep track of the internal work file records that keep track of the statement numbers of the starting and ending limits of a looping region such as a MACRO or LOOP.

ERR417: FAILED TO FIND CIRCLE-CURVE INTERSECTION

ERR418: CANNOT PROJECT POINT ONTO CURVE

ERR419: FAILED TO FIND LINE-CURVE INTERSECTION

ERR420: CURVE OR SURFACE EXPECTED

ERR421: 'FINI' NOT ALLOWED IN LOOP OR MACRO

ERR422: MAX NUMBER OF VOCABULARY WORDS EXPECTED

ERR423: INVALID NUMBER OF MACRO PARAMETERS

ERR424: SCALAR, 'CANCEL' OR 'NOMORE' EXPECTED

ERR425: AUTO NAME GENERATION NOT ALLOWED

ERR426: ATTEMPT TO USE WRONG PATTERN TYPE

ERR427: ERROR ATTEMPTING TO CHANGE VIEW

ERR428: INVALID COLOR PARAMETER

ERR429: MISMATCHED GEOMETRY TYPES

ERR430: INVALID LINE WEIGHT PARAMETER

ERR431: INVALID LAYER PARAMETER

ERR432: INVALID FORMAT NAME

ERR433: 'TLAXIS' OR 'FWD' EXPECTED

ERR434: INVALID VIEW NAME

ERR435: INCORRECT NUMBER OF VIEW SPECIFIED

ERR436: INVALID MINOR WORD

- ERR437: **TRACUT NOT IN EFFECT**
- ERR438: **REFSYS NOT IN EFFECT**
- ERR439: **REQUIRED VIEW PARAMETER NOT SPECIFIED**
- ERR440: **STRING EXPECTED**
- ERR441: **CLOSE QUOTE EXPECTED**
- ERR442: **MIN AND MAX VALUES EXPECTED**
- ERR443: **POINTVECTOR EXPECTED**
- ERR444: **PLANES DO NOT INTERSECT**
- ERR445: **ENTERED VALUE OUT OF RANGE**
- ERR446: **MIN/MAX IS INCOMPATIBLE WITH DEF STRING**
- ERR447: **NO DEFAULT IN EFFECT. ANSWER REQUIRED**
- ERR448: **DEFAULT VALUE AND GEO TYPE INCOMPATIBLE**
- ERR449: **UNIBASE FILE NAME EXPECTED**
- ERR450: **NO UNIBASE FILE IS OPEN**
- ERR451: **ERROR ATTEMPTING TO CLOSE UNIBASE FILE**
- ERR452: **UNIBASE FILE IS ALREADY OPEN**
- ERR453: **LIST OR STAT EXPECTED**
- ERR454: **UNIBASE FILE IS EMPTY**
- ERR455: **'INDEX' EXPECTED**
- ERR456: **GEO PREFIX NAME CANNOT EXCEED 2 CHARS**
- ERR457: **VECTOR, LINE, CIRCLE OR CURVE EXPECTED**
- ERR458: **VECTOR, LINE, CIRCLE OR PLANE EXPECTED**

ERR459: VECTOR, LINE, PLANE OR PNTVEC EXPECTED

ERR460: 'ON' REQUIRED TO CHECK TO A POINT

ERR461: 'PLUS' OR 'MINUS' EXPECTED

ERR462: ']' EXPECTED

ERR463: SCALAR VALUE BETWEEN 0 AND 1 EXPECTED

ERR464: VALUE TOO SMALL (MINIMUM 0.0001)

ERR465: CANNOT TRIM/MIDTRIM CURVE

ERR466: SURFACE EVALUATOR FAILED

ERR467: CONDITIONS TOO SEVERE FOR COMBIN MODE

ERR468: CAN'T COPY UV SPLINE ON SURFACE

ERR469: 'DEFALT', 'ON' OR 'OFF' EXPECTED

ERR470: PART SF CURVE MUST BE CURVE ON DRIVE SF

ERR471: ENTITIES ARE NOT CONNECTED

ERR472: CVONSF DOESN'T EXIST; TRY REDISPLAYING SF

ERR473: POCKET ENTRY VIOLATES GEOMETRY

ERR474: COULD NOT CREATE FILLET

ERR475: DECREASED RAMP DISTANCE/HELICAL RADIUS

ERR476: CANNOT DETERMINE OFFSET DIRECTION

ERR477: SF ANALYSIS FAILED. NOT A PRIMITIVE

ERR478: 'IN' OR 'OUT' EXPECTED

ERR479: 'IN' EXPECTED

ERR480: MUST BE A SURFACE OF REVOLUTION

ERR481: DISCONTINUITY OR CURVE CROSSED BOUNDARY

ERR482: NOT ENOUGH MEMORY FOR SHADING

ERR483: INVALID TOOL NUMBER

ERR484: COULD NOT ACCESS TOOL LIBRARY

ERR485: ERROR READING TOOL LIBRARY

ERR486: COULD NOT FIND REQUESTED TOOL

ERR487: FILLET TOOL AXIS CHANGE EXCEEDS MAXIMUM

ERR488: POCKET: SYSTEM INNERS ARE NOT RIGHT

ERR489: POCKET: CANNOT PROJECT TOOL TO SURFACE

ERR490: POCKET: CANNOT FIND TOOL SURFACE PTS

ERR491: POCKET: ENTRY GOUGES SURFACE BOTTOM

ERR492: POCKET: ENTRY GOUGES GEOMETRY

ERR493: POCKET: TOO MANY INPUT POINTS

ERR494: POCKET: ILLEGAL START POINT

ERR495: POCKET: TOO MANY LOOPS

ERR496: POCKET: TOO MANY POINTS IN LOOP CALC

ERR497: POCKET: ILLEGAL END POINT

ERR498: POCKET: END LOOP TOO SMALL

ERR499: POCKET: POCKET & ISLAND LOOPS DON'T CUT

ERR500: POCKET: CANNOT MOVE BETWEEN LOOPS

ERR501: COULD NOT CREATE STOCK

ERR502: COULD NOT INTERSECT

- ERR503: **COULD NOT CONNECT**
- ERR504: **MISSING ENDIF FOR IF-THEN AT SOURCE LINE**
- ERR505: **IF-THEN NESTED TOO DEEPLY**
- ERR506: **IF-THEN STATEMENT NOT IN EFFECT**
- ERR507: **ELSEIF MUST FOLLOW IF**
- ERR508: **THEN EXPECTED**
- ERR509: **JUMP INTO IF-THEN RANGE NOT ALLOWED**
- ERR510: **POCKET EXIT VIOLATES GEOMETRY**
- ERR511: **POCKET: EXIT FIXED TO AVOID GOUGING**
- ERR512: **COLON EXPECTED**
- ERR513: **TEXT OR TEXT VARIABLE EXPECTED**
- ERR514: **AMPERSAND EXPECTED**
- ERR515: **ERROR CREATING TEXT ENTITY**
- ERR516: **INVALID FORMAT STRING**
- ERR517: **TEXT STRING TOO LONG**
- ERR518: **CANNOT CREATE PROFIL CURVE**
- ERR519: **POCKET: BAD MAX/MIN STEP VALUES**
- ERR520: **WATERLINE: CANNOT ORDER POCKETS**
- ERR521: **WATERLINE: CANNOT DEFINE LEVELS**
- ERR522: **WATERLINE: CANNOT EXPAND CONTOUR**
- ERR523: **ANOTE (ANNOTATION) EXPECTED**
- ERR524: **VALID FONT NAME EXPECTED**

- ERR525: **VALID POSITION SPECIFIER EXPECTED**
- ERR526: **POINT OR PNTVEC EXPECTED**
- ERR527: **COULD NOT UPDATE ATTRIBUTES**
- ERR528: **INVALID OPERATION FOR ANOTE TYPE**
- ERR529: **TERMAC ENCOUNTERED DURING LOOP DEFINE**
- ERR530: **NO MOTION PRODUCED. INVALID PARAMETERS?**
- ERR531: **SOLID EXPECTED**
- ERR532: **COULD NOT SAVE SOLIDS**
- ERR533: **INVALID SOLID FOR CUTTER**
- ERR534: **INVALID MATERIAL PARAMETER**
- ERR535: **ORTHOGONAL MATRIX EXPECTED**
- ERR536: **VALID FILTER ATTRIBUTE EXPECTED**
- ERR537: **COLOR NAME EXPECTED**
- ERR538: **MAXIMUM CUSTOM COLOR (48) IS REACHED**
- ERR539: **COLOR NAME NOT DEFINED**
- ERR540: **'I' EXPECTED**
- ERR541: **COULD NOT DECOMPOSE ENTITY**
- ERR542: **VOLUMILL: INTERNAL ERROR OCCURRED**
- ERR543: **VOLUMILL: OPEN POCKETS NOT ALLOWED**
- ERR544: **VOLUMILL: TOOL DIAMETER TOO SMALL**
- ERR545: **VOLUMILL: NO STOCK DEFINED**
- ERR546: **VOLUMILL: NO PART DEFINED**

ERR547: VOLUMILL: DRILL HOLE DATA INVALID

ERR548: VOLUMILL: DRILL DIAMETER TOO SMALL

ERR549: VOLUMILL: DRILL RANGE TOO SMALL

ERR550: VOLUMILL: DRILL HOLE GOUGES PART

ERR551: VOLUMILL: DRILL HOLE NOT DEEP ENOUGH

ERR552: VOLUMILL: PARAMETERS WILL LEAVE SCALLOPS

ERR553: COMPOSITE SOLID EXPECTED

APPENDIX B: VOCABULARY**B.1 Vocabulary List**

The following table includes all **NCL** VOCABULARY WORDS that are reserved for exclusive use by the **NCL** system.

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
AAXIS	140					X
ABSF	552	ABS	X			
ABSOL	180					X
ACCESS	830					
ACOSF	562	ACOS	X			
ACTIVE	341			X		
ADJUST	159					X
AIR	1011				X	X
ALIGN	1076			X	X	
ALL	816*			X		X
ALTER	951			X		
ANALYZ	874					
ANGLF	559	ANGL	X			
ANOTE	616		X			
ARC	182			X		X
ARCSLP	1029		X		X**	
ARROW	925			X		
AS	832			X		
ASINF	558	ASIN	X			
AT	189			X		X

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	<i>NCL</i> Major	<i>NCL</i> Minor	Post processor Major	Post processor Minor
ATAN2F	565	ATAN2	X			
ATANF	553	ATAN	X			
ATANGL	1	AA		X		X
ATTRIB	879			X		
AUTO	88			X		X
AUTOPS	745		X			
AUXFUN	1022				X	
AVOID	327*			X		X
AXIS	132			X		X
BASE	868			X		
BAXIS	141					X
BEGIN	58			X		
BLACK	904					
BLADE	191			X		X
BLUE	906			X		
BORDER	932			X		
BORE	82					X
BORE7	211					X
BORE8	212					X
BORE9	213					X
BOTH	83			X		X
BOUND	624			X		
BOX	340			X		
BREAK	16				X	
BROWN	912			X		

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
CALL	824		X	X		
CALLS	867					
CAM	237					X
CANF	566	CAN	X			
CANON	645	CA		X		
CAXIS	142					X
CBORE	245					X
CBRTF	569		X			
CCLW	59			X		X
CENTER	634*	CE		X		X
CHAMFR	961			X		
CHECK	1023				X	
CHIPS	331			X		
CHKPTS	669		X			
CHUCK	1073				X	
CIRCLE	607	CI	X	X		
CIRCUL	75			X		X
CLAMP	1060				X	
CLDIST	1077			X		
CLEARP	1004				X	
CLFILE	823					
CLIPF	949		X			
CLONE	576		X			
CLOSE	831			X		
CLPRNT	813					

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	<i>NCL</i> Major	<i>NCL</i> Minor	Post processor Major	Post processor Minor
CLRSRF	1057			X	X	
CLW	60			X		X
COARSE	195					X
COLAPS	343			X		
COLF	942		X			
COLLET	139					X
COLOR	900			X		
COMBIN	1071			X	X	
COMPOS	612			X		
CONE	632			X		
CONIC	611			X		
CONST	328*			X		X
CONTCT	856		X	X		
CONTIN	846		X			
COOLNT	1030	CO			X	
COPY	811		X			
COSF	556	COS	X			
COUPLE	1049				X	
CROSS	819*			X		X
CS	753					
CSINK	119*					X
CSUV	859					
CTRLIN	126					X
CUBE	959			X		
CURVE	608	CV	X	X		

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
CUT	803		X			
CUTANG	160			X		X
CUTCOM	1007				X	
CUTTER	716	CU	X	X		
CYAN	911			X		
CYCLE	1054	CY			X	
CYLNDR	620			X		
DASH	124			X		X
DASHDT	918			X		
DASHLN	917			X		
DASHSP	919			X		
DATA	858		X			
DBFN	829		X			
DBLCIR	957			X		
DEBUGG	1032				X	
DECOMP	618		X			
DECR	65			X		X
DEEP	153					X
DEEPBK	216					X
DEEPCL	208					X
DEFALT	903			X		
DEFNAM	899		X			
DELAY	1010				X	
DELTA	34			X		
DEPTH	510			X		X

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
DIAMTR	205					X
DIMOND	205			X		
DISK	1097					
DISPDB	1090		X			
DISPLAY	1021		X	X	X**	
DISTF	575	DIST	X	X		
DITTO	127					X
DNTCUT	802	DC	X			
DO	844		X			
DOTF	561	DOT	X			
DOTTED	125			X		X
DOWN	113			X		X
DRAFT	1059		X			
DRAG	278					X
DRILL	163	DR				X
DS	729			X		
DSUV	858					
DWELL	197*					X
EDGE	860			X		
ELSE	940		X			
ELSEIF	939		X			
END	499*			X	X	
ENDIF	941		X			
ENDPT	664			X		
ENGAGE	324			X		

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
ERASE	742		X			
ERROR	937					
EXHVVY	922			X		
EXPF	550	EXP	X			
EXTRUD	633			X		
FACE	81			X		X
FAN	723			X		
FEDRAT	1009	FE, FR	X	X	X	
FEDTO	281					X
FEEDZ	219					X
FILLET	402		X	X		X
FILTER	950		X			
FIND	872		X			
FINDEX	573		X			
FINE	193					X
FINI	801				X	
FINISH	323			X		
FIT	834			X		
FIVONE	929			X		
FIXTUR	898		X	X		
FLOOD	89					X
FLUTES	206			X		
FMILL	748		X			
FONT	947			X		
FORMAT	849		X	X		

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
FOUR	928			X		
FROM	701		X			
FRONT	148					X
FWD	651			X		
GENPTS	743		X			
GEO	835			X		
GET	827		X			
GO	702		X			
GOBACK	707	GB	X			
GODLTA	710	GD	X			
GODOWN	709		X			
GOFWD	704	GF	X			
GOFWDA	873	GFA				
GOHOME	17				X	
GOLFT	705	GL	X			
GORGT	706	GR	X			
GOTO	703	GT	X			
GOUGCK	838		X	X		
GOUP	708		X			
GREEN	908			X		
GREY	1104			X		
GRID	67					X
GUIDE	865			X		
HEAD	1002				X	
HEAVY	921			X		

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
HEIGHT	754			X		
HELIX	855			X		
HIDDEN	579			X		
HIGH	62			X		X
HOLDER	157			X		X
HOME	790					X
IF	825		X			
IFTOL	751		X			
IN	652*			X		X
INCHES	303			X		X
INCLUD	744		X			
INCR	66			X		X
INDEX	810		X	X		
INDIRP	712	IP	X			
INDIRV	711	IV	X			
INHIBT	279					
INSERT	1046				X	
INTCOD	1020				X	
INTENS	134					X
INTERP	752			X		
INTF	564	INT	X			
INTOF	727	IO		X		X
INTOF3	758	IO3		X		
INVERS	822*			X		X
INVIS	1096		X			

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
IPM	73					X
IPR	74					X
ISLAND	757			X		
JUMPTO	826		X			
KNIFE	1098					
LABEL	936			X		
LACE	342			X		
LARGE	662*			X		X
LAST	52					X
LATHE	700	LA	X			X
LAYER	902			X		
LAYF	943		X			
LEADER	1013				X	
LEFT	8	LE		X		X
LENGTH	9			X		X
LETDIR	1041			X		
LETORG	1042					
LETSIZ	1040					
LETTER	1043			X	X	
LEVEL	759			X		
LIMIT	1078				X	
LINCIR	95			X		
LINE	605	LI, LN	X	X		
LINEAR	76			X		X
LINTOL	1067				X	

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
LINTYP	131			X		
LINWGT	901			X		
LIST	863					
LNTHF	560	LNTH	X			
LOAD	1075				X	
LOADTL	1055	LT			X	
LOADU	847		X			
LOCK	114					X
LOG10F	568		X			
LOGF	551	LOG	X			
LONG	850			X		
LOOK	871					
LOOPND	809		X			
LOOPST	808		X			
LOW	63			X		X
LPRINT	1065					
LRGDOT	958			X		
LTBLUE	914			X		
LTTAN	913			X		
MACHIN	1015				X	
MACRO	806		X	X		
MAGNTA	909			X		
MAIN	93*					
MANUAL	158					X
MARKER	952			X		

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
MARKF	960			X		
MATERL	584			X		
MATRIX	610	MX	X	X		
MAX1F	570		X			
MAXANG	741		X			
MAXDP	736		X	X		
MAXDPM	1062					
MAXIPM	96					X
MAXIS	870			X		
MAXRPM	79					X
MCHTOL	1016				X	
MDEND	1052				X	
MEDIUM	61			X		X
MESH	839			X		
MIDDLE	891			X		
MILL	151					X
MILLFD	201					X
MILLRP	200					X
MIN1F	571		X			
MINFED	964			X		
MINUS	10			X		X
MIRROR	821*			X		X
MIST	90					X
MM	296*					X
MMPM	315					X

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
MMPR	316					X
MODALS	667	MO		X		
MODCUT	1099		X			
MODE	1003			X	X	
MODIFY	732*			X		X
MODSYS	668		X			
MOTION	836			X		
MOVE	577		X	X		
MULTAX	815		X			
NAME	933			X		
NAMED	931			X		
NEARPT	866		X			
NEGX	657*			X		X
NEYG	658*			X		X
NEGZ	659*			X		X
NEUTRL	166					X
NEXT	162					X
NINTF	567	NINT	X			
NOMORE	53	NM		X		X
NOPOST	814*					X
NOPS	726		X			
NORMAL	820*			X		X
NORMPS	39			X		
NOW	161					X
NOWARN	853			X		

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
NOWRAP	890			X		
NOX	14					X
NOY	15					X
NOZ	16					X
NSHAPE	615	NSH	X			
NSURF	614	NF	X			
NUMF	563	NUM	X			
NUMPTS	737		X	X		
OBTAIN	661	OB	X			
OFF	72			X		X
OFFSET	666*			X		X
OFSETL	275					X
OMIT	172*			X		X
ON	71			X		X
ONCE	325			X		
OPEN	50			X		
OPSKIP	1012				X	
OPSTOP	3				X	
OPTION	144					X
ORANGE	1101			X		
ORIENT	246					X
ORIGIN	1027				X	
OUT	653*			X		X
OVPLOT	1042					
PALLET	239					X

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
PARAB	77					X
PARAMS	934			X		X
PARELM	833			X		
PARLEL	637	PA		X		
PART	260			X		X
PARTNO	1045				X	
PASS	755			X		
PAST	715*			X		X
PATERN	636	PN	X	X		
PEN	128			X	X	
PENDWN	130	PD			X	
PENUP	129	PU			X	
PERCNT	763			X		
PERPTO	630*	PE		X		X
PHANTM	916			X		
PINK	1102			X		
PITCH	1050				X	
PIVOTZ	1017				X	
PLABEL	1061				X	
PLACE	878		X			
PLANE	606	PL	X	X		
PLUNGE	1001			X	X	
PLUS	19			X		X
PNTVEC	613	PV	X	X		
POCKET	738		X			

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
POD	1088		X			
PODDEF	747		X			
PODPTS	746		X			
POINT	603	PT	X	X		
POKMOD	749		X			
POSITN	1072				X	
POSMAP	1068*				X	
POSTN	1024				X	
POSX	654*			X		X
POSY	655*			X		X
POSZ	656*			X		X
PPLOT	1014				X	
PPRINT	1044		X	X	X	
PPTOL	1069*				X	
PREFUN	1048				X	
PRINT	817		X			
PROBE	1081				X	
PROFIL	761		X			
PROJCT	888			X		
PROMPT	935		X			
PS	728			X		
PSEUDO	762			X		
PSIS	713		X			
PSTAN	729					
PSUV	857					

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
PURPLE	1103			X		
PUT	828		X			
QAXIS	1089				X	
QUILL	287					X
QUILT	837			X		
RADIAL	893			X		
RADIUS	23	RA		X		X
RAM	500					X
RAMP	854			X		
RANDOM	326*			X		X
RANGE	145					X
RAPID	5	RP		X	X	
RAPTO	280					X
READ	876					
REAM	262					X
REAR	149					X
RED	907			X		
REDEF	840		X			
REDRAW	924			X		
REFSYS	644	RS	X	X		
REMARK	1091		X	X		
REMOVE	843		X	X		
RENAME	869		X			
REPEAT	1083				X	
RESERV	805		X			

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
RESET	15			X	X	
RESTOR	583			X		
RETAIN	329*			X		X
RETRCT	7	RE		X	X	
RETURN	322			X		
REV	97					X
REVERS	1008		X	X	X**	
REVOLV	861			X		
REWIND	1006				X	
RIGHT	24	RT		X		X
RINDEX	574		X			
RMILL	750		X			
ROTABL	1026	RO			X	
ROTATE	1066				X	
ROTHED	1035				X	
ROTREF	68					X
ROUGH	320			X		X
ROUTE	1205					
RPM	78					X
RTRCTO	295					X
SADDLE	150					X
SAFETY	1028				X	
SAME	730*			X		X
SAVE	582		X	X		
SAVEU	848		X			

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
SCALAR	601			X		
SCALE	25			X		X
SCRUB	739		X			
SEAGRN	915			X		
SELCTL	1056				X	
SELECT	1074				X	
SEQNO	1019				X	
SEQUNC	818		X			
SET	1087		X		X	
SFM	115					X
SHADE	578		X	X		
SHANK	192			X		X
SHAPE	602	SH	X			
SHARP	183			X		
SHIFT	249					X
SHO	740					
SHORT	851			X		
SIDE	94			X		X
SIGNF	572		X			
SINF	557	SIN	X			
SINGLE	926			X		
SIX	930			X		
SIZE	196			X		X
SLOPE	629*					X
SLOT	207			X		

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
SLOWDN	1063				X	
SMALL	663*			X		X
SMILL	756		X			
SMM	505					X
SMOOTH	1085				X	
SOLID	123		X	X		X
SPACE	944			X		
SPHERE	631			X		
SPINDL	1031	SP			X	
SPLINE	628	BSP	X			X
SQRT	555		X			
SQUARE	955			X		
SRFTYP	875					
SRFVCT	857		X			
SSPLIN	619		X			
STACK	927			X		
STAR	953			X		
START	57					X
STAT	864					
STD	920			X		
STEP	92			X		X
STL	330			X		
STOCK	321		X	X		
STOP	2				X	
STRCMP	581		X			

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
STRING	945			X		
STROKE	946			X		
STRTPT	665					
STYLE	948			X		
SUBSTR	580			X		
SURF	609	SF	X	X		
SWAP	111			X		
SYMBOL	877		X			
SYN	804		X			
TABLE	177					X
TANF	554	TAN	X			
TANTO	646*	TT		X		X
TAP	168					X
TAPEFT	660					
TDISTF	585	TDIST	X			
TE	650			X		
TERMAC	807		X			
TEXTF	38		X			
THEN	938					
THETAR	107					X
THICK	717	TH	X	X		
THREAD	1036				X	
THRU	152			X		X
TIMES	28			X		X
TITLES	1092		X			

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
TLAXIS	721	TA	X	X		
TLLFT	719	TL	X	X		
TLOFPS	725		X			
TLON	718	TN	X	X		
TLONPS	724		X			
TLRGHT	720	TR	X	X		
TMARK	1005	TM			X	
TO	714*			X		X
TOLER	731		X	X		
TOOL	617			X		X
TOOLNO	1025				X	
TOOLPN	1053		X			
TORCH	175*					X
TORUS	627			X		
TPI	143					X
TRACUT	812*	TC	X	X		
TRALST	842		X			
TRANS	1037			X	X	
TRANSL	649*			X		X
TRAV	154					X
TRFORM	110					X
TRIAN	954			X		
TURN	80					X
TURRET	1033	TU			X	
TYPEF	98	TYPE	X			

VOCABULARY**APPENDIX B**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
UAXIS	1018				X	
UBFN	862		X			
UNDO	845		X			
UNIT	30			X		X
UNITS	841		X	X		
UNLOAD	10		X			
UP	112			X		X
USONIC	1093				X	
VACUUM	1094				X	
VAXIS	1032					
VECTOR	604	VE	X	X		
VIEW	923			X		
VISIBL	1095		X			
VMPMOD	764		X			
VMPOCK	765		X			
VOCABF	549		X			
VPM3AX	962			X		
WARN	852			X		
WATRLN	892					
WAXIS	1096			X		
WHITE	905			X		
WIRE	1028			X		
WRAP	889			X		
XAXIS	84			X		X
XCOORD	116					X

APPENDIX B**VOCABULARY**

Word	Integer Value	Synonyms	NCL Major	NCL Minor	Post processor Major	Post processor Minor
XLARGE	638*	XL		X		X
XSMALL	641*	XS		X		X
XTYPEF	99	XTYPE	X			
XYPLAN	33					X
XYROT	733*			X		X
XYZPLN	45					X
YAXIS	85			X		X
YCOORD	117					X
YELLOW	910			X		
YLARGE	639*	YL		X		X
YSMALL	642*	YS		X		X
YZPLAN	37					X
YZROT	734*			X		X
ZAXIS	86					X
ZCOORD	118					X
ZIGZAG	170					X
ZLARGE	640*	ZL		X		X
ZONE	760			X		
ZSMALL	643*	ZS		X		X
ZSURF	647		X			
ZXPLAN	41					X
ZXROT	735*			X		X

NOTE:

* : The integer values of these **NCL** words are different from the **PostWorks** postprocessor system. **PostWorks** will interpret them correctly by using an

internal translator if the binary “CL” file is the input file to the **PostWorks** postprocessor.

** : **NCL** will only allow the following commands if DISPLAY and REVERS are used as major postprocessor words.

DISPLAY/ON	or	DISPLAY/OFF
REVERS/ON	or	REVERS/OFF

B.2 Postprocessor Statements

Postprocessor statements are used to pass commands directly to the postprocessor. **NCL** checks only the syntax of a postprocessor statement but does not check the validity of the statement with regards to a particular machine/control combination. Postprocessor commands are passed into the CLfile and are stored according to the ANSI standard. It is the job of the postprocessor to translate the postprocessor statements into the appropriate output codes for a particular machine/control combination. For example:

<u>Postprocessor Statement in NCL</u>	<u>Postprocessor Output</u>
LOADTL/1	T01 M06
COOLNT/ON	M08
SPINDL/2000, CLW	S2000 M03

The valid syntax construct for the postprocessor statement is:

major postprocessor word[/list]

The optional list may be composed of minor postprocessor words and/or scalar expressions. The valid minor postprocessor words vary for each postprocessor. The major postprocessor words **INSERT**, **LETTER**, **PARTNO** and **PPRINT** do not require a ‘/’ after the major word; however, all others do if an optional list follows.

Example Postprocessor Statements

```
COOLNT/ ON
LOADTL/ 1, LENGTH, 5.6
SPINDL/ OFF
MCHTOL/ .001
FEDRAT/ 20
TMARK/ 10
```

B.2.1 Cycle Statements

CYCLE statements control the output of cycle sequences, including automatic and manual cycle.

During playback motion while in an interactive session you can control how the cycle motion will be displayed by setting the CYCLE control inside the *PLAYBACK Form*. If it is set to “DETAILED”, the motion will be simulated according to the **PostWorks** postprocessor cycles syntax as shown below.

```
CYCLE/mode,FEDTO,depth [,IPM ,fd[,rap]] [,RAPTO,r] $  
          dia,ang IPR  
          MMPM  
          MMPR  
          TPI  
  
          [,DWELL,dwl1[,dwl2]] [STEP,peck1[,peck2]] $  
          [,OFFSET,off1[,off2]] [RTRCTO,ret[,plng]] $  
          [,REPEAT,rep] [,START[,seq[,inc[,frq]]]] $  
          [,PARAMS,m1,n1,...,mn,nn]
```

Where:

- BORE = Performs a boring cycle, stopping the spindle at the bottom of the hole and feeding out.
- BORE7 = Performs a boring cycle, stopping the spindle at the bottom of the hole and allowing for a manual feed out.
- BORE8 = Performs a boring cycle, stopping the spindle at the bottom of the hole and Rapid out.
- BORE9 = Performs a back boring cycle, with a tool shift at the top of the hole and the cutting being performed from the bottom of the hole.
- DEEP = Performs a chip breaking drill cycle.
- DRILL = Performs a drilling cycle with rapid out.

FACE	=	Performs a drilling cycle with a dwell at the bottom of the hole.
MILL	=	Performs a positioning only cycle.
REAM	=	Performs a reaming cycle with feed out.
REVERS	=	Performs a left handed thread tapping cycle.
SHIFT	=	Performs a fine boring cycle, orientating and shifting the tool at the bottom of the hole.
TAP	=	Performs a right handed thread tapping cycle.
THRU	=	Performs a deep hole drilling cycle, with a full retract to the clearance plane after each cut.
PARAMS	=	Specific cycle parameters required by certain type of controller such as Siemens 840D.

See the **PostWorks Reference Manual** for details of each of the cycle mode.

B.3

Define Synonyms For Existing Words

There are two ways to define synonyms for **NCL** existing vocabulary words. The first one is by using the SYN command. The second one is adding to the nclvoc.syn file. A total maximum of 1024 vocabulary words and synonyms can be defined within the **NCL** system.

B.3.1

Define Synonyms For Existing Words Inside A Part Program

The SYN statement allows the programmer to define synonyms for existing **NCL** vocabulary words within the part program. The syntax for the SYN statement is:

SYN/new_word,old_word[,new_word,old_word[,...]]

Where new_word is the user defined word (synonym) for the existing word old_word.

For Example:

SYN/PK,POCKET,CN,CLONE

NOTE:

- The newly defined synonym(s) is/are only valid locally inside the program in which this SYN statement is used. The definitions cannot be used for other programs unless the same SYN statement existed in the other programs.
- It is invalid to redefine an existing vocabulary word.
- A maximum of 50 synonyms may be defined within a single SYN statement.
- Multiple SYN statements are allowed. The total number of existing vocabulary words, synonyms and user defined new words cannot be more than 1024.
- The synonyms cannot be more than 24 characters long and with at least one alpha character. Numerical characters are allowed.

B.3.2 Define Synonyms For Existing Words By Using The “nclvoc.syn” File.

Synonyms for existing **NCL** vocabulary words can be defined using the following steps.

1. Check the privilege of the directory where the file nclvol.syn is located. Usually this will be

C:\NCCS\NCL101\interface
2. Check the existence of the file nclvoc.syn in this directory. If this file does not exist, create it. Also check the privilege of this file, one must has the write privilege to this file.
3. Use any text editor to open the file nclvoc.syn.
4. Add the synonyms to the file as follows:

```
new_word    old_word
new_word    old_word
.....
new_word    old_new
```

The new_word is the user defined word (synonym) for the existing old_word. The new_word must start column 1. Then, tab once and enter the old_word.

NOTE:

- The newly defined synonym(s) can be accessed by any part program within the **NCL** system. This is different from the SYN statement which is only effective inside the part program in which it was defined.
- It is invalid to redefine an existing vocabulary word.
- The total number of existing vocabulary words, synonyms and user defined new words cannot be more than 1024.
- The synonym(s) cannot be more than 24 characters long and with at least one alpha character. Numerical characters are allowed.

B.4 Add New Word To The Vocabulary List.

Add new words to the **NCL** vocabulary list by adding the new word(s) to the nclvoc.syn file. A total maximum of 1024 vocabulary words and synonyms can be defined within the **NCL** system.

1. Check the privilege of the directory where the file nclvol.syn is located. One must has the write privilege to this directory. Usually this will be

C:\NCCS\NCL101\interface

2. Check the existence of the file nclvoc.syn in this directory. If this file does not exist, create it. Also check the privilege of this file, one must has the write privilege to this file.
3. Use any text editor to open the file nclvoc.syn.
4. Add the new word(s) to the file as follows:

```
new_word    integer_value  
new_word    integer_value  
....  
new_word    integer_value
```

Where new_word is the user defined word(s) that needed to be added and the integer_value is the corresponding number(s) to the new_word(s).

The new word must be start in column 1. Then, tab once and enter the corresponding number.

The corresponding number must be unique. **NCCS** recommends using numbers from 1300-1399. This number is what **NCL** actually uses for processing.

The following files should be looked at to make sure the number chosen is unique.

C:\NCCS\NCL101\interface\nclvoc.ncl

C:\NCCS\NCL101\interface\nclvoc.syn

The following file should be checked to make sure the number chosen is unique if the **PostWorks** postprocessor is used.

C:\NCCS\PostWorks\dsup\postvocab.txt

C:\NCCS\PostWorks\data\postvocab.syn

NOTE:

- The newly defined word(s) can be accessed by any part program within the **NCL** system.
- The total number of existing vocabulary words, synonyms and user defined new words cannot be more than 1024.
- The new word(s) cannot be more than 24 characters long and with at least one alpha character. Numerical characters are allowed.

APPENDIX C: PREVIOUS VERSION COMPATIBILITY

As with every release of **NCL** extreme efforts are made to insure that programs written in previous versions of **NCL** produce the same results in the new version. However, because of the mathematical complexities involved with modifying tool path and geometry algorithms, 100% compatibility cannot be guaranteed.

If one experiences program failures running programs created in a previous version, one may want to include the appropriate version flag in the program. Following are valid version flags:

*SET/VER,8.1
*SET/VER,8.2
*SET/VER,8.209
*SET/VER,8.3
*SET/VER,8.4
*SET/VER,9.0
*SET/VER,9.007
*SET/VER,9.008
*SET/VER,9.1
*SET/VER,9.2
*SET/VER,9.3
*SET/VER,9.4
*SET/VER,9.5
*SET/VER,9.6
*SET/VER,9.7
*SET/VER,9.8
*SET/VER,9.9
*SET/VER,10.0

The version flag will cause certain **NCL** motion routines to use the same code as was used in the version specified by the version flag. This may not solve all potential problems and it may be necessary to make slight alterations to some programs.

C.1 Previous Version Unibase Files

NCL V8.0 to V9.9 can be loaded into **NCL** V10.0

When running old programs through **NCL** V10.0 it is strongly recommended that one compares the output results (post processed data) with the results from the previous version.

C.2 Mesh Surfaces

This section discusses the implementation and use of the Mesh surface. This requires the purchase of the separately priced Mesh Surface package. This surface is made up of patches provided in geometric form by the former McDonnell Douglas or Northrop companies. **NCL** allows these patches to be input directly to form an **NCL** surface definition. This surface may then be used as a drive, check or part surface during tool motion.

C.3 Loading A Mesh Surface File

The first step in building a Mesh surface is to load the file that was provided by McDonnell Douglas or Northrop onto the computer's disk. The steps to do this vary, depending upon the configuration of your computer and the form in which the file has been provided. Contact your **NCCS** support representative for help in your particular situation.

C.3.1 Converting The Mesh Surface File

The file must be converted into **NCL** format before it can be used by **NCL** to create a surface. This is achieved by running the Mesh Conversion program MCV. At the system prompt enter:

MCV and [RETURN]

MCV will respond with a heading and then request that you enter the input file name. Enter the name of the file that you copied onto your disk and hit return. MCV will then ask for the name of the output file you wish to create. This is the file that you use in your *SURF/MESH* statement in **NCL**. Enter an appropriate file name and hit return. If you hit return without entering a file name, MCV will create a file named MESH.DAT.

C.3.2 Building The Mesh Surface

The file which was provided to you contains one or more surfaces consisting of parametric representations of grids of patches. The *SURF/MESH* statement enables you to build **NCL** surfaces from this data. The first step in building the surface is to determine which patches are necessary. Once this has been done, the surface may then be defined in an **NCL** statement with the following syntax:

SURF/MESH, file-name, surf-name [, start-patch[, end-patch]]

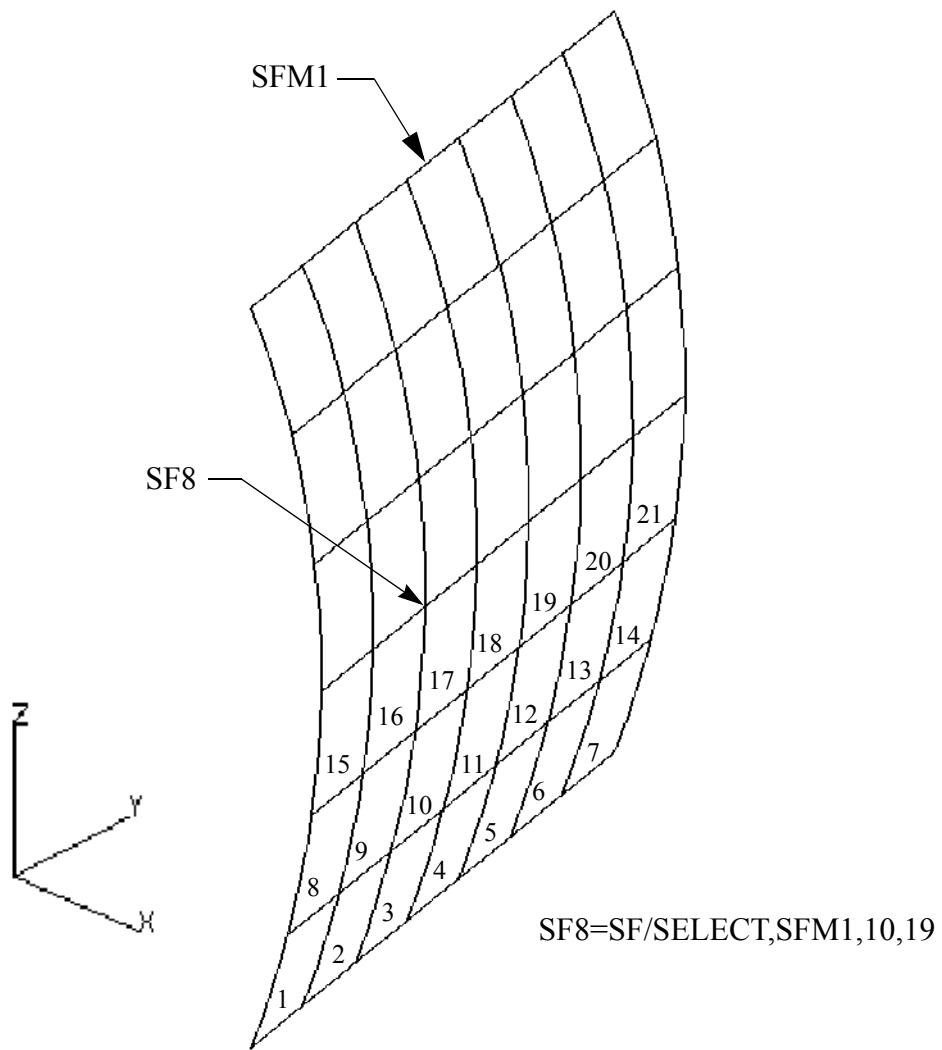
Where:

file-name - is the name of the file output by MCV.

surf-name - is the name of the surface within the file that you wish to create as an **NCL** surface.

start-patch - is an optional integer specifying the number of the first patch to be used in creating the **NCL** surface.

end-patch - is an optional integer specifying the number of the last patch to be used in creating the **NCL** surface.



If both *start-patch* and *end-patch* are omitted then the entire MESH surface is used to create the **NCL** surface. If only one is specified, then the **NCL** surface will consist of just that one patch. When both *start-patch* and *end-patch* are given then the **NCL** surface will be created from a grid of patches in which the first patch of the first row will be *start-patch* and the last patch of the last row will be *end-patch*. The *start-patch* and *end-patch* numbers must define a rectangular array of patches with the *start-patch* being closer than the *end-patch* to the u=0, v=0 corner of the defining MESH surface.

For example, if a MESH surface consisted of an 8 by 3 grid of patches (i.e. 3 rows with 8 patches in each row), then row number 1 will contain patch numbers 1 thru 8, row number 2 will contain patch numbers 9 thru 16 and row number 3 will contain patch numbers 17 thru 24. In this case, if a *start-patch* of 2 and an *end-patch* of 20 were specified, the resulting **NCL** surface would consist of 3 rows of 3 patches with the first row containing patches 2,3 and 4, the second row containing patches 10,11 and 12 and the last row containing patches 18, 19 and 20. If a *start-patch* of 2 and an *end-patch* of 10 were given, then the resulting **NCL** surface would contain 2 rows of 1 patch each, with the first row containing patch 2 and the second row containing patch 10.

Examples:

SF1=SURF/ MESH, MESH1.DAT, FUS1

Creates **NCL** surface SF1 using all the patches of MESH surface FUS1 in file MESH1.DAT

SF2=SURF/ MESH, MESH1.DAT, FUS1, 10

Creates **NCL** surface SF2 using patch 10 of MESH surface
FUS1 in file MESH1.DAT

SF3=SURF/ MESH, MESH1.DAT, FUS1, 3, 4

Creates **NCL** surface SF3 using patches 3 and 4 of MESH surface FUS1 in file MESH1.DAT

SF4=SURF/ MESH, MESH1.DAT, FUS1, 2, 20

Creates **NCL** surface SF4 using a selection of patches from MESH surface FUS1 in file MESH1.DAT. If FUS1 is an 8 by 3 grid of patches, then patches 2, 3, 4, 10, 11, 12, 18, 19 and 20 will be selected.

C.3.3 Displaying The Mesh Surface

MESH surfaces display differently than regular **NCL** surfaces. Only the outlines of the individual patches making up the surface are displayed. Usually the surfaces provided by McDonnell Douglas or Northrop are in the airplane system, which means that to display them on the terminal requires scaling them down, and either translating them through a **REFSYS** matrix or declaring the origin to be well off of the screen.

C.3.4 Using The Mesh Surface

Once the surface has been declared, it can be used like any other **NCL** surface. There is no limit to the number of patches that can be made into a MESH surface. However, the user may notice that **NCL** is relatively slow generating motion using a surface made up of a large number of patches, especially when the surface is used as the part, drive or check surface for the first time.

C.3.5 Example Mesh Surface File

The following shows an example of the start of a MESH surface file. The first line contains the word “SURFACE” followed by the name of the surface, in this case FUS1. This is the name that should be used as surf-name in the SURF/ MESH statement. The second line specifies the number of patches per row and the number of rows, in this case 8 and 3. The third and subsequent lines contain the parametric data describing the surface. There are 16 lines of data for each patch, with the first, second, fifth and sixth lines in each group of 16 specifying the X, Y and Z coordinates of the corner points of the patch. Additional surfaces in the same format may follow the first.

```
SURFACE FUS1
```

```
8 3
```

9.1789530000	163.9000000000	18.5929530000	1	1
11.8152300000	185.0000000000	12.2710010000	2	1
2.6341753006	21.0999298096	6.3229942322	3	1
2.6375846863	21.1056518555	6.3236398697	1	1
12.9810020000	163.9000000000	9.4139940000	1	1
16.7092610000	185.0000000000	0.4557630000	6	1
3.7254371643	1.1007843018	8.9575347900	7	1
3.7292919159	21.1010284424	8.9592676163	8	1
7.3032331467	0.0	7.3032331467	9	1

9.4007863998	0.0	9.4007863998	10	1
2.0106554031	0.0	2.0106554031	11	1
2.1820812225	0.0	2.1820812225	12	1
0.0000004156	0.0	10.3283309937	13	1
0.0000005349	0.0000000000	13.2947196960	14	1
0.0000001150	0.0	2.8571596146	15	1
0.0000001236	0.0000000000	3.0720119476	16	1
•				
•				
•				

C.4 Quilt Surfaces

This section discusses the implementation and use of the Quilt surface. It required the purchase of the separately priced Quilt Surface package. This surface is comprised of patches provided in parametric form from Rockwell International. **NCL** allows these patches to be input directly to form an **NCL** surface definition. This surface may then be used as a drive, check or part surface during tool motion.

C.4.1 Loading The Quilt Surface File

The first step in building a Quilt surface is to load the file that was provided by Rockwell onto the computer's disk. The steps to do this vary, depending upon the configuration of your computer and the form in which the file has been provided. Contact your NCCS support representative for help in your particular situation.

C.4.2 Converting The Quilt Surface File

The file must be converted into **NCL** format before it can be used by **NCL** to create a surface. This is achieved by running the Quilt Conversion program QCV.

Enter QCV at the system level.

QCV will respond with a heading and then request that you enter the input file name. Enter the name of the file that you copied onto your disk and hit return. QCV will then ask for the name of the output file you wish to create. This is the file that you use in your SURF/ QUILT statement in **NCL**. Enter an appropriate file name and hit return. If you hit return without entering a file name, QCV will create a file named QUILT.DAT.

C.4.3 Building The Quilt Surface

The file which was provided by Rockwell will be made up of individual patches. The number will vary, and you may or may not need all of these patches for your particular part. The first step in building the surface is to determine which patches are necessary. Once this has been done, the surface may then be declared in an **NCL** statement with the following syntax:

```
SURF/QUILT, filename, patch-number-1, ...patch-number-n
```

The *filename* is the name of the file on disk that was output by the QCV Quilt Conversion utility. The resulting surface will contain all patches specified by patch- numbers. Currently, there can be no more than 12 patches in a surface.

Example

```
SF1=SURF/ QUILT, ROCK1.DAT, 320257, 320258, 320263, $  
320273, 320274, 320275, 320276
```

C.4.4 Displaying The Quilt Surface

Quilt surfaces display differently than regular **NCL** surfaces. Only the outlines of the individual patches making up the surface are displayed. Usually the surfaces are provided by Rockwell in the airplane system, which means that to display them on the terminal requires scaling them down and either translating them through a **REFSYS** matrix or declaring the origin to be well off of the screen.

C.4.5 Using The Quilt Surface

Once the surface has been declared, it can be used like any other **NCL** surface, with a few exceptions. It cannot be used with the **SCRUB** statement, and in general the tool should always be kept in-bounds of the surface to avoid driving on the extensions.

Quilt surfaces cannot be used in a **NET** surface or with the CV as an edge of a surface definition.

C.4.6 Example Quilt Surface File

The following example file contains two patches; patch 5011 and patch 5740, and shows the format of the file once it is loaded onto the disk:

APPENDIX C**PREVIOUS VERSION COMPATIBILITY**

PATCH 5011

FWD FUS PATCH YF250-YF305 MHB-LWR CL			25	5011
11/26/84 14:31 KNUDSON	B5	5011		
41.010303400000	250.0000000000000	29.726202000000	5011	1
42.056568100000	263.747772200000	30.578781100000	5011	2
43.010505600000	277.499847400000	31.355705200000	5011	3
43.970893800000	291.251525800000	32.135475100000	5011	4
44.898998200000	305.0000000000000	32.788700100000	5011	5
41.010303400000	250.0000000000000	24.144760100000	5011	6
42.056568100000	263.747772200000	24.528287800000	5011	7
43.010505600000	277.499847400000	24.751125300000	5011	8
43.970893800000	291.251525800000	24.978862700000	5011	9
44.898998200000	305.0000000000000	24.972930800000	5011	10
39.488822900000	249.999969400000	18.798044200000	5011	11
40.453613200000	263.747955300000	18.858077900000	5011	12
41.365619600000	277.499816800000	18.777843400000	5011	13
42.284339800000	291.251464800000	18.701663900000	5011	14
43.201541800000	305.0000000000000	18.416322700000	5011	15
37.958518900000	250.0000000000000	13.454053800000	5011	16
38.841598400000	263.747772200000	13.191402400000	5011	17
39.713539100000	277.499847400000	12.808225600000	5011	18
40.592094400000	291.251525800000	12.428487700000	5011	19
41.501522000000	305.0000000000000	11.861743900000	5011	20
35.030406900000	250.0000000000000	8.777999800000	5011	21
35.748573300000	263.747772200000	8.335167800000	5011	22
36.516223900000	277.499847400000	7.830894400000	5011	23
37.291175800000	291.251525800000	7.329905400000	5011	24
38.140102300000	305.0000000000000	6.705500100000	5011	25

PATCH 5740

UPR FWD FUS PATCH YF273-YF295 (5)			25	5740
11/26/84 14:31 KNUDSON	B5	5740		
41.792700000000	273.0000000000000	53.3774000000000	5740	1
42.293600000000	278.4999000000000	53.8106000000000	5740	2
42.738600000000	284.0000000000000	54.2235000000000	5740	3
43.184200000000	289.5000000000000	54.6364000000000	5740	4
43.575500000000	295.0000000000000	55.0299000000000	5740	5
42.254400000000	273.0000000000000	48.4129000000000	5740	6
42.699900000000	278.4999000000000	48.8081000000000	5740	7
43.112500000000	284.0000000000000	49.1911000000000	5740	8
43.525800000000	289.5000000000000	49.5696000000000	5740	9
43.909700000000	295.0000000000000	49.9262000000000	5740	10
42.473000000000	273.0000000000000	42.9822000000000	5740	11

42.8904000000000	278.5000000000000	43.3702000000000	5740	12
43.2873000000000	284.0000000000000	43.7361000000000	5740	13
43.6845000000000	289.5000000000000	44.1036000000000	5740	14
44.0641000000000	295.0000000000000	44.4464000000000	5740	15
42.6933000000000	273.0000000000000	37.5571000000000	5740	16
43.0826000000000	278.4999000000000	7.9389000000000	5740	17
43.4638000000000	284.0000000000000	38.2882000000000	5740	18
43.8448000000000	289.5000000000000	38.6456000000000	5740	19
44.2203000000000	295.0000000000000	38.9750000000000	5740	20
42.6933000000000	273.0000000000000	31.1006000000000	5740	21
43.0826000000000	278.4999000000000	31.4154000000000	5740	22
43.4638000000000	284.0000000000000	31.7151000000000	5740	23
43.8448000000000	289.5000000000000	32.0147000000000	5740	24
44.2203000000000	295.0000000000000	32.2954000000000	5740	25
•				
•				
•				

C.4.7 **Obsolete Data Base File Name (DBFN)**

The DBFN statement is the predecessor to [UBFN](#) statement and works in similar fashion. However, the DBFN statement does not store geometry attributes or NURB curves and surfaces. The DBFN statements are obsolete and are supported only for the purpose of providing compatibility with older programs. It is recommended that they not be used on new projects. This command is only supported in the UNIX platform. This command is not supported in the PC Window platforms

The old syntax was as follows:

```
DBFN/geometry_data_base_file_name[,write password]
```

All other DBFN statements are similar to corresponding Unibase statements.

C.5 **Obsolete Commands**

The following commands have been made obsolete with the implementation of new features and new commands. The older commands will continue to be supported as they currently function for the purpose of running older programs. However, it is recommended that the following commands no longer be used on new projects.

C.5.1 *DBSHOW

The DBSHOW command is an obsolete command that was used to display information about a geometric entity stored in a **NCL** data base file.

C.5.2 *QUIT

QUIT is an obsolete command that was used to exit **NCL** in earlier versions. To exit **NCL** use the *MAIN / EXIT NCL* menu or the *FILE / EXIT* menu when using the Motif interface.

**C.5.3 REDEF/line , point-1, point-2
 circle**

This statement causes the line or circle to be redefined to go from point-1 to point-2.

If a circle is being redefined, the resulting circle will be the shortest arc between the two points. The points may lie on or off the line or circle being redefined. If a point does not lie on the line or circle, it is projected normal to the line or circle to determine the start or end point.

In releases prior to version 7.0 the words STRPT and ENDPT were allowed as optional vocabulary words before point-1 and point-2. As of release version 7.0 these optional words are no longer valid in the REDEF statement.

C.5.4 REDEF/ line, geometry [, near-pt] [, modifier-1 [, modifier-2]]

This statement causes the line to be redefined based on tangency/ intersection points with the geometry. Valid geometry types are: Point, Line, Circle or Curve. Modifier-1 is used to indicate which portion of the line is to be kept.

The modifier may be XLARGE, XSMALL, YLARGE or YSMALL. Modifier-2 is used to indicate which of two possible intersection points with a circle the geometry is to be redefined to. It may also be XLARGE, XSMALL, YLARGE or YSMALL. The optional [near-pt] is required if geometry is a curve to indicate which of several possible intersections is the correct one to redefine the line to. Modifier-1 and/or modifier-2 are not required if the redefinition can be calculated without them. This is the case with a curve being specified as the geometry. Modifier-2 is not required if the line is tangent to a circle being used as geometry because there is only one intersection point.

C.5.5 REDEF/ circle, geometry [, modifier-1 [, modifier-2]]

This statement causes the circle to be redefined based on tangency/ intersection points with the geometry. Valid geometry types for geometry are: Point, Line or Circle. Modifier-1 is used to indicate which of the two possible intersections the redefinition of the circle is to start at.

The modifier may be XLARGE, XSMALL, YLARGE or YSMALL. If the specified geometry is a point, modifier-1 may or may not be specified.

If the specified geometry is a line that is tangent to the circle being redefined, modifier-1 may be specified but it has no effect since there is only one tangency/intersection point of the circle and the line.

Some modifier-1 must be given if a modifier-2 is given. The modifier-2 is used to indicate which direction the redefined circle is to go in from the starting intersection point until it reaches the existing end point of the current circle. It may be CLW or CCLW. If no modifier-2 is given, it defaults to CLW.

C.5.6 *RESET/ APTCOM

Obsolete syntax: Use [*RESET/ APTSRC, REMARK](#)

C.5.7 *RESET/ CIRAPT

Obsolete syntax. Use [*RESET/ APTSRC, CIRCUL](#)

C.5.8 DISPDB Statement

The DISPDB statement is an obsolete statement that was used to display the contents of a **NCL** data base file without having to first GET the geometry from the data base. The syntax for the DISPDB command is similar to the DISPLAY command.

C.5.9 *RESET/ ECHO

This is an obsolete command for version 501 only. This command has not effect for all the later versions. This construct notifies **NCL** to turn off the ECHO control option. See the *SET/ ECHO command for more information.

C.5.10 *RESET/ LABEL, geometry-type-list

Obsolete syntax. Use the [DRAFT/ LABEL](#) command to control the display of entity labels.

**C.5.11 *RESET/ PLOT, terminal-type
GRAPH**

This is an obsolete command formally used to control the graphic display on older display devices such as Tektronix, and DEC VT terminals.

C.5.12 SCRUB

This is an obsolete command. Use the [FMILL](#) command instead.

The SCRUB statement causes tool motion to be generated over a surface or a portion of a surface, effectively milling an entire region. The tool motion is generated in a back and forth pattern in a direction specified by the boundaries of the region to be scrubbed. If the cutter is ball-end (i.e. the corner radius is equal to half the diameter), then the tool axis must either remain constant ([TLAXIS/ SAME](#)) or be normal to the part surface ([TLAXIS/ NORMAL, PS](#)). If the cutter is flat-nosed (i.e. the corner radius is less than half the diameter), then the only tool axis allowed is normal to part surface. An error will occur if, when using a fixed tool axis, the portion of the surface being machined contains a normal that is greater than 90 degrees from the current tool axis. The valid syntax construct for the SCRUB statement is:

`SCRUB/part-surf,passes,steps[,boundary-points1 thru 4]`

This statement causes the SURF geometry specified by part-surf to be scrubbed.

Passes specifies the number of passes the tool will make.

Steps specifies the maximum number of steps on each pass that will be generated. The actual number of steps on each pass will be reduced as much as is possible while still staying within the current tolerance setting.

If the optional boundary points 1 thru 4 are included, then only the portion of the surface bounded by the 4 boundary points will be scrubbed, otherwise the whole surface will be scrubbed. The points will be surface/cutter contact points and not tool end points. This means the points are projected normal onto the surface and

those projected points are used to set the limits of the SCRUB motion generated. The motion generated by the SCRUB command will start at the first boundary point and go in the direction of the second boundary point. It will then step over for the next pass in the direction of the fourth boundary point and go in the direction of the third boundary point to complete the return pass. If the points are not on the surface, they will be projected onto the surface. If no boundary points are given, the surface will be scrubbed in the direction that it was built.

The SCRUB command is not valid for [QUILT](#) or [NET](#) surfaces.

Example SCRUB Statements:

```
SCRUB/SF1,25,150  
SCRUB/SF1,30,30,PT1,PT2,PT3,PT4  
SCRUB/SF1,30,30,1,THRU,4
```

C.5.13 *SET/APTCOM

Obsolete syntax. Use [*SET/APTSRC, REMARK](#)

C.5.14 *SET/AUTOL1

This is an obsolete command for version 501 only. This construct is used to notify **NCL** to start displaying the current line on the first line of the screen during an interactive **NCL** session. This is the default setting.

C.5.15 *SET/CIRAPT

Obsolete syntax. Use [*SET/APTSRC, CIRCUL](#).

C.5.16 *SET/ECHO

This is an obsolete command for version 501 only. This command have no effect for all the later versions. This construct notifies **NCL** to turn on the ECHO control option.

C.5.17 *SET/ LABEL

This construct turns on the automatic labeling feature. Once set all subsequent geometric entities will be labeled as they are defined.

Obsolete syntax. Use the [DRAFT/LABEL](#) command to control the display of entity labels.

C.5.18 SHAPE Command

The former SHAPE command used in **NCL** Turn programs has been replaced by the NSHAPE command.

For a full description on the use of the [NSHAPE](#) command see chapter 11, Lathe Module.

C.6 IGES Out Label Options

The following label options:

1. From IGES Files
2. From IGES Using Subscripts
3. Using CV Style Labels
4. Using Property Entity
5. From IGES using max 6 chars

from previous versions have been replaced by the following options and flags.

New Label Options:

1. From IGES Label Field
2. From IGES Property

New Flags:

1. Max Label Size
2. Use Subscript
3. Concatenate Subscript

The table on next page shows the equivalent settings for all versions before and after **NCL9.3**.

Before NCL9.3 Label Option	Setting after NCL9.3			
	Label Option	Max Label Size	Use Subscript	Concatenate Subscript
From IGES File	From IGES Label Field	OFF	OFF	OFF
From IGES Using Subscripts	From IGES Label Field	OFF	ON	OFF
Using CV Style Labels	From IGES Label Field	1	OFF	ON
Using Property Entity	From IGES Property	-	ON	OFF
From IGES Using max 6 char	From IGES Label Field	6	OFF	OFF

The following table shows the settings for UG-II style label options which were not supported by previous versions.

	Label Options	Max Label Size	Use Subscript	Concatenate Subscript
UG-II Concatenated	From IGES Label Field	-	OFF	ON
UG-II from property	From IGES Property	-	OFF	OFF

C.7 **NCL** Recover Utility

The **NCL** recovery utility ‘nclrec.exe’ has been obsoleted. See [Chapter 7](#) “Restoring The Current Program In The Event Of An Abnormal Termination Of **NCL**” for detail of how to recover the current part program in the event of abnormal termination of **NCL** interactive session.

C.8 Previous Versions Colors Numbering Scheme

The corresponding colors numbers are not consistent between different commands for all versions before **NCL V9.9**.

See “[Chapter 8: Graphic Display Control Statements - Section 8,1 Colors](#)” for the current colors numbering scheme.

A. Color Number Scheme 1

This color number system is utilized by the following commands.

DRAFT/ANOTE,.....
FXITUR/MODIFY,...
STOCK/MODIFY,...

<u>Color Label</u>	<u>Number</u>
BLACK	0
WHITE	1
BLUE	2
RED	3
GREEN	4
MAGNTA	5
YELLOW	6
CYAN	7
BROWN	8
LTTAN	9
LTBLUE	10
SEAGRN	11
ORANGE	12
PINK	13
PURPLE	14
GREY	15

B. Color Number Scheme 2

This color number system is utilized by the following commands.

DRAFT/DEFALT,geometry_type,.....
DRAFT/MODIFY,.....

<u>Color Label</u>	<u>Number</u>
DEFALT	0
WHITE	1

BLUE	2
RED	3
GREEN	4
MAGNTA	5
YELLOW	6
CYAN	7
BROWN	8
LTTAN	9
LTBLUE	10
SEAGRN	11
ORANGE	12
PINK	13
PURPLE	14
GREY	15

C. Color Number Scheme 3

This color number system is utilized by the following commands.

DRAFT/CUTTER,.....
DRAFT/LABEL,...

<u>Color Label</u>	<u>Number</u>
WHITE	0
YELLOW	1
BLUE	2
RED	3
GREEN	4
MAGNTA	5
CYAN	6
BLACK	7
BROWN	8
LTTAN	9
LTBLUE	10
SEAGRN	11
ORANGE	12
PINK	13
PURPLE	14
GREY	15

C.9 Obsolete *NCL/IPV* Command

This section contains the obsolete ***NCL/IPV*** commands. Use the “PPRINT IPV” commands instead.

**C.9.1 FIXTUR/ BOX, id, 1, point , point [, minz, maxz]
STOCK x1,y1,z1 x2,y2,z2**

This command creates a box shaped stock/fixture through two points in space.

Where:

- id = An integer specifies the ID number of the stock/fixture to be created with this command.
- 1 = A required integer value to denote the box shaped stock/fixture would be created by going through two points in space.
- point or “x,y,z” = Defines the opposite corners of the stock/fixture box to be created.
- minz, maxz = Optional parameters required if the two points defining the opposite corners of the stock/fixture are in the same Z-level. "minz" defined the Z-level value of the bottom corner and "maxz" defined the Z-level value of the top corner of the box to be created.

**C.9.2 FIXTUR/ BOX, id, 2, point, width, length, height
STOCK x,y,z**

This command creates a box shaped stock/fixture centered at a point with a specified width, length and height of the box.

Where:

- id = An integer specifies the ID number of the stock/fixture to be created with this command.

2	= A required integer value to denote the box shaped stock/fixture would be created by specifying the width, the length, the height and one corner.
point or “x,y,z”	= Defines the center point of the box to be created.
width	= A numerical value defines the width of the box along the X-axis.
length	= A numerical value defines the length of the box along the Y-axis.
height	= A numerical value defines the height of the box along the Z-axis.

**C.9.3 FIXTUR/ CONE, id, 2, point_1 , point_2, radius_1, radius_2
STOCK x1,y1,z1 x2,y2,z2**

This command creates a cone shaped stock/fixture through two points with radius specified.

Where:

id	= An integer specifies the ID number of the stock/fixture to be created with this command.
2	= A required integer value to denote the cone shaped stock/fixture would be created by going through two points with each end radius specified.
point or “x,y,z”	= Defines the two opposite ends of the cone shape stock or fixture. This also defines the cone axis.
radius	= A numerical value defines the radius of the cone stock or fixture to be created at each end.

**C.9.4 FIXTUR/ CONE, id, 3, point , vector, height, radius_1, radius_2
STOCK x,y,z i,j,k**

This command creates a cone shaped stock/fixture through a point, along a vector with height and the upper and lower radius specified.

Where:

- id = An integer specifies the ID number of the stock/fixture to be created with this command.
- 3 = A required integer value to denote the cone shaped stock/fixture would be created by going through a point along a vector with height and each end radius specified.
- point or “x,y,z” = Defines the one end of the cone shape stock or fixture.
- vector or “i,j,k” = Defines the cone axis.
- height = A numerical value defines the height of the cone shaped stock/fixture. A negative value means the axis direction will be opposite in direction to the specified “vector”. A positive value means the axis direction will be in the same direction of the specified “vector”.
- radius = A numerical value defines the radius of the cone stock or fixture to be created at each end.

C.9.5 FIXTUR/ CYLNDR, id, 1, circle, length STOCK

This command creates a cylindrical shaped stock/fixture through a circle with a specified height along the axis normal to the specified circle.

Where:

- id = An integer specifies the ID number of the stock/fixture to be created with this command.
- 1 = A required integer value to denote the cylindrical shaped stock/fixture would be created by going through a circle with a specified height along the axis normal to the specified circle.
- circle = A circle which defined one end of the circular shaped stock/fixture to be created.

length = A numerical value defines the height of the circular shaped stock/fixture. A negative value means the extrusion will be along a vector opposite in direction to the vector normal to the specified circle. A positive value means the extrusion will be in the same direction of the vector normal to the specified circle.

**C.9.6 FIXTUR/ CYLNDR, id, 2, point , point , radius
 STOCK x1,y1,z1 x2,y2,z2**

This command creates a cylindrical shaped stock/fixture with its axis going through two points with radius specified.

Where:

id = An integer specifies the ID number of the stock/fixture to be created.

2 = A required integer value to denote the cylindrical shaped stock/fixture would be created with its axis going through two points with radius specified.

point or “x,y,z” = Defines the two opposite ends of the cylindrical shape stock or fixture. This also defines the cylindrical axis.

radius = A numerical value defines the radius of the cylindrical stock or fixture to be created.

**C.9.7 FIXTUR/ CYLNDR, id, 3, point , vector , height, radius
 STOCK x,y,z i,j,k**

This command creates a cylindrical shaped stock/fixture through a point with the axis along a vector, height and radius specified.

Where:

id = An integer specifies the ID number of the stock/fixture to be created.

- 3 = A required integer value to denote the cylindrical shaped stock/fixture would be created with its axis going through two points with radius specified.
- point or “x,y,z” = Defines the end of the cylindrical shape stock or fixture.
- vector or “i,j,k” = Defines the cylindrical axis.
- height = A numerical value defines the height of the circular shaped stock/fixture. A negative value means the extrusion will be along a vector opposite in direction to the specified “vector”. A positive value means the extrusion will be in the same direction of the specified “vector”.
- radius = A numerical value defines the radius of the cylindrical stock or fixture to be created.

C.9.8 FIXTUR/ SOLID, id, solid STOCK

This command loads a **NCL** solid as a fixture/stock.

Where:

- id = An integer specifies the ID number of the stock/fixture to be created with this command.
- 1 = The label of the **NCL** solid.

C.9.9 FIXTUR/ SPHERE, id, 1, circle STOCK

This command creates a spherical shaped stock/fixture with the center point and radius of a circle.

Where:

- id = An integer specifies the ID number of the stock/fixture to be created with this command.

1 = A required integer value to denote the spherical shaped stock/fixture would be created by using the center point and radius of a circle.

circle = A circle which defined the center point and radius of the spherical shaped stock/fixture to be created.

**C.9.10 FIXTUR/ SPHERE, id, 2,point, radius
STOCK x,y,z**

This command creates a spherical shaped stock/fixture with a center point and a radius.

Where:

id = An integer specifies the ID number of the stock/fixture to be created with this command.

2 = A required integer value to denote the spherical shaped stock/fixture would be created with a center point and radius.

point or “x,y,z” = Defines the center of the spherical stock/fixture to be created.

radius = A numerical value defines the radius of the spherical stock or fixture to be created.

**C.9.11 FIXTUR/ TORUS, id, 1, circle_1, circle_2
STOCK**

This command creates a torus shaped stock/fixture using a first circle to define the center, axis, inner/outer radius of the torus and a second circle to define the outer/inner radius.

Where:

id = An integer specifies the ID number of the stock/fixture to be created with this command.

1 = A required integer value to denote the torus shaped stock/fixture would be created with two circles.

circle_1 = A circle which defines the center, axis and inner/outer radius of the torus.

circle_2 = A circle which defines the outer/inner radius of the torus. This circle does not require to be parallel or concentric with the first circle. Only the radius will be utilized. If the radius of this circle is smaller than the radius of the first circle, this circle specifies the inner radius of the torus and the first circle specifies the outer radius of the torus.

C.9.12 FIXTUR/ TORUS, id, 2, circle, radius STOCK

This command creates a torus shaped stock/fixture using a circle to define the center, axis, axial radius and a value to specify the circular radius.

Where:

id = An integer specifies the ID number of the stock/fixture to be created with this command.

2 = A required integer value to denote the torus shaped stock/fixture would be created.

circle = A circle which defined the center, axis and axial radius of the torus.

radius = A numerical value defines the torus circular radius. This value cannot be bigger than the radius of the specified circle.

C.9.13 FIXTUR/ TORUS, id, 3, point, vector, radius_1, radius_2 STOCK x,y,z i,j,k

This command creates a torus shaped stock/fixture using a center point, axis, axial radius, and circular radius.

Where:

id = An integer specifies the ID number of the stock/fixture to be created.

- 3 = A required integer value to denote the torus shaped stock/fixture would be created.
- point or "x,y,z" = Defines the center of the torus shape stock or fixture.
- vector or "i,j,k" = Defines the torus axis.
- radius_1 = A numerical value defines the axial radius of the torus stock/fixture to be created.
- radius_2 = A numerical value defines the circular radius of the torus stock or fixture to be created. This value cannot be larger than "radius_1".

C.9.14 FIXTUR/ CLONE, id1, id2 [, m] STOCK

This command creates "m" copies of the stock/fixture specified by "id2".

Where:

- id1 = An integer specifies the starting ID number of the stock/fixture to be created. If multiple copies are made, then **NCL/IPV** will continue numbering them in ascending order.
- id2 = The ID number of the stock/fixture to be cloned.
- m = An optional parameter to specify how many copies to be made. Only one copy will be made if this parameter is not specified.

C.10 Loading A Stock Or Fixture File

Besides the creation of stock/fixture, the stock/fixture model can be loaded into **NCL/IPV** by an external file in the "stk" or "stl" format.

**C.10.1 FIXTUR/ LOAD, id, [“]file_name[“][, n]
 STOCK**

This command will load an **NCL/IPV** primitives file "file_name.stk". It does not matter which command is used, either STOCK or FIXTUR, as the primitive file contains the Stock and Fixture designations. **NCL/IPV** will first look in the current directory for the file and if it is not found there, it will look in the system directory which is defined by the "NCL_INCDIR" parameter inside the ncl.init file. The stock(s)/fixture(s) loaded will have the ID number specified by "id". If there are multiple stocks/fixtures in the primitive file, then **NCL/IPV** will continue numbering them in ascending order.

“n” is a scalar variable to receive the number of stocks and fixtures defined in the file. When “n” is specified, then it is mandatory that the filename be enclosed in double quotes, otherwise the scalar variable will be assumed to be part of the filename.

The STOCK/LOAD command will return the actual number of stocks and fixtures stored in the external stock file, not just the number of stocks. This is the same for the FIXTUR/LOAD command. So if the external stock file contains both stocks and fixtures, then this number cannot be used to determine the ending stock ID number or the fixture ID number assigned for the loaded file. If you need to know the exact number of stocks or fixtures loaded then you cannot store both stocks and fixtures in the same file.

**C.10.2 FIXTUR/ STL, id, INCHES, [“]file_name[“]
 STOCK MM**

This command will load an STL file with the name "file_name.stl". **NCL/IPV** will first look in the current directory for the file and if it is not found there, it will look in the system directory which is defined by the "NCL_INCDIR" parameter inside the ncl.init file. The stock or fixture loaded will have the ID number specified by "id". If there are multiple Stocks/Fixtures associated with this STL file, then **NCL/IPV** will continue numbering them in ascending order. "INCHES" or "MM" specifies the unit that the STL file is actually stored in.

C.10.3 FIXTUR/ MODIFY, id1 [[[...] [, idn, THRU, idm]] [...]] \$ STOCK

[, COLOR=color] [, VISIBL=ON] \$
 OFF

[, TOLER=tol] [, TRANS=tra] [, ACTIVE=ON]
 OFF

This command modifies the attributes of the specified stock(s)/fixture(s) with the ID denoted by "id1 [[[...] [, idn, THRU, idm]] [...]]" in the command.

Where:

COLOR = Specifies the new color of the stock/fixture.

VISIBL = Specifies the visibility of the stock/fixture. It can be either "ON" or "OFF".

TOLER = Specifies the geometry tolerance.

TRANS = Defines the transparency of the stock/fixture and can be in the range of 1 to 100.

ACTIVE = Specifies whether the stock/fixture will be used in the simulation. It can be either "ON" or "OFF".

C.10.4 FIXTUR/ MOVE, id1 [[[...] [, idn, THRU, idm]] \$ STOCK

[...]], AT, matrix

This command applies the specified matrix to the stock(s)/fixture(s) with the ID denoted by "id1 [[[...] [, idn, THRU, idm]] [...]]" in the command. The matrix will be applied to the original Stock/Fixture and will not be multiplied by any matrix already associated with this Stock/Fixture.

**C.10.5 FIXTUR/ REMOVE, id1 [[[...] [, idn, THRU, idm]] [...]]
STOCK**

This command removes the specified stock(s)/fixture(s) denoted by the ID numbers "id1 [[[...] [, idn, THRU, idm]] [...]]" in the command.

C.10.6 STOCK/REMOVE,CHIPS,pv1[...]

This command removes the excess chips from the stock directly within the part program. A list of point-vectors can be input with this command. These point-vectors will be used to select the portion of the stock to keep. There should be one point-vector per defined stock and each of these should lie within the stock and point towards an edge of the stock to ensure that the correct portion of the stock is retained.

C.10.7 TOOLPN/ x, y, z, i, j, k, u, v, w

This command is used to define the tooling pin of the part for placing the stocks and fixtures onto the machine in “Machine Simulation” mode. This command does not have any effect on the stock and fixtures if “Machine Simulation” mode is not active.

Where:

x, y, z = Origin of the tooling pin in relationship to the part.

i, j, k = Axis of the tooling pin cylinder in relationship to the part.

u, v, w = Vector that points from the center of the tooling pin towards its flat side.

C.10.8 *SET/RAPID,r

This command defines the default rapid feed rate for **NCL/IPV**. Issuing this command is the same as changing the Rapid Rate field in the **NCL/IPV** Tool Modals form, in that any tools added to the tool list after this command has been issued will use this rapid rate.

Enter this command after some tools have already been defined does not necessarily mean that these previous tools will use the old rapid rate, since the tool

list is not actually built until either the Edit Tool List form is displayed or an **NCL/IPV** simulation is started.

APPENDIX D: Summary of *NCL* Commands**CIRCLE Definitions**

- `CIRCLE/x,y[,z],radius`
- `CIRCLE/point,point,point`
- `CIRCLE/pntvec,pntvec`
or `CIRCLE/pntvec ,point`
 `point,vector`
or `CIRCLE/point,pntvec`
 `point,vector`
- `CIRCLE/TANTO,line,XLARGE,point,RADIUS,radius`
 `XSMALL`
 `YLARGE`
 `YSMALL`
- `CIRCLE/XLARGE,line,XLARGE,line,RADIUS,radius`
 `XSMALL XSMALL`
 `YLARGE YLARGE`
 `YSMALL YSMALL`
- `CIRCLE/XLARGE,IN ,circle,IN ,circle,RADIUS,radius`
 `XSMALL OUT OUT`
 `YLARGE`
 `YSMALL`
- `CIRCLE/XLARGE,line,XLARGE,IN ,circle,RADIUS,radius`
 `XSMALL XSMALL OUT`
 `YLARGE YLARGE`
 `YSMALL YSMALL`
- `CIRCLE/XLARGE,line,XLARGE,line,XLARGE,line`
 `XSMALL XSMALL XSMALL`
 `YLARGE YLARGE YLARGE`
 `YSMALL YSMALL YSMALL`

- **CIRCLE/XLARGE,LINE,XLARGE,curve,near-point ,RADIUS,radius**
XSMALL XSMALL pntvec
YLARGE YLARGE
YSMALL YSMALL
- **CIRCLE/IN ,circle,XLARGE,curve,near-point ,RADIUS,radius**
OUT XSMALL pntvec
YLARGE
YSMALL
- **CIRCLE/point ,XLARGE,IN, circle,RADIUS,radius**
pntvec XSMALL OUT
YLARGE
YSMALL
- **CIRCLE/CENTER,point ,TANTO,line**
pntvec
- **CIRCLE/CENTER,point ,RADIUS,radius**
pontvec
- **CIRCLE/CENTER,pntvec,pntvec**
point point
- **CIRCLE/CENTER,point ,LARGE,TANTO,circle**
pntvec SMALL
- **CIRCLE/XLARGE,point ,point ,RADIUS,radius**
XSMALL pntvec pntvec
YLARGE
YSMALL
- **CIRCLE/circle[,delta-x[,delta-y[,delta-z]]]**
- **CIRCLE/OFFSET,OUT,circle,offset-value**
IN
- **CIRCLE/CANON,x,y,z,i,j,k, radius[,i_{Limit-Plane},j_{Lp},k_{Lp},d_{Lp}]**
or CIRCLE/CANON,pntvec ,radius[,plane]
point,vector
- **FILLET/radius,line ,line [...] ,line]**
circle circle circle

CURVE Definitions

- **SPLINE** / [FIT,]pntvec [...], pntvec
CURVE point[,vector] point[,vector]
or SPLINE/[FIT,]point, THRU, point, INCR, m
CURVE
or SPLINE/[FIT,]pntvec, THRU, pntvec, INCR, m
CURVE
or SPLINE/[FIT,] "any combination of above syntax"
CURVE

- **CURVE/CONIC**, point, point, point, point
or CURVE/CONIC, point, THRU, point

- **CURVE/CONIC**, point, point, point, vector
or CURVE/CONIC, point, point, pntvec
or CURVE/CONIC, point, vector, point, point, point
or CURVE/CONIC, pntvec, point, point, point

- **CURVE/CONIC**, point, vector, point, point, vector
or CURVE/CONIC, pntvec, point, pntvec
or CURVE/CONIC, pntvec, pntvec, pntvec
or CURVE/CONIC, pntvec, THRU, pntvec

- **CURVE/COMPOS**, [LINEAR, [CLOSE,]]line [[...], line]
 SMOOTH OPEN circle circle
 CHAMFR curve curve curve
 OMIT spline spline spline

- **SPLINE**/curve, delta-x, delta-y, delta-z
CURVE spline
 line
 circle

- **SPLINE**/INTOF, surface, plane, point
CURVE pntvec
or SPLINE/INTOF, plane, surface, point
CURVE pntvec

- **SPLINE**/INTOF, surface, surface, point
CURVE pntvec
- **SPLINE**/OFFSET, curve, modifier, distance
CURVE
- **SPLINE**/OFFSET, compos_cv, ALL , modifier, dist[, LINEAR]
CURVE comp1 [, comp2] SMOOTH
CHAMFR
- **SPLINE**/PART, offset, surface[[, THRU] [, surface]] [...], AT, zlev
CURVE solid solid plane
n1 n2
LAYER=n
- **SPLINE**/surface[, edge[[, PERCNT] , U-or-V[, num-points]]]
CURVE
- **SPLINE**/OUT, surface[, bn[, en1] [, CLW , en2]]
SSPLIN CCLW
- **SPLINE**/PROJCT, curve, ATANGL, start, [end,] [surf-2,] surf-1[[u,v]]
SSPLIN
- **SPLINE**/PROJCT, curve[, vector] , surface[[u,v]] \$
SSPLIN
 - [, NOWRAP] [, START] [, near-point]
 - WRAP MIDDLE pntvec
 - REVOLV END
 - RADIAL CENTER
 - AT, point
- **SSPLIN**/surface[, n], roff]
- **SSPLIN**/INTOF, surface, surface[, near-point]
plane pntvec
- **SSPLIN**/COMPOS, ssplin[[...] , ssplin]
- **SSPLIN**/OFFSET, sspline, modifier, distance

LINE Definitions

- `LINE/x1,y1,x2,y2`
or `LINE/x1,y1,z1, x2,y2,z2`
- `LINE/point ,point`
`pntvec pntvec`
- `LINE/pntvec`
- `LINE/point ,PARREL,line`
`pntvec pntvec`
- `LINE/point ,PERPTO,line`
`pntvec pntvec`
- `LINE/PARREL,line,XLARGE,distance`
`XSMALL`
`YLARGE`
`YSMALL`
- `LINE/FWD`
- `LINE/point ,RIGHT,TANTO,circle`
`pntvec LEFT`
- `LINE/RIGHT,TANTO,circle,RIGHT,TANTO,circle`
`LEFT LEFT`
- `LINE/XAXIS[,distance]`
`YAXIS`
- `LINE/point ,ATANGL,angle[,XAXIS]`
`pntvec YAXIS`
`line pntvec`
- `LINE/point ,PERPTO,curve,near-point`
`pntvec TANTO pntvec`
- `LINE/INTOF,plane,plane`

- `LINE/XLARGE,circle,ATANGL,angle[,XAXIS]`
 XSMALL YAXIS
 YLARGE line
 YSMALL pntvec
- `LINE/line [,delta-x[,delta-y[,delta-z]]]`
 pntvec
- `LINE/plane`

PATTERN Definitions

- `PATTERN/LINEAR,point,point,number-of-points`
 or `PATTERN/LINEAR,pntvec,pntvec,number-of-pntvecs`
- `PATTERN/LINEAR,point ,vector,number-of-entity`
 `pntvec`
- `PATTERN/LINEAR,point ,vector,INCR,dist1[...]` \$
 `pntvec num1,AT,dist1`

 `[[...],INCR,distn[...]]`
 numn,AT,distn
- `PATTERN/ARC,circle,start-angle,end-angle,CLW ,pt-num`
 CCLW
- `PATTERN/ARC,circle,start-ang,CLW ,INCR,deg1[...]` \$
 `CCLW num1,AT,deg1`

 `[[...] [,INCR],degn[...]]`
 numn,AT,degn
- `PATTERN/PARREL,patern,vector,number-of-set`
 `pntvec`
- `PATTERN/PARREL,patern,vector,INCR,dist1[...]` \$
 `pntvec num1,AT,dist1`

 `[[...] [,INCR],distn[...]]`
 numn,AT,distn

- **PATERN/RANDOM**,point [[,THRU],point] [...]
 point-patern point-patern
or **PN/RANDOM**,pntvec [[,THRU],pntvec] [...]
 pntvec-patern pntvec-patern
- **PATERN/PROJCT**,patern[,vector],surface[[u,v]] \$
 ATANGL,start[,end]

[,NOWRAP] [,START] [,near-point]
WRAP MIDDLE pntvec
REVOLV END
RADIAL CENTER
AT,point

PLANE Definitions

- **PLANE/i,j,k,d**[,display-point]
 pntvec
- **PLANE/point ,point, point**[,display-point]
 pntvec pntvec pntvec
- **PLANE/point ,PARREL,vector,vector**[,display-point]
 pntvec pntvec
or **PLANE/point ,PARREL,pntvec,pntvec**[,display-point]
 pntvec pntvec
- **PLANE/point ,PARREL,plane**[,display-point]
 pntvec pntvec
- **PLANE/PARREL,plane,XLARGE,distance**[,display-point]
 surf XSMALL pntvec
 YLARGE
 YSMALL
 ZLARGE
 ZSMALL
- **PLANE/pntvec**[,display-point]
 pntvec
- **PLANE/point ,PERPTO,vector**[,display-point]
 pntvec pntvec

- **PLANE/point** ,**point** ,**PERPTO**,**plane**[,**display-point**]
 pntvec **pntvec**
- **PLANE/point** ,**PERPTO**,**plane**,**plane**[,**display-point**]
 pntvec **pntvec**
- **PLANE/line**[,**PERPTO**,**plane**] [**display-point**]
 pntvec
- **PLANE/plane**[,**display-point**]
 pntvec
- **PLANE/surface**[,**display-point**]
 pntvec
- **PLANE/FIT**,**curve** [,**scalar-variable**]
 surface

POINT Definitions

- **POINT/x,y[,z]**
- **POINT/pntvec**
- **POINT/TE**
- **POINT/CENTER**,**circle**
 curve
 surface
- **POINT/point** [,**delta-x**[,**delta-y**[,**delta-z**]]]
 pntvec
- **POINT/point** ,**DELTA**,**scaler**,**TIMES**,**vector**
 pntvec
- **POINT/INTOF**,**line** ,**line**
 pntvec **pntvec**
- **POINT/INTOF**,**line**,**plane**
or **POINT/INTOF**,**plane**,**line**

- `POINT/INTOF,pntvec,plane`
- `POINT/INTOF,plane,plane,plane`
- `POINT/INTOF,circle,curve,near-point
 plane pntvec`
- `POINT/INTOF,curve,curve ,ALL
 line near-point
 pntvec pntvec
 circle
 plane`
- `POINT/INTOF3,curve ,curve [,ALL]
 line line near-point
 circle circle near-pntvec`
- `POINT/XLARGE,ENDPT,line
 XSMALL circle
 YLARGE curve
 YSMALL pntvec
 ZLARGE
 ZSMALL`
- `POINT/XLARGE,INTOF,circle,circle
 XSMALL line
 YLARGE pntvec
 YSMALL`
- `POINT/circle,ATANGL,angle`
- `POINT/point ,circle,angle
 pntvec`
- `POINT/line ,distance[,start-point]
 circle
 curve`
- `POINT/ON,circle, per
 line
 curve [PERCNT,]`
- `POINT/ON,surface,[PERCNT,]uper,vper`

- **POINT**/patern, id-number
- **POINT/PROJCT**,point [,vector],surface[u,v] [,near-point]
pntvec pntvec solid pntvec
- **GENPTS**/ [POINT,] [TE ,]num-points-scalar, \$
ARC
DS
PS

reserved-point-array \$
[,reserved-ve-1-array[,reserved-ve-2-array]] \$
[,NOW]
NEXT
- **GENPTS/PNTVEC**[,type][,TE],n,pv-array[,vec-array][,NOW]
ARC NEXT
DS
PS
- **GENPTS/NOMORE**
- **CHKPTS/tol,maxdp,maxang** [,MACRO,m1[,PS]]
DS
- **CHKPTS/NOMORE**
REMOVE
- **PODDEF/dia,hig,ang,PN1[,h1],...[,MACRO,mname]**]
n,ptar,pvar1[,pvar2]
- **PODPTS/[PN1[,h1],]SF1,SFn,...[,MACRO,mname]**]
n,ptar,pvar1,pvar2

POINT-VECTOR Definitions

- **PNTVEC/x,y,[z,],i,j,k**
- **PNTVEC/point,point**
- **PNTVEC/point,vector**

- `PNTVEC/TE, TLAXIS`
- `PNTVEC/TE, FWD`
- `PNTVEC/line`
- `PNTVEC/CENTER, circle`
 `curve [,scalar-variable]`
 `surface`
- `PNTVEC/curve,distance`
- `PNTVEC/curve,point`
or `PNTVEC/point,curve`
- `PNTVEC/point,surface`
- `PNTVEC/PERTO,plane`
- `PNTVEC/pntvec,PERPTO,[ON ,]line [,near-point]`
 `point OFF circle pntvec`
 `curve`
 `plane`
 `surface`
- `PNTVEC/pntvec,TANTO,[ON ,]line [,near-point]`
 `point OFF circle pntvec`
 `curve`
- `PNTVEC/INTOF,plane,plane,plane,POSX`
 `NEGX`
 `POSY`
 `NEGY`
 `POSZ`
 `NEGZ`
- `PNTVEC/pntvec,CROSS,vector`
- `PNTVEC/pntvec,PLUS ,vector`
 `MINUS`
- `PNTVEC/pntvec,TIMES,scalar`

- **PNTVEC**/pntvec,OFFSET,scalar
- **PNTVEC**/UNIT,pntvec
- **PNTVEC**/patern,index
- **PNTVEC**/PROJCT,point [,vector],surface[u,v][,near-point]
 pntvec pntvec
- **PNTVEC**/ON,circle[,PERCNT],per
 curve
 line
- **PNTVEC**/ON,surface[,PERCNT,]uper,vper
- **PNTVEC**/surface

SURFACE Definitions

- **NSURF**/point ,line * Point not allowed for NSURF
SURF line circle
 circle curve
 curve
- or **NSURF**/line ,point * Point not allowed for NSURF
SURF circle line
 curve circle
 curve
- **NSURF**/line ,0 [...],line ,0
SURF circle point circle point
 curve vector curve vector
 line line
 circle circle
 curve curve
 pntvec pntvec
- **SURF**/FILLET,sf,sf,near-point ,radius-curve [,limit-curve]
 pl pntvec st-rad[,end-rad] patern
or **SURF**/FILLET,sf,sf,near-pt,radius-curve [,limit-curve]
 pl pv st-rad[,end-rad] patern

- **NSURF/FIT**,curve[...]curve ,curve
line line line
circle circle circle
or SURF/FIT,curve[...],curve,curve * Spline not allowed
- **SURF/XLARGE**,pl,XLARGE,pl,RADIUS,st-rad[,end-rad,]pt,pt
XSMALL XSMALL
XLARGE YLARGE
YSMALL YSMALL
ZLARGE ZLARGE
ZSMALL ZSMALL
- **SURF/OFFSET**,sf,XLARGE,distance
XSMALL
YLARGE
YSMALL
ZLARGE
ZSMALL
vector
- **SURF/OUT**,trimmed-surface
- **SURF/sf[...][,THRU,sf][,sf]**
- **SURF/REDEF**[,sf],OUT,cv[,IN,cv[[...],cv]]
pl
- **NSURF/REVOLV**,line ,pntvec [,st-angle,end-angle]
SURF circle point,vector
curve
- **NSURF/EGDE**,interp,cv-U0,cv-V1,cv-U1,cv-V0
ci ci ci ci
ln ln ln ln

VECTOR Definitions

- **VECTOR/x,y[,z]**
- **VECTOR/point ,point**
pntvec pntvec

- `VECTOR/pntvec`
- `VECTOR/FWD`
- `VECTOR/TLAXIS`
- `VECTOR/UNIT, vector
pntvec`
- `VECTOR/vector, TIMES, scalar
pntvec`
- `VECTOR/vector, CROSS, vector
pntvec pntvec`
- `VECTOR/vector, PLUS , vector
pntvec MINUS pntvec`
- `VECTOR/PERPTO, plane, POSX
NEGX
POSY
NEGY
POSZ
NEGZ`
- `VECTOR/PARREL, INTOF, plane, plane, POSX
NEGX
POSY
NEGY
POSZ
NEGZ`
- `VECTOR/point , surf
pntvec`
- `VECTOR/TANTO, curve, point
pntvec`
or `VECTOR/TANTO, point , curve
pntvec`

- **VECTOR**/ATANGL,angle,line,POSX
XLARGE
POSY
YLARGE
NEGX
XSMALL
NEY
YSMALL

MATRIX Definitions

- **MATRIX**/Ix,Jx,Kx,Dx, Iy,Jy,Ky,Dy, Iz,Jz,Kz,Dz
- **MATRIX**/TRANSL,Dx,Dy[,Dz]
- **MATRIX**/XYROT,angle
YZROT
ZXROT
- **MATRIX**/SCALE[,point],X-scale[,Y-scale[,Z-scale]]
pntvec
- **MATRIX**/matrix,matrix
- **MATRIX**/INVERS,matrix
- **MATRIX**/MIRROR,plane
- **MATRIX**/point,pntvec,pntvec
vector vector
- **MATRIX**/in-pt1,in-pt2,in-pt3,out-pt1,out-pt2,out-pt3
- **MATRIX**/pntvec,pntvec
vector

Multi-Entities Definitions

- **CURVE**/OUT,composite-curve,ALL [,NUM,val]
component-id-list * THRU cause
allowed

- **SPLINE**/INTOF, ALL , surface[[, THRU] [, surface]] [...] \$
CURVE COMPOS solid solid
n1 n2
LAYER=m

- AT, zlev [, num-cv-scalar]
plane[, XLARGE, offset]
surf XSMALL
YLARGE
YSMALL
ZLARGE
ZSMALL

- **SOLID**/OUT, solid, ALL [, NUM, ncomp]
id-list

- **SURF**/OUT, geo, ALL [, NUM, ncomp]
id-list

DATA Statement

- d1 = **DATA**/element[delimiter element[...]]

- label = **DATA**/file[, num-def[, start[, end]]] [, ERROR] \$
vocab-word
SPACE
SCALAR[, value]
[, VOCABF, ON]
OFF

SYMBOL Definitions

- sym = **SYMBOL**/ [AT,point,]geometry-list

- ins = **PLACE**/[symlib,]sym,AT,pt[,SCALE,s] [,ROTATE,r]

- lab = **DECOMP**/ALL [,nent[,RESET]]
ins1[,ins2,[...],insn] PLUS

SOLID Definitions

- **SOLID/BOX**,point,point[,minz,maxz]
 x,y,z x,y,z
- **SOLID/BOX**,CENTER,point,length,width,height
 x,y,z
- **SOLID/BOX**,BOUND[,xexp[,yexp,zexp]],geo-list
 LAYER=n
 ALL
 MOTION
- **SOLID/COMPOS**[,CLOSE][,INVIS],solid-list
 OPEN RETAIN surf-list
 LAYER=n
 ALL
- **SOLID/CONE**,point,point,RADIUS, radius, radius
 x,y,z x,y,z
- **SOLID/CONE**,CENTER,pntvec ,height,RADIUS, radius, radius
 point, vector
 x,y,z,i,j,k
- **SOLID/CYLNDR**,circle,height
- **SOLID/CYLNDR**,point,point,RADIUS, radius
 x,y,z x,y,z
- **SOLID/CYLNDR**,CENTER,pntvec ,height,RADIUS, radius
 point, vector
 x,y,z,i,j,k
- **SOLID/EXTRUD**,curve,vector,length
 i,j,k
- **SOLID/PART**,expansion,surface-list[,at,zlev,height] \$
 LAYER=n plane, height
 ALL LEVEL[,bofs[,tofs]]

 [,OFFSET,vector] [,PROFIL]
 i,j,k BOX

- **SOLID/REVOLV**,curve,pntvec [,start-angle,end-angle]
point, vector
x,y,z,i,j,k
- **SOLID/SPHERE**,circle
- **SOLID/SPHERE**,CENTER,point,RADIUS, radius
x,y,z
- **SOLID/STL**[, INCHES] [, EDGE, ON], "file-name"
MM OFF
- **SOLID/TORUS**,circle,circle
- **SOLID/TORUS**,circle,RADIUS, radius
- **SOLID/TORUS**,CENTER,pntvec ,RADIUS, radius, radius
point, vector
x,y,z,i,j,k
- **SOLID/LOAD**, "file-name" [,nent] [,TRFORM,matrix]
- **SAVE/SOLID**, STOCK , "file-name", solid-list
FIXTUR LAYER,n
ALL
- **SAVE/STL**[,1][,tol], "file_name", solid-list
2

Annotation Definition

- **ANOTE/text**[,AT,LETTER] [,LETTER, pos2] [,LETTER, pos3] \$
LINE
pos1
[,CURVE, cv1] [,PROJEC[,ve1] ,sf1[,NOWRAP] [,START]]
WARP MIDDLE
REVOLV END
RADIAL CENTER
AT, pt1

ANALYZ & OBTAIN Statements

- **ANALYZ**/sf1[[,THRU],sfn][...] [,sfm[[,THRU],sfmn]]
- **OBTAIN**/ATTRIB,geometry-id,color,layer,lintyp,linwgt
- **OBTAIN**/geometry-id,scalar-id[...]
- **OBTAIN**/data-id,scalar-id[...]
- **OBTAIN**/CONTCT,scalar-id[...]
 - CUTTER
 - FEDRAT
 - HOLDER
 - MAXANG
 - MAXDP
 - NUMPTS
 - PSEUDO
 - SHANK
 - THICK
 - TOLER
 - UNITS
- OBTAIN/COLOR,color_name,r,g,b
- **OBTAIN**/PS,text-variable
 - DS
 - CS
- **OBTAIN**/pokname,scalar,pntvec-array[,UP]
 DOWN
- OBTAIN/x,y,z[,i,j,k]
- OBTAIN/,,i,j,k

Geometry Editing

- **REDEF**/circle
 - curve
 - line

- **REDEF/circle**[,line,line,XLARGE]
 XSMALL
 YLARGE
 YSMALL
- **REDEF/curve-1**[,near-point-1],EDGE,curve-2[,near-point-2]
 XLARGE XLARGE
 XSMALL XSMALL
 YLARGE YLARGE
 YSMALL YSMALL
- **REDEF/curve-1**[,curve-name][,near-point-1],EDGE,curve-2 \$
 CENTER
 XLARGE
 XSMALL
 YLARGE
 YSMALL

[,near-point-2],EDGE,curve-3[,near-point-3]
 XLARGE XLARGE
 XSMALL XSMALL
 YLARGE YLARGE
 YSMALL YSMALL
- **REDEF/curve**,CLOSE
 OPEN
- **REDEF/sf**, [PARAMS, [SWAP,] [REVERS, 0 [,1 [,NORMAL]]]]] \$
 1 NORMAL
 NORMAL

[[,] PARLEM, 0] [[,] BASE]
 1 FACE
 -1
- **REDEF/surface**,CLOSE,0
 OPEN 1
- [sfn=] **REDEF/[p11,]OUT,cv1[,IN,cv2,...]**
 sf1
- [sfn=] **REDEF/sf1[,OUT,cv1][,modifier][,IN,cv2,...]**
 sf2
 p11

- [sfn=] **REDEF**/surface
- [sfn=] **REDEF**/sf1, REMOVE, i1 [,i2, ...]
[0,]ALL

Geometry Control

- **CANON**/ON
OFF
- **new-name**=GEOMETRY-TYPE/old-name
- **DEFNAM**/geo-type1, name1 [, INDEX] [, geo-type2, name2 [, INDEX]] [...]
- **CLONE**/geometry, matrix
- **MOVE**/geometry, matrix
- **RENAME**/old-name-1, new-name-1 [, old-name-2, new-name-2] [...]
- **MODSYS**/matrix
NOMORE
- **REFSYS**/matrix
NOMORE
- **LOADU**/filename [, MM]
INCLES
SAME
UNITS
- **SAVEU**/filename
- **UBFN**/filename
- **UBFN**/CLOSE
- **GET**/ent-id [, THRU, ent-id] [ent-id [, THRU, ent-id]] [...]
PUT
- **GET**/alphanumeric-character-string
PUT

- `GET/geometry-id,AS,new-id`
`PUT`
- `GET/entity-type,...`
`PUT`
- `GET/ALL [,OFFSET,n]`
`PUT`
- `GET/RENAME,....`
- `GET/ [RENAME,] LAYER,n1[[,THRU],nn] [. . .] [,m[,THRU],mn] [,OFFSET,k]`
- `PUT/LAYER,n1[[,THRU],nn] [. . .] [,m[,THRU],mn]`
- `REMOVE/geo-list`
 `type-list`
 `ALL`
 `scalar-id`
 `"text-string-id"`
- `ZSURF/scalar`
 `plane`
 `NOMORE`
- `REVERS/data-id`
 `geometry-id[,POSX]`
 `NEGX`
 `POSY`
 `NEGY`
 `POSZ`
 `NEGZ`

PRINT Control

- `FORMAT/SHORT`
`LONG`
- `PRINT/0`
- `PRINT/ALL`
 `geometry-type`
 `SCALAR`

- PRINT/[LARGE](#)
[SMALL](#)
- PRINT/[OFF](#), IN
ON
- [REMARK](#) text-string
- PPRINT/@var-name, . . . , @var-name
message message

Text Control Commands

- tx1="*text string here*"
- tx2=[text-element](#) [& [text-element](#) [...]]
[format-function](#) [format-function](#)
- tx3=tx1[n:[m]]
- FORMAT(%[-][+][0][w][.][d]f , scalar-var)
%[-][w]s , text-var
- [LNTH](#)(text-element)
- [FINDEX](#)(text-element-1, text-element-2)
- [RINDEX](#)(text-element-1, text-element-2)
- [STRCMP](#)(text-element-1, text-element-2)
- {text-element}
- [TEXTF](#)(label)

UNITS

- UNITS/INCHES
MM

CUTTER Definitions

- `CUTTER/diameter[,corner-radius[,height[,side-angle]]]`
- `CUTTER/diameter,corner-radius,height,side-radius,Z-height $ [,flat-angle]`
- `CUTTER/d,r,e,f,a,b,h`
- `CUTTER/BLADE,width,chisel,height,angle`
- `CUTTER/LATHE, radius, diameter, height[,angle[,mount-angle]]`
- `CUTTER/LATHE, radius, width, height, 0, length`
- `CUTTER/POFIL,"lib_name"`
- `CUTTER/PSEUDO,diameter[,corner-radius[,height[,side-angle]]]`
- `CUTTER/PSEUDO,diameter,corner-radius,height,side-radius, Z-height[,flat-angle] $`
- `CUTTER/TOOL,[lib,]tool[,params]`
- `CUTTER/READ,[lib,]tool[,params]`

CUTTER Display Commands

- `CUTTER/DISPLAY,diameter[,corner-radius[,height[,side-angle]]]`
- `CUTTER/DISPLAY,diameter,corner-radius,height,side-radius, Z-height[,flat-angle] $`
- `CUTTER/DISPLAY,surface
cv[,pv]
pt-list
solid
[symlib,]symbol`

- **CUTTER/DISPLY**, curve
pt-list
solid
[symlib,] symbol
- **CUTTER/DISPLY, HOLDER**, diameter, height[, side-angle][,ofs]
- **CUTTER/DISPLY, HOLDER**, width, length, depth, y-offset
- **CUTTER/DISPLY, HOLDER**, surface [,ofs]
cv[,pv]
pt-list
solid
[symlib,] symbol
- **CUTTER/DISPLY, HOLDER**, curve [,x,y] [,OFFSET,zatt,zdep]
pt-list
solid
[symlib,] symbol
- **CUTTER/DISPLY, SHANK**, dia, height[, side-angle][,ofs][,CUTTER]
HOLDER
- **CUTTER/DISPLY, SHANK**, surface [,ofs][,CUTTER]
cv[,pv]
HOLDER
pt-list
solid
[symlib,] symbol
- **CUTTER/DISPLY, SHANK**, width, length, depth, y-offset [,CUTTER]
HOLDER
- **CUTTER/DISPLY, SHANK**, curve [,x,y] \$
pt-list
solid
[symlib,] symbol
[OFFSET,zatt,zdep][,CUTTER]
HOLDER
- **CUTTER/DISPLY**, PART
ALL

- CUTTER/DISPLAY,MOVE,ON
OFF
- CUTTER/DISPLAY,SHADE,ON [,CUTTER]
OFF SHANK
HOLDER

MULTAX Commands

- MULTAX [/ON]
OFF

Tool Axis Control Commands

- TLAXIS/*i,j,k* [,NORMAL] [,modify-clause]
vector
pntvec
- TLAXIS/1
SAME [,NORMAL] [,modify-clause]
- TLAXIS/NORMAL,PS[,surface] [,PERPTO,[LAST,]vector] \$
NORMPS plane pntvec
[,modify-clause]
- TLAXIS/ATANGL,angle,PS[,surface] [,CLDIST,dist] [,CONTCT] \$
plane
[,PERPTO,[LAST,]vector] [,modify-clause]
pntvec
- TLAXIS/TANTO,DS,height[,PS,surface] [,PERPTO,[LAST,]vector] \$
plane pntvec
[,modify-clause]
- TLAXIS/TANTO,DS,height,FAN[,CENTER,OFF] [,modify-clause]
ON
AUTO
- TLAXIS/TANTO,DS,height,PARELM[,modify-clause]

- **TLAXIS/COMBIN, height**, [CENTER, OFF ,] [PS, surface,]
ON plane
AUTO

leave-dist [, SMOOTH[, d, r]] [, approach-dis]
[, modify-clause]
- **TLAXIS/COMBIN, height**, [CENTER, OFF ,] PARELM, leave-dis
ON
AUTO

, SMOOTH[d, r]] [, approach-dis] [, modify-clause]
- **TLAXIS/THRU, point**
- **TLAXIS/THRU, curve[, dist]**
- **TLAXIS/INTERP, vector**[, SMOOTH[, d, r]]
pntvec
i, j, k
- **TLAXIS/ [. . . ,]RIGHT**, right-angle, FWD, fwd-angle
- **TLAXIS/ [. . . ,]GUIDE**, curve[, CONCTCT] [, TLLFT] [, thick]
OFFSET TLRGT
TLON
- **TLAXIS/ [. . . ,]GOUGCK**, ON
OFF
- **TLAXIS/ [. . . ,]LOCK**, OFF
END
ds1[, LINEAR] [, ds2[, INTERP]] [, OMIT]
RADIUS FAN RETAIN

Tool Condition and Offset Statements

- **THICK/OFF**
- **THICK/ps-thick[, ds-thick[, cs1-thick[, cs2-thick
[cs3-thick[, cs4-thick[, cs5-thick]]]]]]**

- **TLLFT**
TLRGT
TLON
- **TLONPS**
TLOFPS

FEDRAT Commands

- **FEDRAT**/feedrate[,IPM]
 IPR
- **FEDRAT**/ [AT,dist-1[,SCALE],fs1[,nfed][,ONCE]] \$
 [[,]OUT[,dist-2[,SCALE],fs2[,nfed][,ONCE]]] \$
 [,LENGTH,dis]
- **RAPID**[/Optional-Parameters}

Directional Indicators

- **INDIRP**/point
 x,y,z
- **INDIRV**/vector
 pntvec
 i,j,k
- **SRFVCT**/ [ds-vector] [,cs-vector]
 i1,j1,k1 i2,j2,k2

Positioning Commands

- **FROM**/pntvec [, feedrate]
 point [,tool-axis-vector]
 x,y,z
 x,y

- **GO/** [TO ,] **drive-surface** [[,ON], **part-surface** [[,TO],
ON
NOPS
PAST
ON
PAST
[**check-surface**]]]
- **GOTO/pntvec** [, feedrate]
point [, tool-axis-vector]
x,y,z
x,y
- **GOTO/pattern** [, INVERS] [, CONST] [, AVOID] [, RETAIN] [, OMIT]
- **GODLTA/delta-x,delta-y,delta-z** [, feedrate]
vector
distance
plane
surface
pntvec

Part Surface Control

- **AUTOPS**
- **NOPS**
- **PSIS/surface**
plane

Motion Commands

- **ARCSLP/FILLET**, radius[,tol][,WARN] [,ONCE]
NOWARN COMBIN
[,INTERP] [,FEDRAT,fdrt,fmax,LEFT ,cdia]
LOCK [,maxang] RIGHT
- **ARCSLP/FILLET**, **SAVE**
RESTOR
- **CONTCT/ON**
OFF

- **CUT**
- **DNTCUT** [/NOMORE]
- **GOLFT** /ds, [TO ,]cs[,NEARPT,point]
GORGT ON
GOFWD PAST
GOBACK PSTAN
GOUP TANTO
GODOWN
- **GOLFT** /ds, [TO ,]n,INTOF,cs
GORGT ON
GOFWD PAST
GOBACK PSTAN
GOUP TANTO
GODOWN
- **GOLFT** /ds, [TO ,]cs-1,lab-1,cs-2,lab-2 \$
GORGT ON
GOFWD PAST
GOBACK PSTAN
GOUP TANTO
GODOWN
- **GOFWDA**/ [LOOK,n,] [AVOID,] cmprcv
FIND
- **GOFWDA**/ds1, \$
[LOOK,n,] [AVOID,] [modifier,] [n,INTOF,] ds2 \$
FIND
[,NEARPT,pt][[,LOOK,n][,AVOID][,modifier]\$
FIND
[,n,intof],dsi[,NEARPT,pt]][...] \$
[,CHECK,[AVOID,][modifier,][n,INTOF,]\$
FIND
cs1[,NEARPT,pt][[,AVOID][,modifier]\$
[,n,INTOF],csi[,NEARPT,pt]][...]] \$

Simulated CYCLE Commands

- CYCLE/ON
OFF
- RETRCT/ON
OFF
- CYCLE/mode,FEDTO,depth [,IPM ,fd[,rap]] [,RAPTO,r] \$
dia,ang IPR
MMPM
MMPR
TPI
[,DWELL,dw11[,dw12]][STEP,peck1[,peck2]] \$
[,OFFSET,off1[,off2]][RTRCTO,ret[,plng]] \$
[,REPEAT,rep][,START[,seq[,inc[,frq]]]] \$
[,PARAMS,n1,m2...,nk,mk]

POINT Pocket Command

- POCKET/radius,dist,fin-cut,plng-fr,gen-fr,fin-fr,
cov,type,pt-list \$

Advanced Pocket Commands

- POCKET/bot-pl,[SAME ,]top-pl,[dir-mod,]perim \$
bot-sf NORMAL[,PS,surf] dist
[,dir-mod][,island,...][,START,start-point]
END,end-pt
scalar
[,max-loops][,PS,THICK,ptthk][,DS,THICK,dth1[,dth2]] \$
[,FINISH,THICK,thk][,OFF,PART[,dis]] \$
[,OPEN,ind1[,THUR,ind2]]
ALL

- **POKMOD**/num-ramp, RAMP , [CLW ,] [PERPTO,] \$
 num-revs PLUNGE CCLW TANTO
 ent-ang HELIX
 COUPLE
 CYCLE
 OMIT
 text-variable
- [WARN ,] ent-dist, [OUT, ARC,] [ISLAND, ARC,] \$
 NOWARN
 AVOID[, NOWARN]
- RETRCT, ON , clr-lev, [INCR,] stp-dwn-dist, [DIST ,] \$
 OFF DEPTH
 rapto-dist, [hretdis, [vertdis,]
- [COLAPS,] CCLW, OUT, \$
 CLW IN
 LACE , POSX , [FINISH, SAME ,] [BOTH, [SHORT ,]
 SCRUB NEGX CCLW SAME
 POSY CLW CCLW
 NEGY OMIT CLW
 LONGC REVERS REVERS
 SHORT
 vector
- UP , SHARP, [IN, rad,] [ATANGL, ANG,] [TRANS, ARC,] \$
 DOWN ARC
- maxstp-dist, minstp-dist, [HEIGHT, scallop,] \$
 0 0
- gen-fr, pstn-fr, ret-fr, ent-fr, trans-fr, fin-fr \$
 0 RAPID RAPID 0 0 0
- [, first-pass-fr] \$
 0
- [, CUTCOM, [LEFT ,] [XYPLAN,] arg1, . . . , argn [, NOMORE]]
 RIGHT YZPLAN
 ON ZXPLAN

- **POCKET/LAYER=n**, [POCKET,] [PLANE ,] STOCK, [offset,] \$
 VMPOCK REVOLV, surf plane1, [offset,]
 VMP3AX DEPTH, dep,
 AT, z-level,

- STOCK [, offset] \$
 plane2 [, offset]
 OFFSET, dist
 AT, z-level

- [, FINISH, LEVEL [, ADJUST, UP, dup, DOWN, ddn]] \$
 [, OMIT [, AVOID, sflist2]
 IN
 PART [, ex [amd]
 FIT [, expand]
 BOUND, sflist1 [, expand]

- [, ERROR, max-gap] [, START, RANDOM] \$
 TOLER NEGX
 POSX
 NEGY
 POSY
 NEGZ
 POSZ
 IN
 OUT
 SMALL
 LARGE
 NEARPT, point

- [, max-loops] [, ZONE, cv-list, [STOP,] ALL] \$
 LAST MAIN
 PART

- [, LEVEL] [, OFF, PART [, offset]] [, STEP, UP, stp]
 DEEP

- **POCKET/pokname**
 pokname=POCKET/....

VoluMill Pocket Command

- **VMPMOD**/ [RAMP ,ang] [,CLW] [,CLRSRF,cpln] [,RAPTO,rapd] \$
HELIIX,ang,rad CCLW cdis\$
[,RTRCTO,ret1,ret2][,DIST ,dep1][,STEP,stp1] \$
DEPTH
[RADIUS,rad1][,ROUGH] [,SIDE] \$
SMOOTH,sang,srad SLOT,sdep,sstp
[,FLUTES,n[,flen[,tlen]]][,LEVEL] \$
DEEP
[,FEDRAT,f1,f2,f3,f4[,IN]][,SPINDL,s1,s2,s3] \$
OUT
[,DWELL,d][,SMALL,ON][MINFED,fedmin] \$
- **VMPOCK**/bottom-plane-or-surface,top_plane,[OFFSET,
dis IN
peri_geo[,islands][,PS,THICK,pthk] \$
[,DS,THICK,dthk[,ofsthk]][,OPEN,ind1[,THRU,ind2]]
ALL

RMILL Command

- **RMILL**/ps,[cs1-mod,]cs1,[cs2-mod,]cs2,[ds1-mod,]ds1,\$
[ds2-mod,]ds2,motn-typ,[clr-pl,plng-dist,]
dist1
step-over-type,stepover-dist,gen-fr,\$
pstn-fr,plnr-fr[,ret-pl]\$
ret-pt
dist2
[,ROUGH,rcsth1[,rcsth2[,rdsth1[,rdsth2]]]]\$
[,FINISH,fcsth1[,fcsth2[,fdsth1[,fdsth2]]]]

FMILL Command

- **FMILL**/sf1,HEIGHT,scallop-height
 PASS,num-passes

 [,CS,cs1[,th1][,cs2[,th2]][...][,cs10[,th1-]]]]
 [,START,pt1][,FWD,0][,TOLER,tol][,OMIT,OFF]
 u,v 1 STEP,stp START
 END
 BOTH

 [,ON,curve][,RAPTO,p11[,f1]]
 sf1 RAPID
 ds1
 0

 [,RETRCT,p12[,f2]][,CONTCT]
 sf2 RAPID TO
 ds2 ON
 0 PAST

 [,AVOID,OFF
 THRU
 DOWN,[di,]
 [BOTH]p13[,FEDRAT,fr3],p14[,FEDRAT,f4]
 sf3 RAPID sf4 RAPID
 ds3 ds4

 [,STOP]
 LAST[,FEDRAT,f5]

SMILL Command

- **SMILL/sf-list [,TO]ds1,ds2 [, BOUND, PART]** \$
PAST vel cv1
ON
CONTCT

[, PASS, num-passes] [, START, pt1] \$
[HEIGHT, sclop-hgt] [, STEP, stp]

[, SCRUB] [, TOLER, tol] \$
COMBIN
LACE [, ent1[, fr1]]
dis1 RAPID

[RAPTO,ent2[,fr2]][,RETRCT,ent3[,fr3]]
dis2 RAPID dis3 RAPID \$

[, OMIT, IN]
NOMORE

PROFIL Command

- **PROFIL/anote[,PS,depth] [,CLRSRF,plane1]** \$
[DIST,]dis1
INCR

[,RAPTO,plane2]
[DIST,]dis2
INCR

[,RETRCT,plane3] [,RAPTO,dis4] \$
ON
OFF
[DIST,]dis3
INCR

[STEP,top-plane,max-step]
dis5 PASS,n
PASS,n[,max-step]

[FEDRAT,gen-fr, pos-fr, ret-fr, ent-fr]

- **PROFIL**/ [ON ,]curve [,th1][,AUTO][,PAST,both] \$
 OFF circle LEFT start,end
 line RIGHT
- [,START[,END] [,CCLW] \$
 ENDPT CLW
 NEARPT,pt1 ve1
 pv1 pv2
- [, OMIT]] \$
 [ATANGL,ds1,ang1[,h2]] [,ARC,r1[,h1]]
- [CS,ENDPT] [,PS[,pl1][,th2][,SAME]] \$
 NEARPT,pt2 sf1 NORMAL[,pl2]
 pv3 sf2
- [,CLRSRF,pl3] [,RAPTO,p14] \$
 [DIST,]ds2 [DIST,]ds3
 INCR INCR
- [,RETRCT,p15 \$
 ON
 OFF
 [DIST,]ds5
 INCR
- [, OMIT]] \$
 [ATANGL,ds4,ang2[,h4]] [,ARC,r2[,h3]]
- [,RTRCTO,ds6] [,OFFSET,ds7,stp1] [,DOWN]] \$
 PASS,n1 UP
 PASS,n1,stp1 OFF
- [,STEP,top-plane,stp2] [,LEVEL]] \$
 ds8 PASS,n2 DEEP
 PASS,n2,stp2
- [,FEDRAT,gen-fr, pos-fr,ret-fr,ent-fr,tran-fr, \$
 exit-fr] [,FILLET,ang3]
- [CUTCOM,arg1,...,argn[,OFFSET,dis] [,ALL] \$
 START
- [,NOMORE]] [MAXDP,maxstp] \$
 OFF

Lathe Commands

- **LATHE/FINISH**, shp2, STOCK, stk-x, stk-y, ENGAGE, ang, lngth, \$
RETURN, ret-pt
- **LATHE/ROUGH**, shp1, CLDIST, dist, shp2, STOCK, stk-x, stk-y, \$
dp-per-pass, CUTANG, ang, RETRCT, ang1, ret-dist, \$
RETURN, ret-pt
- **NSHAPE**/line ,... ,line
circle circle
curve curve

Tolerance Specifications

- **GOUGCK**/OFF
ON[,1]
2
3
- **GOUGCK**/PS, 0, DS, 0, CS, 0
1 1 1
2 2 2
3 3 3
4 4
- **MAXDP**/maxstp[, STEP, ON] [, AUTO] [, max-loops] [, min-dp] \$
OFF OFF
[, NOWARN] [, ONCE]
WARN
- **MAXANG**/ang [, ONCE]
- **NUMPTS**/n [, ONCE]
- **TOLER**/chord-tol[, positn-tol]

MACRO Statements

- CALL/macro-name[,arg-1[=input-1][,...[arg-n[=input-n]]]]
 - macname=MACRO/arg-1[=default-1][,...[arg-n[=default-n]]]
 - ...
 - ...
 - TERMAC
 - PROMPT/macro-name,[class,][OUT ,][DEFALT ,]"description-text"
 OMIT RETAIN
 - PROMPT/arg-name,[lngh,prcsn,]"prompt-string"
 [,SUBSTR,RETAIN][,NOW][,min,max]
 LABEL NEXT word-list
 NUM geo-list

Program Control Commands

- **LOOPST**
....
....
LOOPND
 - **DO**/lab-name, idx-name=strt-value,max-value[,INCR,incr-val]
....
....
lab-name: [Last-do-loop-statement]
 CONTIN
 - **lab-name1**: [Regular-statement/**CONTIN**]
....
....
JUMPTO/lab-name1
or **JUMPTO**/lab-name2
....
....
lab-name2: [Regular-statement/**CONTIN**]
 - **ON**/**ERROR**, THEN, **CONTIN**, **WARN**
 STOP NOWARN
 label

- UNDO
- IF (expression) label-1,label-2,label-3
- IF (logical-expression) label
 - [,]Regular-statement
- IFTOL/toler
- IF (logical-expression) THEN
 - Regular-statement(s)
 - [[ELSEIF (logical-expression) THEN
 - Regular-statement(s)
 - [...]
 - ELSE
 - Regular-statement(s)]
- ENDIF

CLFILE Operations Commands

- TRACUT/matrix-id
 - NOMORE
- TRACUT/LAST,matrix-id
 - NOMORE
- COPY/index-n[, SAME [, num-times]]
- COPY/index-n,MODIFY,matrix[,num-times]
- COPY/index-n,TRANSL,x,y,z[,num-times]
- COPY/index-n,XYROT,ang[,num-times]
 - YZROT
 - ZXROT
- INDEX/index-n
 -
 -
 - INDEX/index-n,NOMORE
- REVERS/ON
 - OFF

DRAFT Commands

- **DRAFT/CUTTER** [,STEP=ns] [,COLOR=mc,cc] [,SHANK,COLOR=sc] \$
[,HOLDER [,COLOR=hc] [,LINTYP=mline-type]] \$
[,RAPID [,COLOR=rc] [,LINTYP=rline-type]] \$
[,TRACUT=ON] [,TRANS=n]
OFF
- **DRAFT/DEFALT**,geometry-type[,subtype,n],COLOR=color
- **DRAFT/DEFALT**,MAXIS=color
WAXIS
- **DRAFT/FORMAT**=SINGLE,NAME=name-list [,BORDER=ON] [,AXIS=ON] \$
OFF OFF
[,NAMED=ON] [,SIZE=ON] [,MOTION=ON] [,MODE=WIRE]
OFF OFF OFF BOTH
SHADE
HIDDEN
- **DRAFT/FIT**,ALL [,SAME] [,OMIT=omit-list]
VIEW=name-list
ALL
- **DRAFT/LABEL**,ON [,geometry-type-list] [,LEADER,ON]
OFF OFF
- **DRAFT/LABEL**,ON ,MODIFY=geometry-list-or-type[,LEADER,ON]
OFF OFF
- **DRAFT/LABEL**[,ON] [,BOX,ON] [,COLOR,txt,box] [,SIZE,wd,ht] \$
OFF OFF
[,OFFSET,dis] [,LEADER [,ON] [,COLOR,led] [,ARROW,ON]]
OFF OFF
- **DRAFT/LABEL**,ALTER,geo[, [POINT ,]point [,LEADER,ON [,point]]]
VECTOR vector OFF
- **DRAFT/NAME**=name-list,RESET

- **DRAFT/REDRAW**, ALL
VIEW=name
n
- **DRAFT/NAME=drw-view** [, BORDER=ON] [, AXIS=ON] \$
Front OFF OFF
Back
Top
Bottom
Left
Right
Isometric
Left Iso
Dimetric
Left Dimetric

[NAMED=ON] [, SIZE=ON] [, MOTION=ON] [, MODE=WIRE]
OFF OFF OFF BOTH
OFF OFF OFF SHADE
OFF OFF OFF HIDDEN
- **DRAFT/ANOTE** [, MODIFY=anote-list] [, COLOR=color] \$
ALL

[, SIZE=hgt, exp] [, ATANGL=ang] [, FONT="name"] \$

[, STYLE=STROKE] [, START=pos] [, LETDIR=LEFT] \$
STRING RIGHT
UP
DOWN

[, SPACE=hor, ver] [, LINWGT=dens] [, LAYER=lay] [PEN=pn]
- **DRAFT/VIEW=name**, TRACUT
REFSYS
MAXTRIX=mx
TOOL
PARAMS [, CENTER=point] [, NORMAL=vector] \$
[, YAXIS=vector] [, SCALE=n]

- **DRAFT/MODIFY**, [=geo-list] [, COLOR=color] [, PEN=pn] \$
geo-type-list
ALL
[, LINTYP=SOLID(1)] [, LAYER=ln] [, LINWGT=STD(1)] \$
DASH(2) MEDIUM(2)
DOTTED(3) HEAVY(3)
CENTER(4) EXHVY(4)
PHANTM(5)
DASHLN(6)
DASHDT(7)
DASHSP [(8)
[, SHADE=ON] [, TRANS=(1 THRU 100)] [, MATERL=(1-16)] \$
OFF mat-name
[, EDGE=OFF] [, MARKER=DOT(1)]
color PLUS(2)
STAR(3)
CIRCLE(4)
CROSS(5)
TRIAN(6)
DIMOND(7)
SQUARE(8)
DBLCIR(9)
LRGDOT(10)
CUBE(11)

DISPLAY and ERASE Commands

- **DISPLAY/ALL**
VISIBL
- **DISPLAY/curve** [, num-pts]
VISIBL
- **DISPLAY/CUTTER**
VISIBL
- **DISPLAY/geo-list**
VISIBL geo-type-list
ALL

- **DISPLAY/LAYER**,n[, . . . [, THRU,nn]] [,k1[, . . . [,kn]]]
VISIBL
- **DISPLAY/MAXIS**
VISIBL WAXIS
- **DISPLAY/mx-name**[,ax,boxx[boxy,boxz]]
VISIBL
- **DISPLAY/plane**,AT,point
VISIBL
- **DISPLAY/SURF**[,u-pts,u-lns,v-pts,v-lns]
VISIBL
- **DISPLAY/SURF**[,u-lns,v-lns]
VISIBL
- **ERASE/geo-list**
INVIS geo-type-list
- **ERASE/ALL**
INVIS
- **ERASE/LAYER**,n[, . . . [, THRU,mm]] [,k1[, . . . [,kn]]]
INVIS
- **ERASE/MAXIS**
INVIS WAXIS
- **ERASE/MOTION**
INVIS

NCL Control Commands

- *CONSOL
- *DELETE/ln-num[,ln-num]
- *EDIT[/ln-num]
- *EDT

- *FIND/string
 - *FINDTK/token
 - *INPUT
 - *INSERT
 - *LOADPP/file-name
 - *PAUSE
 - *RESET/ADISPL
 - *RESET/APTSRC,CIRCUL
DATA
IPV
IPVCOM
VERIFY
REMARK[,ACTIVE]
 - *RESET/AUTOST[,tol,angtol][,RETAIN]
*SET OMIT
 - *RESET/AUTOUV
*SET
 - *RESET/CANON
*SET
 - *RESET/CASE
*SET
 - *RESET/CALL[,n]
 - *RESET/DISPLY[,geo-type-list]
*SET
 - *RESET/EXPCL
*SET
 - *RESET/INDENT

- *RESET/MOTION
*SET
- *RESET/NOWARN
*SET
- *RESET/PAUSE
*SET
- *RESET/RUNCMD
*SET
- *RESET/SCHECK
*SET
- *RESET/STATLN
*SET
- *RESET/STOP
*SET
- *RESET/STPCMD
*SET
- *RUN [/ln-num]
MACRO
TERMAC
LOOPST
LOOPND
- *SAVEPP
- *SET/ADISPL, [num-of-points,]num-u-lines,num-v-lines
- *SET/ADISPL, [num-of-points,]num-u-pts, \$
num-u-lines, num-v-pts, num-v-lines
- *SET/ADISPL,TOLER,scalar
- *SET/APTSRC,CIRCUL[,IPV]
VERIFY

- *SET/APTSRC,CUTTER,APT
NCL
PPRINT
scalar
- *SET/APTSRC,DATA
IPV
IPVCOM
LOW
HIGH
REAL
VERIFY
REMARK[,ACTIVE]
TRACUT[,matrix]
- *SET/CMDLEN,n
- *SET/CMDCOM,n
- *SET/ELIMIT,scalar
- *SET/geo-type,scalar
- *SET/INDENT,ALL,scalar
SEP
OFF
- *SET/TRIMMED,DEFALT
BASE
FACE
- *SET/VER,vflag
- *SET/WLIMIT,scalar
- *SHOW/ADISPL
- *SHOW/CUTTER
- *SHOW/FEDRAT
- *SHOW/FILES

- *SHOW/identifier
- *SHOW/MAXANG
- *SHOW/MAXDP
- *SHOW/MODALS
- *SHOW/MODSYS
- *SHOW/NUMPTS
- *SHOW/REFSYS
- *SHOW/identifier
- *SHOW/SOURCE[, scalar]
 S
- *SHOW/STATLN
- *SHOW/THICK
- *SHOW/TLAXIS
- *SHOW/TOLER
- *SHOW/TOOL
- *SHOW/TRACUT
- *SHOW/UNITS
- *SHOW/vocab-word
- *SKIP
- *SKIP/num-of-lines (+ or -)
- *SKIP/TO, line-num
 END
- *STOP

- *SYSTEM[/system_command[,OFF]]
 ON
- *TIME
- *VER
- *WINDOW[/OPEN]
 CLOSE
- **any of the above commands

Miscellaneous Commands

- FINI
- INCLUD/file-name[,ON]
- INSERTtext-strings
 @var-name
- PROMPT/name,[defvalue ,,"prompt-str",[,min,max]
 "defstring" geotype
- PROMPT/SCALAR,scalar[, "Class-string"], "description-string"
- READ/description,file-name
- REMARK/ON
 OFF
- RESERV/identifier-1,size-1[,...[,identifier-n,size-n]]
- SEQUNC/label
 END
- SET/MODE,LINEAR
 CIRCUL
- SYN/new-word,old-word[,...[,new-word,old-word]]
- TITLEStext-strings

- **TRALST**/ON
OFF

NCL/IPV PPRINT IPV Commands

- **PPRINT IPV CUTTER** parameters
- **PPRINT IPV CUTTER BLADE** parameters
- **PPRINT IPV CUTTER LATHE** parameters
- **PPRINT IPV CUTTER DISPLAY** "pt-list"
- **PPRINT IPV DNTCUT**
- **PPRINT IPV FIXTUR BOX** id x1,y1,z1,x2,y2,z2
STOCK
- **PRINT IPV FIXTUR CLONE** id idn,ncopies[,~]
STOCK

PPRINT AT,x1,y1,z1,d1,x2,y2,z2,d2,x3,y3,z3,d3
TRANSL,x,y,z
XYROT,ang
YZROT
ZXROT
- **PPRINT IPV FIXTUR COMPOS** id id-list
STOCK
- **PPRINT IPV FIXTUR DECOMP** id id-list
STOCK
- **PPRINT IPV FIXTUR CONE** id x,y,z,i,j,k,r1,r2,h
STOCK
- **PPRINT IPV FIXTUR CYLNDR** id x,y,z,i,j,k,r,h
STOCK
- **PPRINT IPV FIXTUR LOAD** id "file"
STOCK

- PPRINT IPV FIXTUR MODIFY 0 color,visible,lucency,active,~
STOCK

PPRINT toler,id-list
- PPRINT IPV FIXTUR MOVE 0 x1,y1,z1,d1,x2,y2,z2,d2,~
STOCK 1
INCR

PPRINT x3,y3,z3,d3,id-list
- PPRINT IPV FIXTUR REMOVE id-list
STOCK
- PPRINT IPV FIXTUR SPHERE id x,y,z,r
STOCK
- PPRINT IPV FIXTUR STL id INCHES "file"
STOCK MM
- PPRINT IPV FIXTUR TORUS id x,y,z,i,j,k,r1,r2
STOCK
- PPRINT IPV FIXTUR TRANSL 0 x,y,z,id-list
STOCK 1
INCR
- PPRINT IPV FIXTUR XYROT 0 angle,id-list
STOCK YZROT 1
ZXROT INCR
- PPRINT IPV HOLDER parameters
- PPRINT IPV HOLDER DISPLAY "pt-list" parameters
- PPRINT IPV MODALS AUTO_HIDE [YES] [TRANS tval] [EDGES YES]
NO NO
- PPRINT IPV MODALS COLORS [CUT color] [CUTTER color] ~
PPRINT [SHANK color] [HOLDER color] [FIXTUR_CUT color] ~
PPRINT [HOLDER_CUT color] [RAPID_CUT color] ~
PPRINT [AUTO_COLOR color] [USE_STOCK YES] [USE_FIXTUR YES]
NO NO

- PPRINT IPV MODALS MACHINE MILL
 LATHE
 MILLTURN
 STRINGER
- PPRINT IPV MODALS STACK [ON] [FIXTUR YES] [size]
 OFF NO
- PPRINT IPV MODALS STOCK [COLOR color] [VISIBLE ON] ~
 FIXTUR OFF

PPRINT [TRANS tval] [TOLER tol] [IMPORTANT YES] ~
 NO

PPRINT [EDES color] [STL ASCII] [STL_STOP YES] ~
 BINARY NO

PPRINT [STL_DEACT YES] [STL_SKIP_ERROR YES]
 NO NO
- PPRINT IPV MODALS TOOL [TOLER tol] [MAXANG ang] ~
PPRINT [TRANS tval] [EDGES color] [MIN_HEIGHT hgt] ~
PPRINT [MIN_DIAMETER dia] [FROM_NEXT YES] ~
 NO

PPRINT [SHANK CUTTER] [RAPID rap}
 HOLDER
- PPRINT IPV OFFSET "file.ofs"
- PPRINT IPV OFFSET lab1 ofs [lab1 ofs2 [...] labn ofsn]
- PPRINT IPV POSITN lab1 pos1 [lab2 pos2 [...] labn posn]
- PPRINT IPV PRINT_SCREEN BMP size ["file"]
 GIF
 JPG
 PS
- PPRINT IPV SESSION EXPORT ["file"]
- PPRINT IPV SESSION IMPORT ["file"]
- PPRINT IPV SHANK parameters

- **PPRINT SHANK DISPLAY** "pt-list" parameters
- **PPRINT IPV SPINDLE** n1[,n2,[...],n10]
- **PPRINT IPV STOCK REMOVE_CHIPS** 0 x1,y1,z1,i1,j1,k1,...~
PPRINT xn,y,zn,in,j,kn
- **PPRINT IPV STOCK RESET_CUTCOLOR** [id-list]
- **PPRINT IPV STOCK SAVE** id ["file"]
FIXTUR
- **PPRINT IPV TOOL** [CUT_COLOR color] [CUTTER_COLOR color] ~
PPRINT CUTTER_EDGES color] [CUTTER_TRANS ctval] ~
PPRINT [HOLDER_COLOR color] [HOLDER_EDGES color] ~
PPIRNT [HOLDER_TRANS htval] [MAXANG ang] [RAPID rap] ~
PPRINT [SHANK CUTTER] [SHANK_COLOR color] ~
HOLDER
PPRINT [SHANK_EDGES color] [SHANK_TRANS stval] [TOLER tol]
- **PPRINT IPV TOOLPN** [label] [x,y,z,i,j,k,u,v,w]
- **PPRINT IPV VIEW FIT**

Functions

- **ABS**(scalar)
ACOS(scalar)
- **ANGLF**(line-1,line-2
 plane-1,plane-2
 vector-1,vector-2
 point-vector-1,point-vector-2
 vector,point-vector
 point-vector,vector
 circle,point)
- **ASIN**(scalar)

- **ATAN**(scalar)
- **ATAN2**(scalar-1, scalar-2)
- **CAN**(geo , scalar)
CONTCT
CUTTER
FEDRAT
HOLDER
MAXANG
MAXDP
NUMPTS
PSEUDO
SHANK
THICK
TOLER
UNITS
- **CBRTF**(scalar)
- **CLIPF**(plane-1, XLARGE[, CROSS][[...][, plane-6, XLARGE[, CROSS]]])
XSMALL ON XSMALL ON
YLARGE OFF YLARGE OFF
YSMALL
ZLARGE
ZSMALL
SAME
- **COLF**(color1[, color2[...]])
- **COS**(angle)
- **data-statement-label[element-position]**
DIST(curve)
- **DIST**(geo-1, geo-2)
- **DOT**(vector-1, vector-2)
plane-1, plane-2
point-vector-1, point-vector-2
point-vector, vector
vector, point-vector

- **EXPF**(scalar)
- **FILTER**(attribute-1[, . . .])
- **INT**(scalar)
- **LAYF**(layer1[, THRU, layern] [, layerm, [. . .]])
- **LOG**(scalar)
- **LOG10**(scalar)
- **LNTHF**(vector)
 point-vector
 line
 circle
 curve
- **MARKF**(marker-type1,marker-typ2,)
- **MAX1F**(scalar-1, scalar-2,)
- **MIN1F**(scalar-2, scalar-2,)
- **NINT**(scalar)
- **NUMF**(name-of-patern)
 NUM name-of-composite-curve
 name-of-array
 name-of-DATA-statement
- **SIGNF**(scalar-1, scalar-2)
- **SIN**(angle)
- **SQRT**(scalar)
- **SRFTYP**(surface)
- **TAN**(angle)

- **TDIST**(point-vector,entity,1)
 point,vector 2
 x,y,z,i,j,k -1
 -2
 near-point
- **TYPE**(variable)
- **XTYPE**(geometry)
- **VOCABF**(vocabulary-word)

APPENDIX E: Programmable Functions

The following is a list of programmable functions. Functions can either be assigned to a key in the following file:

C:\NCCS\NCL101\interface\nclkeys.kdf

or assigned to a specific menu in the files:

C:\NCCS\NCL101\interface\menu*.menu

Function	Description	CAM	CADD
KEY_ACTIVE_LINE	Mark the Current Line as Active Line Without Executing It While in Multiple-Line Command Line Mode	X	X
KEY_ALTACT	Alternate Action		X
KEY_BLINE	Move the Cursor to the Beginning of the Command Line	X	X
KEY_CAPTURE_TOGGLE	Toggle Window Capture	X	X
KEY_CLEAR	Clear The Command Line	X	X
KEY_DEFVAL	Accept Default Displayed in the Command Line/Prompt Area	X	X
KEY_DELETE_LINE	Delete a Line in Multiple-Line Window	X	X
KEY_DELLAST	Delete Last Entity		X
KEY_DNFIELD	Down Arrow	X	X
KEY_DONE	Accept Operation (DONE)	X	X
KEY_ELINE	Move the Cursor to the End of the Command Line	X	X
KEY_FUNCTION	Select the Next Entity in Picking Mode	X	X
KEY_HELP	Display Help Text for the Current Prompt if There is Any	X	X
KEY_INSERT_LINE	Insert a New Line While in Multiple-Line Window	X	X
KEY_LOCATE	Toggle to Locate Mode	X	X
KEY_NOOP	No Operation	X	X
KEY_PANIC	Panic Stop	X	X
KEY_PICK	Toggle to Pick Mode	X	X
KEY_PICK_LOCATE	Replicate the Standard Pick/Locate Functions in Previous Versions Before NCL99	X	X
KEY_PIKLOC	Toggle Pick/Locate	X	X

Function	Description	CAM	CADD
KEY_PLAYBACK [,”file_name”]	Start Playback, “file_name” is the name of the playback file to execute. If file_name is not specified, NCL will prompt the user for the name.	X	X
KEY_PLAYPAUSE	Pause Playback	X	X
KEY_PLAYRESUM	Resume Playback	X	X
KEY_PLAYSTOP	Stop Playback	X	X
KEY_RECMSG	Record User Message	X	X
KEY_RECOFF	Stop Recording	X	X
KEY_RECON	Start Recording	X	X
KEY_RECPRM	Record Prompt for Input	X	X
KEY_REJECT	Reject Operation	X	X
KEY_ROOT	Go to ROOT	X	X
KEY_TEXT	Toggle to Text Mode	X	X
KEY_UPFIELD	Up Arrow	X	X
NIS_BATCH_ONCE	One Shot Utility Batch	X	X
NIS_BATCH_STATUS	Show Batch Status	X	X
NIS_COPY_FILE	Copy a File	X	X
NIS_DELETE_FILE	Delete a File	X	X
NIS_DISPLAY_BATLOG	Display Batch Log	X	X
NIS_DISPLAY_CURRENT	Display Current PP File	X	X
NIS_DISPLAY_FILE	Display by File Name	X	X
NIS_DISPLAY_MODALS	Display File Modals	X	X
NIS_DISPLAY_NCLQUE	Display NCL Queue	X	X
NIS_DISPLAY_PR	Display Current Print File	X	X
NIS_DISPLAY_PR1	Display Current PR1 File	X	X
NIS_DISPLAY_PU1	Display Current PU1 File	X	X
NIS_EDIT_CURRENT	Edit Current PP File	X	X
NIS_EDIT_FILE	Edit by File Name	X	X
NIS_EDIT_PR	Edit Current Print File	X	X
NIS_EDIT_PR1	Edit Current PR1 File	X	X
NIS_EDIT_PU1	Edit Current PU1 File	X	X

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
NIS_IFACE_MODALS	Interface Modals	X	X
NIS_LOAD_LAYOUT	Load Layout File	X	X
NIS_LOAD_MOUSE[,"filename"]	Load an External Mouse Definition File. If "filename" is not specified then the user will be prompted for the name.	X	X
NIS_MENU DESIGN	Display Menu Design Form	X	X
NIS_MODAL_OUTPUT	Display Modals Files Output Form	X	X
NIS_NCL_BATCH	Run NCL Batch	X	X
NIS_NCL_QUE	NCL Batch Queue	X	X
NIS_PLOT_DRAWING	Plot Drawing File	X	X
NIS_PLOT_FILE	Off-line Plot Routines	X	X
NIS_PLOT_MODAL	Off-line Plot Modals	X	X
NIS_PRINT_CURRENT	Print Current PP File	X	X
NIS_PRINT_FILE	Print by File Name	X	X
NIS_PRINT_PR	Print Current Print File	X	X
NIS_PRINT_PR1	Print Current PR1 File	X	X
NIS_PRINT_PU1	Print Current PU1 File	X	X
NIS_PRINT_SCREEN	Print Current Screen	X	X
NIS_PRINT_STATUS	Show Print Queue	X	X
NIS_QUE_MODAL	NCL Queue Modals	X	X
NIS_RENAME_FILE	Rename a File	X	X
NIS_SAVE_LAYOUT	Save Layout File	X	X
NIS_SYSTEM_ENTER	Enter Operating System	X	X
NIS_SYSTEM_MENU	Operating System Menu	X	X
NIS_SYSTEM_ONCE [, "command, NO/YES"]	Execute System Command <i>command</i> = System command to execute YES = Open a system window to run the command in NO = This command does not require a system window If no <i>command</i> is specified, NCL will prompt the user for the command.	X	X
NCL_ANALYZE_GEO	Display Measurement Form	X	
NCL_ANALYZE_SURF	Analyze Surface	X	X
NCL_ANALYZE_SURF_THRU	Analyze Surface Through Surface	X	X
NCL_AUTOSAVE_FORM	Open the AutoSave Form	X	X

Function	Description	CAM	CADD
NCL_BACKGROUND	Display Background Display Form	X	X
NCL_CALCULATOR	Display Calculator	X	X
NCL_CHANGE_VIEW	Change View Form	X	X
NCL_CHG_COLOR	Change Default Color	X	X
NCL_COLOR_MODAL	Display the Geometry Colors Form	X	X
NCL_DELETE_VIEW	Delete View	X	X
NCL_DISPLAY_BUFFER	Display the Buffer Display Form	X	X
NCL_DOMINATE_TOGGLE	Turn SpaceMouse Dominate Mode On/Off, Windows Only	X	X
NCL_DYNAMIC_CENTER	Dynamic View Center	X	X
NCL_DYNAMIC_MODAL	Dynamic Viewing Modals	X	X
NCL_DYNAMIC_MOUSE	Dynamic Mouse Viewing	X	X
NCL_DYNAMIC_PAN	Dynamic Pan	X	X
NCL_DYNAMIC_TUMBLE	Dynamic Tumble	X	X
NCL_DYNAMIC_UNZOOM	Reset View	X	X
NCL_DYNAMIC_VCROT	Dynamic Vector Rotation	X	X
NCL_DYNAMIC_XYROT	Dynamic XY-Rotation	X	X
NCL_DYNAMIC_ZOOM	Dynamic Zoom	X	X
NCL_DYNAMIC_ZROT	Dynamic Z-Rotation	X	X
NCL_EDGE_OFF_ALL	Turn Off Edge Display for All Surfaces	X	X
NCL_EDGE_ON_ALL	Turn On Edge Display for All Surfaces	X	X
NCL_ENTITY_ATTRIB	Geometric Entity Attribute	X	X
NCL_ENTITY_DATA	Geometric Entity Data	X	X
NCL_EXIT	Exit NCL	X	X
NCL_FIT_VIEW	Fit View	X	X
NCL_FLIP_VIEW	View From Opposite Direction	X	X
NCL_GAIN_DEFAULT	Set SpaceMouse Sensitivity to Default Value, Windows Only	X	X
NCL_GAIN_DOWN	Set SpaceMouse Sensitivity Down 10%, Windows Only	X	X
NCL_GAIN_UP	Set SpaceMouse Sensitivity Up 10%, Windows Only	X	X
NCL_GRID_OFF	Turn Off Grid	X	X
NCL_GRID_ON	Grid Active Form	X	X
NCL_INPUT_LAYER	Set Input Layer	X	X

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
NCL_INVISIBLE_ALL	Invisible All Displayed	X	X
NCL_INVISIBLE_GEO	Invisible Geometry	X	X
NCL_LABEL_ALTER	Alter Label Locations	X	X
NCL_LABEL_MODALS	Display Label Modals Form	X	X
NCL_LABEL_OFF	Erase Selected Labels	X	X
NCL_LABEL_ON	Displayed Selected Labels	X	X
NCL_LABEL_OVERLAP	Redisplay Labels to Minimize Overlap in a Given View	X	X
NCL_LABEL_RESET	Redisplay All Labels at Their Default Locations	X	X
NCL_LAYERS	Displays Layers Management Form	X	X
NCL_LBL_LDR_OFF	Do Not Display Label Leader Line	X	X
NCL_LBL_LDR_ON	Display Label Leader Line	X	X
NCL_LIMIT_GEO	Limit Pickable Geometry Type	X	X
NCL_LINE_STYLE	Set Default Line Style	X	X
NCL_LOAD_COLORS	Load an External Color Modals File	X	X
NCL_LOAD_SESSION	Load NCL Session	X	X
NCL_LOAD_VIEW	Load View	X	X
NCL_MDAXIS_ALIGN	Align Model Axis to View Port	X	X
NCL_MDAXIS_AXIS	Set Modeling Axis Axes	X	X
NCL_MDAXIS_OFF	Invisible Modeling Axis	X	X
NCL_MDAXIS_ON	Display Modeling Axis	X	X
NCL_MDAXIS_ORIG	Set Modeling Axis Origin	X	X
NCL_MDAXIS_RESET	Reset Modeling Axis	X	X
NCL_MDAXIS_ROT	Rotate Modeling Axis	X	X
NCL_MDAXIS_SNAP	Snap Modeling Axis to Axis	X	X
NCL_MERGE_COLORS	Merge an External Color Modals File into the Current Color Definitions.	X	X
NCL_MID_TRIM	Mid-Trim Geometry	X	
NCL_MIN_ALL	Minimize Menus and Graphics	X	X
NCL_MIN_MENU	Minimize Menus Only	X	X
NCL_MISC_MODALS	Miscellaneous Modals Form	X	X
NCL_MOD_ATTR	Set Default Attributes	X	X

Function	Description	CAM	CADD																																						
NCL MODIFY ATTRIBS	Modify Geometry Attributes	X	X																																						
NCL NAME MODALS	Set Geometry Naming Modals	X	X																																						
NCL NEW SESSION	Start a new interactive session without allocating an NCL license	X	X																																						
NCL OPEN FILE, [longerfilename,"filename[,ftype]"]	<p>Allow user to open various file types that can currently be opened using the interface.</p> <p>Following are values supported by <i>ftype</i> and the associated file types.</p> <table> <thead> <tr> <th>Values</th><th>File Type</th></tr> </thead> <tbody> <tr><td>0</td><td>Part Program File</td></tr> <tr><td>1</td><td>Unibase File</td></tr> <tr><td>2</td><td>Record/Playback File</td></tr> <tr><td>3</td><td>IGES Model</td></tr> <tr><td>4</td><td>NCL Session File</td></tr> <tr><td>5</td><td>Stock Commands File</td></tr> <tr><td>6</td><td>STL Model</td></tr> <tr><td>7</td><td>Layout File</td></tr> <tr><td>8</td><td>Menu File</td></tr> <tr><td>9</td><td>Key Definition File</td></tr> <tr><td>10</td><td>Tool Library</td></tr> <tr><td>11</td><td>Clfile</td></tr> <tr><td>12</td><td>MCD file</td></tr> <tr><td>13</td><td>CATIA binary Clfile</td></tr> <tr><td>14</td><td>MasterCam Clfile</td></tr> <tr><td>15</td><td>Simulation File</td></tr> <tr><td>16</td><td>NCL/IPV Session File</td></tr> <tr><td>17</td><td>Drawing File</td></tr> </tbody> </table>	Values	File Type	0	Part Program File	1	Unibase File	2	Record/Playback File	3	IGES Model	4	NCL Session File	5	Stock Commands File	6	STL Model	7	Layout File	8	Menu File	9	Key Definition File	10	Tool Library	11	Clfile	12	MCD file	13	CATIA binary Clfile	14	MasterCam Clfile	15	Simulation File	16	NCL/IPV Session File	17	Drawing File	X	X
Values	File Type																																								
0	Part Program File																																								
1	Unibase File																																								
2	Record/Playback File																																								
3	IGES Model																																								
4	NCL Session File																																								
5	Stock Commands File																																								
6	STL Model																																								
7	Layout File																																								
8	Menu File																																								
9	Key Definition File																																								
10	Tool Library																																								
11	Clfile																																								
12	MCD file																																								
13	CATIA binary Clfile																																								
14	MasterCam Clfile																																								
15	Simulation File																																								
16	NCL/IPV Session File																																								
17	Drawing File																																								
NCL PAN	Pan View	X	X																																						
NCL PAN_TOGGLE	Turn SpaceMouse Translation On/Off, Windows Only	X	X																																						
NCL PICK_APERTURE	Change Pick Aperture	X	X																																						
NCL PICK_MARK	Choose Pick Marker	X	X																																						
NCL PICK_MODALS	Pick Modals Form	X	X																																						
NCL PREVIOUS_VIEW	Previous View	X	X																																						
NCL REDEF_SF	Display the Modify Surface Parameters Form	X																																							
NCL REPAINT_FULL	Full Screen Repaint	X	X																																						
NCL REPAINT_VIEW	Repaint View	X	X																																						
NCL RESET_SPACEMOUSE	Reset the SpaceMouse Connection	X	X																																						
NCL RESET_VIEW	Reset View	X	X																																						
NCL RESTART	Restart NCL	X	X																																						
NCL ROTATE_TOGGLE	Turn SpaceMouse Rotation On/Off, Windows Only	X	X																																						

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
NCL_SAVE_SESSION	Save NCL Session	X	X
NCL_SAVE_VIEW	Save View	X	X
NCL_SAVE_VIEWFMT	Save Displayed View	X	X
NCL_SCALE_VIEW	Scale View	X	X
NCL_SCALE_ZOOM	Scale Zoom	X	X
NCL_SCREEN_SELECT [screen_name]	To Control the Screen Layout Forms. “screen_name” is the screen name to be defined and displayed.	X	X
NCL_SELECT_ALL	Select All Geometry	X	X
NCL_SELECT_CHAIN	Chain Geometry Selection	X	X
NCL_SELECT_CHMOD	Chain Select Modals	X	X
NCL_SELECT_CLIP	Filter Bounding Region Form	X	X
NCL_SELECT_FALL	Select Filtered All	X	X
NCL_SELECT_FCHN	Select Filtered Chain	X	X
NCL_SELECT_filt	Set Selection Filter	X	X
NCL_SELECT_FREGIN	Select Filtered Region In	X	X
NCL_SELECT_FREGINX	Select Filtered Region In Cross	X	X
NCL_SELECT_FREGOUT	Select Filtered Region Out	X	X
NCL_SELECT_FREGOUTX	Select Filtered Region Out Cross	X	X
NCL_SELECT_FSING	Select Filtered Single	X	X
NCL_SELECT_GEO	Select Geometry Types	X	X
NCL_SELECT_REGIN	Region in Selection	X	X
NCL_SELECT_REGINX	Region in Cross Selection	X	X
NCL_SELECT_REGOUT	Region Out Selection	X	X
NCL_SELECT_REGOUTX	Region Out Cross Selection	X	X
NCL_SELECT_REJECT	Reject Geometry Selection	X	X
NCL_SELECT_SINGLE	Single Geometry Selection	X	X
NCL_SELECT_UNALL	Deselect All	X	X
NCL_SELECT_UNLAST	Deselect Last	X	X
NCL_SET_ATTRIBS	Set Geometry Attributes	X	X
NCL_SET_INCH	Set Units to Inch	X	X
NCL_SET_LAYER	Set Layer Number	X	X

Function	Description	CAM	CADD
NCL_SET_MM	Set Units to Millimeter	X	X
NCL_SHADE_ENTITY	Shade Entity	X	X
NCL_SWAP_ENTITIES	Swaps the Visibility of Geometry, i.e. Visible Geometry Will be Made Invisible and Invisible Geometry Will be Made Visible	X	X
NCL_SWAP_VIEW	Swap the Active View With the Invisible View and Vice Versa	X	X
NCL_TOGGLE_SPACEMOUSE	Toggle SpaceMouse between NCL and NCL/IPV , Windows Only	X	X
NCL_TRIM_EXTEND	Trim/Extend Geometry	X	X
NCL_TVVIEW_DOWN	Toggle Current View Port Down	X	X
NCL_TVVIEW_UP	Toggle Current View Port Up	X	X
NCL_UNIBASE_MODALS	Unibase Save Modals	X	X
NCL_UNIBASE_STAT [,"TRUE"]	Open the Unibase Statistics Form. Optional TRUE Parameter Allows the User to Change All Of the Statistics Associated With the Unibase. If TRUE is Not Specified, Then Only the User Defined Notes Section Can be Modified.	X	X
NCL_UNSHADE_ENTITY	Unshade Entity	X	X
NCL_UNTRIM	Untrim Geometry	X	X
NCL_UPDATE_TIME	Open the time status menu function	X	X
NCL_VERIFY_TOGGLE	Toggle Verify Mode	X	X
NCL_VIEW_HIDDEN	Set View Port to Hidden Line Removal	X	X
NCL_VIEW_MATRIX	View by Matrix	X	X
NCL_VIEW_NEGX	View Along Negative X-axis	X	X
NCL_VIEW_NEGY	View Along Negative Y-axis	X	X
NCL_VIEW_NEGZ	View Along Negative Z-axis	X	X
NCL_VIEW_REFSYS	View by REFSYS Matrix	X	X
NCL_VIEW_SHADED	Set View Port to Shaded.	X	X
NCL_VIEW_SHADED_ONLY	Set View Port to Shaded Only	X	X
NCL_VIEW_TOOL	View Down Tool Axis	X	
NCL_VIEW_TRACUT	View by TRACUT Matrix	X	X
NCL_VIEW_WIREFRAME	Set View Port to Wireframe	X	X
NCL_VIEW_X	View Along X-axis	X	X

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
NCL_VIEW_Y	View Along Y-axis	X	X
NCL_VIEW_Z	View Along Z-axis	X	X
NCL_VISIBLE_ALL	Visible All Entities	X	X
NCL_VISIBLE_GEO	Visible Entities	X	X
NCL_WINDOW_ZOOM	Window Zoom	X	X
NCL_WPAXIS_ALIGN	Align Work Plane Axis to View Port	X	X
NCL_WPAXIS_AXIS	Set Work Plane Axis Axes	X	X
NCL_WPAXIS_OFF	Invisible Work Plane Axis	X	X
NCL_WPAXIS_ON	Display Work Plane Axis	X	X
NCL_WPAXIS_ORIG	Set Work Plane Axis Origin	X	X
NCL_WPAXIS_PLANE	Snap Work Plane Axis to Plane	X	X
NCL_WPAXIS_RESET	Rest Work Plane Axis	X	X
NCL_WPAXIS_ROT	Rotate Work Plane Axis	X	X
NCL_WPAXIS_SNAP	Snap Work Plane Axis to Axis	X	X
NCL_XPAN_LEFT	Immediate PAN Left	X	X
NCL_XPAN_RIGHT	Immediate PAN Right	X	X
NCL_XROT_LEFT	Immediate Rotate X Left	X	X
NCL_XROT_RIGHT	Immediate Rotate X Right	X	X
NCL_YPAN_LEFT	Immediate PAN Down	X	X
NCL_YPAN_RIGHT	Immediate PAN Up	X	X
NCL_YROT_LEFT	Immediate Rotate Y Left	X	X
NCL_YROT_RIGHT	Immediate Rotate Y Right	X	X
NCL_ZOOM_DOWN	Immediate Zoom Out	X	X
NCL_ZOOM_UP	Immediate Zoom In	X	X
NCL_ZROT_LEFT	Immediate Rotate Z Left	X	X
NCL_ZROT_RIGHT	Immediate Rotate Z Right	X	X
CAM_ANOTE_ATTRIBS	Set Default Annotation Attributes	X	
CAM_ANOTE_ENGRAVE	Display the Letter Engraving Form For Engraving Annotation	X	
CAM_ANOTE MODIFY	Modify Annotation Attributes	X	
CAM_ARCSLP_FILLET_OFF	Output ARCSLP/FILLET,0 Command to Part Program	X	
CAM_ARCSLP_FILLET_RAD	Set ARCSLP/FILLET Command	X	

Function	Description	CAM	CADD
CAM_AUTONAME_TOGGLE	Toggle Auto Naming	X	
CAM_AUTOPS	AUTOPS	X	
CAM_COMMAND	Command Mode	X	
CAM_COMMAND_SRC	Set Command Source Format	X	
CAM_CONSOL_MODE	Insert Mode Off	X	
CAM_CONTCT_OFF	CONTCT/OFF	X	
CAM_CONTCT_ON	CONTCT/ON	X	
CAM_CONTOUR	Contouring Form	X	
CAM_COPY_MATRIX	Copy Geometries Using Matrix	X	
CAM_COPY_MIRROR	Copy Geometries Using Mirror Matrix	X	
CAM_COPY MODIFY	Copy Motion Matrix	X	
CAM_COPY_PTPT	Copy Geometries Point to Point	X	
CAM_COPY_ROTATE	Copy Geometries Using Rotation	X	
CAM_COPY_SAME	Copy Motion with No Modification	X	
CAM_COPY_SCALE	Copy Geometries Through Scale Matrix	X	
CAM_COPY_TRANSL	Copy Motion Translation	X	
CAM_COPY_VECTOR	Copy Geometries Using Vector	X	
CAM_COPY_XYROT	Copy Motion XY Rotation	X	
CAM_COPY_YZROT	Copy Motion YZ Rotation	X	
CAM_COPY_ZXROT	Copy Motion ZX Rotation	X	
CAM_CREATE_AS	Create APT Source	X	
CAM_CREATE_CL	Create Clfile	X	
CAM_CUT	Cut (Pass Points to Clfile)	X	
CAM_CUTTER_PARMS	Cutter Parameter Form	X	
CAM_DATA_FORM	Display Select Data Form	X	
CAM_DECOMP_ENTITY	Extract Components from Composite Solids and Net Surfaces	X	
CAM_DECOMP_SYM	Decompose Symbol Instance	X	
CAM_DELETE_LINE	Delete Part Program Line	X	
CAM_DISPLAY_CUTTER	Display Cutter	X	
CAM_DNTCUT	DNTCUT (Do Not Output Points to Clfile)	X	
CAM_DNTCUT_NM	DNTCUT/NOMORE	X	

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
CAM_DOLOOP	Start DO Loop	X	
CAM_EDIT_CL	Edit Active Clfile	X	
CAM_EDIT_LINE	Edit Part Program Line	X	
CAM_EDIT_PP	Edit Part Program File	X	
CAM_EDIT_TOGGLE	Toggle Edit Mode	X	
CAM_ERASE_CL	Erase Active Clfile	X	
CAM_ERASE_MOTION	Erase Motion	X	
CAM_FEDRAT	Set Current Feedrate	X	
CAM_FEDRAT_AT	Set Slowdown/Acceleration Feedrate	X	
CAM_FEDRAT_FPM	Set FPM Feedrate	X	
CAM_FEDRAT_FPR	Set FPR Feedrate	X	
CAM_FEDRAT_RAPID	Set RAPID Feedrate	X	
CAM_FIND_LINE	Find Part Program Line	X	
CAM_FIND_TOKEN	Find Part Program Text	X	
CAM_FINI	FINI	X	
CAM_FMILL	Flowline Milling	X	
CAM_FROM_PT	From Point or Point-Vector	X	
CAM_FROM_XY	From XY Location	X	
CAM_FROM_XYZ	From XYZ Location	X	
CAM_FUNC_CAN	CAN Function	X	
CAM_FUNC_NPATERN	NUM(Pattern) Function	X	
CAM_FUNC_NRESERV	NUM(Subscript_name) Function	X	
CAM_GEO_ANOTE	Create Annotation	X	
CAM_GEO_CI_CECI	Circle Center Tangent to Circle	X	
CAM_GEO_CI_CELN	Circle Center Tangent to Line	X	
CAM_GEO_CI_CEPT	Circle Center and Point on Circle	X	
CAM_GEO_CI_CERA	Circle Center and Radius	X	
CAM_GEO_CI_CETT	Circle Center Tangent to Entity	X	
CAM_GEO_CI_GEOM	Circle Center/Vector/Radius	X	
CAM_GEO_CI_LNLNLN	Circle Tangent to 3 Lines	X	
CAM_GEO_CI_OFFSET	Circle Offset from Circle	X	

Function	Description	CAM	CADD
CAM_GEO_CI_PARAM	Circle Using 7 Parameters	X	
CAM_GEO_CI_PTPTPT	Circle Through 3 Points	X	
CAM_GEO_CI_PTPTVE	Circle Through 2 Points with Tangent Vector	X	
CAM_GEO_CI_TANTO	Circle Tangent to 2 Entities with Radius	X	
CAM_GEO_CI_TRANSL	Circle Translated by Vector	X	
CAM_GEO_CI_XYR	Circle XY Center and Radius	X	
CAM_GEO_CI_XYZR	Circle XYZ Center and Radius	X	
CAM_GEO_COMPCV_OUT	Extract Composite Curve Component(s)	X	
CAM_GEO_CV_COMPOSIT	Composite Curve	X	
CAM_GEO_CV_CONIC	Curve Defined as Conic	X	
CAM_GEO_CV_EDGE	Curve From U or V Lines of Surface	X	
CAM_GEO_CV_FIT_PTPT	Curve Fit Point Through Point	X	
CAM_GEO_CV_FIT PTS	Curve Fit From Multiple Points	X	
CAM_GEO_CV_FIT_PTVE	Curve Fit Point and Vector Through Point and Vector	X	
CAM_GEO_CV_IOSF	Curve Intersection of Surface and Surface/Plane	X	
CAM_GEO_CV_OFFCOMP	Curve Offset From Composite Curve	X	
CAM_GEO_CV_OFFSET	Curve Offset From Curve	X	
CAM_GEO_CV_PT	Curve From Multiple Points/Point-vectors	X	
CAM_GEO_CV_PTPT	Curve Point Through Point	X	
CAM_GEO_CV_PTVE	Curve From Multiple Points and Vectors	X	
CAM_GEO_CV_TRANSL	Curve Translated by Vector	X	
CAM_GEO_DATA	Bring Up the Select Data Entities Form that Brings Up the Select Data Entities Form Allowing User to Interactively Create a DATA Statement By Picking Geometry From the Screen.	X	
CAM_GEO_DATA_FILE	Bring Up the Load Delimited File Form Allowing User to Create a DATA Definition that Loads an Existing Comma Or Tab Delimited File.	X	
CAM_GEO_LN_CIAALN	Line Tangent to Circle at Angle to a Line	X	
CAM_GEO_LN_CICI	Line Tangent to 2 Circles	X	
CAM_GEO_LN_CONNECT	Line Connected to Last Line	X	
CAM_GEO_LN_FWD	Line From Forward Direction of Motion	X	
CAM_GEO_LN_OFFSET	Line Offset From a Line by a Vector	X	

Function	Description	CAM	CADD
CAM_GEO_LN_PALN	Line Parallel to Line	X	
CAM_GEO_LN_PL	Line From a Plane	X	
CAM_GEO_LN_PLPL	Line Intersection of Plane and Plane	X	
CAM_GEO_LN_PTAALN	Line Through Point at Angle to a Line	X	
CAM_GEO_LN_PTPALN	Line Through Point Parallel to a Line	X	
CAM_GEO_LN_PTPERPTO	Line Through Point Perpendicular to a Line or Curve	X	
CAM_GEO_LN_PTPT	Line Through 2 Points	X	
CAM_GEO_LN_PTTANTO	Line Through Point Tangent to a Circle or Curve	X	
CAM_GEO_LN_XAXIS	Line Parallel to X-axis on Current XY-Plane	X	
CAM_GEO_LN_XY	Line Through 2 XY Locations	X	
CAM_GEO_LN_XYZ	Line Through 2 XYZ Locations	X	
CAM_GEO_LN_YAXIS	Line Parallel to Y-axis on Current XY-Plane	X	
CAM_GEO_MX_COORDSYS	Create Matrix that Will Convert From One Coordinate System to Another Coordinate System By Specifying Three Points in Each Coordinate System.	X	
CAM_GEO_MX_INVERSE	Matrix Inverse Matrix	X	
CAM_GEO_MX_MIRROR	Mirror Matrix	X	
CAM_GEO_MX_MXMX	Matrix Times Matrix	X	
CAM_GEO_MX_PARAMS	Matrix Using 12 Parameters	X	
CAM_GEO_MX_PTVEVE	Matrix Using Point Vector Vector	X	
CAM_GEO_MX_ROTATE	Rotation Matrix	X	
CAM_GEO_MX_SCALE	Scale Matrix	X	
CAM_GEO_MX_TRANSL	Translation Matrix	X	
CAM_GEO_NF_4CV	N-Surf Using 4 Boundary Entities	X	
CAM_GEO_NF_CURVES	N-Surf Through Multiple Entities	X	
CAM_GEO_NF_CURVES0	N-Surf Through Entities Using Slope	X	
CAM_GEO_NF_FIT	N-Surf Fit Through Multiple Entities	X	
CAM_GEO_NF_FIT_THRU	N-Surf Fit Through Entities	X	
CAM_GEO_NF_REVOLVE	N-Surf of Revolution	X	
CAM_GEO_NF RULED	N-Surf Through 2 Entities	X	
CAM_GEO_PL_FIT	Optimal Plane From a Curve or a Surface	X	

Function	Description	CAM	CADD
CAM_GEO_PL_IJKD	Plane IJK Vector and Distance	X	
CAM_GEO_PL_LNPEPL	Plane Through Line Perpendicular to Plane	X	
CAM_GEO_PL_PAPL	Plane Parallel to a Plane	X	
CAM_GEO_PL_PNTVEC	Plane Perpendicular to a Point-vector	X	
CAM_GEO_PL_PTPAPL	Plane Through Point Parallel to a Plane	X	
CAM_GEO_PL_PTPEPL	Plane Through Point Perpendicular to 2 Planes	X	
CAM_GEO_PL_PTPEVE	Plane Through Point Particular to Vector	X	
CAM_GEO_PL_PTPTPE	Plane Through 2 Points Particular to a Plane	X	
CAM_GEO_PL_PTPTPT	Plane Through 3 Points	X	
CAM_GEO_PL_PTVEVE	Plane Through Point Parallel to 2 Vectors	X	
CAM_GEO_PN_CIRC	Circular Pattern	X	
CAM_GEO_PN_CIRCIN	Circular Increment Pattern	X	
CAM_GEO_PN_PNVE	Pattern Parallel to Pattern Along Vector	X	
CAM_GEO_PN_PNVEIN	Pattern Parallel to Pattern Increment Along Vector	X	
CAM_GEO_PN_PROJECT	Project Pattern onto Surface	X	
CAM_GEO_PN_PTPN	Random Pattern	X	
CAM_GEO_PN_PTPTVE	Pattern Using 2 Points and Vector	X	
CAM_GEO_PN_PTVEIN	Pattern at Point Increment Along Vector	X	
CAM_GEO_PT_AACI	Point at Angel on Circle	X	
CAM_GEO_PT_CECI	Point at Center of Circle	X	
CAM_GEO_PT_DSCV	Point at Distance Along Curve	X	
CAM_GEO_PT_ENDPT	End Point of Line or Circle	X	
CAM_GEO_PT_FIT	Point at Center of Gravity of Curve or Surface On the Optimal Plane	X	
CAM_GEO_PT_GENPT	Generate Tool End Points	X	
CAM_GEO_PT_GPTNM	Stop Generating Tool End Points	X	
CAM_GEO_PT_INTOF	Point at Intersection of 2 Entities	X	
CAM_GEO_PT_INTOF3	Intersection Point of 2 Entities in 3-D space	X	
CAM_GEO_PT_OFFSET	Point Offset From Point	X	
CAM_GEO_PT_ONCV	Point at U Value or Percentage on a Curve	X	
CAM_GEO_PT_ONSF	Point at UV Value or Percentage on a Surface	X	

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
CAM_GEO_PT_PN	Point From Pattern	X	
CAM_GEO_PT_PROJECT	Project Point onto Surface or Visual Solid	X	
CAM_GEO_PT_PTCI	Point Through Point at Angle to Circle	X	
CAM_GEO_PT_TEND	Point at Current Tool Location	X	
CAM_GEO_PT_XY	Point at XY Coordinates	X	
CAM_GEO_PT_XYZ	Point at XYZ Coordinates	X	
CAM_GEO_PV_CECI	Point-vector Through Center of Circle Perpendicular to the Circular Plane	X	
CAM_GEO_PV_CROSS	Point-vector Cross of Point-vector and Vector	X	
CAM_GEO_PV_DSCV	Point-vector at Distance Along Curve	X	
CAM_GEO_PV_FIT	Point-vector at Center of Gravity of Curve or Surface and Perpendicular to the Optimal Plane	X	
CAM_GEO_PV_GENPT	Generate Tool Point/Vector Locations	X	
CAM_GEO_PV_LN	Point-vector on a Line	X	
CAM_GEO_PV_MINUS	Point-vector Minus Point-vector and Vector	X	
CAM_GEO_PV_OFFSET	Point-vector Offset From Point-vector	X	
CAM_GEO_PV_ONCV	Unit Point-vector at U Value or Percentage on a Curve in the Forward Sense of the Curve	X	
CAM_GEO_PV_ONSF	Unit Point-vector at UV Value or Percentage on Surface Along the Surface Normal	X	
CAM_GEO_PV_PEPL	Point-vector Perpendicular to a Plane	X	
CAM_GEO_PV_PLPLPL	Point-vector Intersection of 3 Planes	X	
CAM_GEO_PV_PLUS	Point-vector Plus Point-vector and Vector	X	
CAM_GEO_PV_PN	Point-vector From Pattern	X	
CAM_GEO_PV_PROJECT	Project Point-Vector onto Surface or Visual Solid	X	
CAM_GEO_PV_PTPT	Point-vector Through 2 Points	X	
CAM_GEO_PV_PTVE	Point-vector At a Point with Vector	X	
CAM_GEO_PV_REVSF	Unit Point-vector Along the Axis of a Surface of Revolution and at the Center of the Base	X	
CAM_GEO_PV_TEFWD	Point-vector at Tool End and Along Forward Vector	X	
CAM_GEO_PV_TETA	Point-vector at Tool End and Along Tool Axis	X	
CAM_GEO_PV_TIMES	Point-vector Times Point-vector and Vector	X	
CAM_GEO_PV_TTPE	Point-vector Perpendicular or TANTO Another Entity	X	

Function	Description	CAM	CADD
CAM_GEO_PV_UNIT	Unit Point-vector Along Point-Vector	X	
CAM_GEO_PV_XYIJK	Point-vector XY Coordinates with IJK Vector	X	
CAM_GEO_PV_XYZIJK	Point-vector XYZ Coordinates with IJK Vector	X	
CAM_GEO_SF_ATTRIB	Modify Surface Attributes	X	
CAM_GEO_SF_CURVES	Surface Through Multiple Entities with Slope	X	
CAM_GEO_SF_CURVESO	Surface Through Multiple Entities	X	
CAM_GEO_SF_FILLET	Variable Fillet Surface	X	
CAM_GEO_SF_FIT	Surface Fit Through Multiple Entities	X	
CAM_GEO_SF_FIT_THRU	Surface Fit Through Entities	X	
CAM_GEO_SF_MESH	Mesh Surface	X	
CAM_GEO_SF_MODALS	Set Surface Modals	X	
CAM_GEO_SF_NET	Net Surface	X	
CAM_GEO_SF_NET_THRU	Net Surface, Surface Through Surface	X	
CAM_GEO_SF_OFFSET	Surface Offset From Surface	X	
CAM_GEO_SF_OUT	Extract Surface From Trimmed Surface	X	
CAM_GEO_SF_PLPLRA	Fillet Surface Tangent to 2 Planes	X	
CAM_GEO_SF_QUILT	Quilt Surface	X	
CAM_GEO_SF_REDEF	Redefine a Surface	X	
CAM_GEO_SF_REVOLVE	Surface of Revolution	X	
CAM_GEO_SF RULED	Surface Through 2 Entities	X	
CAM_GEO_SF_TRIM	Trim Surface	X	
CAM_GEO_SFS_CONTOUR	Create Contour Around Surfaces	X	
CAM_GEO_SFS_IO	Create Intersection Curve of Z-Level Plane With Surfaces	X	
CAM_GEO_SH_ATTRIB	Modify Shape Attributes	X	
CAM_GEO_SH_MODALS	Set Shape Modals	X	
CAM_GEO_SH_PPWORDS	Set Shape Postprocessor Words	X	
CAM_GEO_SHAPE	Lathe Shape	X	
CAM_GEO_SP_EDGE	Spline From U or V Line of Surface	X	
CAM_GEO_SP_EDGE_OUT	Extract Trimmed Surface Edge Curve	X	
CAM_GEO_SP_EDGES_OUT	Extract Chain of Trimmed Surface Edges as a Composite Curve	X	

Function	Description	CAM	CADD
CAM_GEO_SP_FIT_PTPT	Spline Fit Point Through Point	X	
CAM_GEO_SP_FIT PTS	Spline Fit Multiple Points	X	
CAM_GEO_SP_FIT_PTVE	Spline Fit Point-vector/Vector Through Point-vector/Vector	X	
CAM_GEO_SP_IOSF	Spline Intersection of Surface and Surface/Plane	X	
CAM_GEO_SP_OFFSET	Spline Offset from Spline (Along Normal)	X	
CAM_GEO_SP_OUT	Extract Trimmed Surface Curve	X	
CAM_GEO_SP_PROJECT	Project Spline onto Surface	X	
CAM_GEO_SP_PT	Spline From Multiple Points/Point-vectors	X	
CAM_GEO_SP_PTPT	Spline Point Through Point	X	
CAM_GEO_SP_PTVE	Spline From Multiple Points and Vectors	X	
CAM_GEO_SP_REVSF	A Composite Curve or Spline Along the Generating Curve of Surface of Revolution or Surface Edge at U=0 For Cone or Cylinder	X	
CAM_GEO_SP_TRANSL	Spline Translated by Vector	X	
CAM_GEO_SS_EDGE	Surface Spline on Surface Edge	X	
CAM_GEO_SS_EDGE_OUT	Extract Trimmed Surface Edge as Surface-Spline	X	
CAM_GEO_SS_EDGES_OUT	Extract Chain of Trimmed Surface Edges as a Composite Curve of Surface-Splines	X	
CAM_GEO_SS_IOSF	Surface Spline at Intersection of Surface and Surface/Plane	X	
CAM_GEO_SS_OFFSET	Surface-Spline Offset From Surface-Spline	X	
CAM_GEO_SS_OUT	Extract Trimmed Surface Surface-Spline	X	
CAM_GEO_SS_PROJECT	Project Surface Spline onto Surface	X	
CAM_GEO_SS_REVSF	A Composite or a Single Surface Spline Along the Generating Curve of Surface of Revolution or Surface Edge at U=0 For Cone or Cylinder	X	
CAM_GEO_TRIM_REMALL	Removes All Inner Boundaries of Trimmed Surface	X	
CAM_GEO_TRIM_REMOVE	Removes Trimmed Surface Boundaries	X	
CAM_GEO_VE_CROSS	Vector Cross of 2 Vectors	X	
CAM_GEO_VE_FWD	Vector Along Tool Forward Direction	X	
CAM_GEO_VE_IJK	Vector IJK Components	X	
CAM_GEO_VE_IOPL	Vector Intersection of 2 Planes	X	
CAM_GEO_VE_MINUS	Vector Minus Vector	X	
CAM_GEO_VE_PEPL	Vector Perpendicular to a Plane	X	

Function	Description	CAM	CADD
CAM_GEO_VE_PLUS	Vector Plus Vector	X	
CAM_GEO_VE_PTPT	Vector Through 2 Points	X	
CAM_GEO_VE_PTSF	Vector Through a Point and Perpendicular to a Surface	X	
CAM_GEO_VE_PV	Vector From Point Vector	X	
CAM_GEO_VE_TA	Vector Along Tool Axis	X	
CAM_GEO_VE_TIMES	Vector Times Vector	X	
CAM_GEO_VE_TTCV	Vector Tangent to a Curve	X	
CAM_GEO_VE_UNIT	Unit Vector From a Vector	X	
CAM_GEO_WATERLINE	Z-level or Waterline Machining	X	
CAM_GET_ALL	Get All Entities From Secondary Unibase	X	
CAM_GET_ALL_RENAME	Get and Rename All Geometry	X	
CAM_GET_DATA	Get Variables from Secondary Unibase	X	
CAM_GET_GEO	Get Geometry From Secondary Unibase	X	
CAM_GET_GEO_RENAME	Get and Rename Geometry From Secondary Unibase	X	
CAM_GET_GEO_THRU	Get THRU Geometry From Secondary Unibase	X	
CAM_GET_GEO_THRU_RE	Get THRU and Rename Geometry From Secondary Unibase	X	
CAM_GET_LAYER	Get By Layer From Secondary Unibase	X	
CAM_GET_TYPE	Get By Type From Secondary Unibase	X	
CAM_GO	GO/ds (,ps),cs))	X	
CAM_GOBACK	GOBACK/ds,cond,cs	X	
CAM_GOBACK_IMPLIED	GOBACK/ds	X	
CAM_GODLTA_PL	Go Delta to Plane or Surface	X	
CAM_GODLTA_VE	Go Delta Along a Vector	X	
CAM_GODLTA_XYZ	Go Delta XYZ	X	
CAM_GODLTA_Z	Go Delta Along Current Tool Axis	X	
CAM_GODOWN	GODOWN/ds,cond,cs	X	
CAM_GOFWD	GOFWD/ds,cond,cs	X	
CAM_GOFWD_IMPLIED	GOFWD/ds	X	
CAM_GOFWD_LINE	GOFWD/ (LINE/FWD),cond,ds	X	
CAM_GOLFT	GOLFT/ds,cond,cs	X	
CAM_GOLFT_IMPLIED	GOLFT/ds	X	

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
CAM_GORGT	GORGT/ds,cond,cs	X	
CAM_GORGT_IMPLIED	GORGT/ds	X	
CAM_GOTO_PN	Go to Pattern	X	
CAM_GOTO_PT	Go to Point or Point-Vector	X	
CAM_GOTO_XY	Go to XY Location	X	
CAM_GOTO_XYZ	Go to XYZ Location	X	
CAM_GOUP	GOUP/ds,cond,cs	X	
CAM_IF	IF Statement	X	
CAM_INDEX	Start INDEX/COPY Range	X	
CAM_INDEX_NM	End INDEX/COPY Range	X	
CAM_INDIR	Tool Indirection Vector/Point, SRFVCT Specification	X	
CAM_INPUT_MODE	*INPUT	X	
CAM_INSERT_MODE	Insert Mode On	X	
CAM_JUMPTO	JUMPTO Statement	X	
CAM_LATHE_FINISH	Finish Lathe Shape	X	
CAM_LATHE_ROUGH	Rough Lathe Shape	X	
CAM_LOAD_PP	Load Part Program File	X	
CAM_LOOPND	LOOPND Statement	X	
CAM_LOOPST	LOOPST Statement	X	
CAM_MACRO_CALL[, "macro_name [, ONCE/MODAL]"]	Dynamic Macro Call macro_name = Macro name to call MODAL = Call form stays active until cancelled by user ONCE = Call macro once If no macro_name specified, the macro call name list form will be opened.	X	
CAM_MACRO_FORM	Allows User to Interactively Create a Custom Interface Form for a Predefined Macro	X	
CAM_MODSYS	Enable MODSYS Matrix	X	
CAM_MODSYS_NM	Disable MODSYS Matrix	X	
CAM_MOTION_CALC	Motion Calculations Form	X	
CAM_MOTION_DATA	Motion Data Form	X	
CAM_MOVE_MATRIX	Move Geometries Using Matrix	X	
CAM_MOVE_MIRROR	Move Geometries Using Mirror Matrix	X	

Function	Description	CAM	CADD
CAM_MOVE_PTPT	Move Geometries Point to Point	X	
CAM_MOVE_ROTATE	Move Geometries Using Rotation	X	
CAM_MOVE_SCALE	Move Geometries Through Scale Matrix	X	
CAM_MOVE_VECTOR	Move Geometries Using Vector	X	
CAM_MULTAX_OFF	MULTAX/OFF	X	
CAM_MULTAX_ON	MULTAX/ON	X	
CAM_NOPS	No Part Surface	X	
CAM_OVERRIDE_CHECK	Override Implied Check Surface	X	
CAM_PLAYBACK	Motion Playback Form	X	
CAM_PLAYCLIP	Playback Boundaries Form	X	
CAM_PLAYFEED	Playback Feedrate Form	X	
CAM_PLAYFILE	Motion Playback File Form	X	
CAM_PLAY_INTERP	Playback Interpolation Form	X	
CAM_PLAY_MODALS	Motion Display Modals	X	
CAM_POCKET	Advanced Pocketing	X	
CAM_POCKET_MODALS	Advanced Pocket Modals	X	
CAM_POINT_POCKET	Simple Pocketing	X	
CAM_PROFILE	Profile Machining a Curve	X	
CAM_PSIS	Define Part Surface	X	
CAM_PUT_ALL	Put All Entities into Secondary Unibase	X	
CAM_PUT_DATA	Put Variables into Secondary Unibase	X	
CAM_PUT_GEO	Put Geometry into Secondary Unibase	X	
CAM_PUT_GEO_THRU	Put THRU Geometry into Secondary Unibase	X	
CAM_PUT_GEO_TYPE	Put By Type into Secondary Unibase	X	
CAM_PUT_LAYER	Put By Layer into Secondary Unibase	X	
CAM_REDEF_CLOSE	Define Curve/Surface as Opened/Closed	X	
CAM_REDEF_FILLET	Trim Circle to 2 Lines	X	
CAM_REFSYS	Enable REFSYS	X	
CAM_REFSYS_NM	Disable REFSYS	X	
CAM_REMOVE_GEO	Remove Geometries	X	
CAM_RENAME_GEO	Rename Geometry with Source Entry	X	

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
CAM_RESET_CALL [, “PROMPT”]	Issues a *RESET/CALL Command. “PROMPT” Is an Optional Parameter. NCL Will Prompt for the Number of Calls/Loops to Back Out Of. If “PROMPT” Not Specified, Then All Calls/Loops Will BeTerminated.	X	
CAM_REVERS_OFF	REVERS/OFF	X	
CAM_REVERS_ON	REVERS/ON		
CAM_REVERSE_GEO	Reverse Direction of Geometries	X	
CAM_RMILL	Regional Mill	X	
CAM_RUN	Run Program Menu	X	
CAM_SAVE_PP	Save Part Program File	X	
CAM_SAVE_STL	Saves Selected Solids to an External STL File	X	
CAM_SCALAR_DEFINE	Display Define Scalar Form	X	
CAM_SCALARS	Display Scalar List Form	X	
CAM_SCRUB	Scrub a Surface	X	
CAM_SEQUNC_END	End Machining Sequence	X	
CAM_SEQUNC_ERASE	Erase Machining Sequence	X	
CAM_SEQUNC_LIST	List Machining Sequences	X	
CAM_SEQUNC_START	Start Machining Sequence	X	
CAM_SET_MODAL [, “type”]	Display the Settings Form. “ <i>type</i> ” Is an Optional Parameter. The General Setting Form Would Display If Not Specified. The Motion Setting Form Would Display If “MOTION” is Specified. The Geometry Setting Form Would Display If “GEOMETRY” is Specified.	X	
CAM_SHOW_SOURCE	Display Source File in Scrolling Window	X	
CAM_SKIP_LINE	Skip to Part Program Line #	X	
CAM_SMILL	Multiple Surfaces Milling	X	
CAM_SOLID_BOUND	Create an XYZ Bounding Box Around a List of Geometry or Generated Motion	X	
CAM_SOLID_BXLEN	Defines a Box Solid Using a Center Point and Its Length, Width, and Height	X	
CAM_SOLID_BXPT	Defines a Box Solid Using Opposite Corners of the Box	X	
CAM_SOLID_COMPOS	Display the Create Composite Solid Form	X	
CAM_SOLID_CONCE	Defines a Cone Solid Using a Center Point and Axis, Height, and Bottom and Top Radii	X	

Function	Description	CAM	CADD
CAM_SOLID_CONPT	Defines a Cone Solid Using Its Base and Top Points Along With the Base and Top Radii	X	
CAM_SOLID_CONTOUR	Defines an Extruded Solid That Follows the Contour Of the Part	X	
CAM_SOLID_CYLCE	Defines a Cylinder Solid Using a Center Point and Axis, Height, and a Radius	X	
CAM_SOLID_CYLCI	Defines a Cylinder Solid Using a Circle and Height	X	
CAM_SOLID_CYLPT	Defines a Cylinder Solid Using a Center Point and Axis, Height, and a Radius	X	
CAM_SOLID_LOAD	Loads Solids From an External NCL/IPV Stock File	X	
CAM_SOLID_REVOLVE	Defines a Solid of Revolution Using a Generating Curve, Center and Axis, and Optional Start and End Angles	X	
CAM_SOLID_SAVE	Saves Defined Solids to an External NCL/IPV Stock File.	X	
CAM_SOLID_SPHCE	Defines a Sphere Solid Using a Center Point and a Radius	X	
CAM_SOLID_SPHCI	Defines a Sphere Solid Using a Circle	X	
CAM_SOLID_STL	Defines a Solid By Loading an STL File	X	
CAM_SOLID_SWEEP	Defines an Extruded Solid Using a Generating Curve, Directional Vector, and a Distance	X	
CAM_SOLID_TORCE	Defines a Torus Solid Using a Center Point and Axial Vector, an Axial Radius, and a Circular Radius	X	
CAM_SOLID_TORCI	Defines a Torus Solid Using a Circle Defining Its Center, Axial Vector, and Inner Radius, and a Second Circle Defining Its Circular Radius	X	
CAM_SOLID_TORRA	Defines a Torus Solid Using a Circle Defining Its Center, Axial Vector, and Inner Radius, and a Value Defining Its Circular Radius	X	
CAM_SSRC_CURRENT	Show Source From Current Line	X	
CAM_SSRC_LINE	Show Source From Line #	X	
CAM_STAT_CUTTER	Show Current Cutter Parameters	X	
CAM_STAT_FILES	Show Current Files	X	
CAM_STAT_GEO	Show Geometry Data	X	
CAM_STAT_MODALS	Show CAM Modals	X	
CAM_STAT_THICK	Show Current Thick	X	
CAM_STAT_TOLER	Show Current Tolerance	X	
CAM_STAT_TOOL	Show Tool Position	X	

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
CAM_STATEMENT	Single NCL Command	X	
CAM_STEP_BACKWARD	Step Motion Backward	X	
CAM_STEP_BEGIN	Step the Tool to The Beginning of the Displayed Motion	X	
CAM_STEP_END	Step the Tool to The End of the Displayed Motion	X	
CAM_STEP_FORWARD	Step Motion Forward	X	
CAM_STEP_PICK	Prompts the User to Pick a Displayed Motion Segment and Step the Tool Back to this Segment	X	
CAM_SYM_CREMASTER	Create Master Symbols	X	
CAM_TERMAC	Terminate Macro Definition	X	
CAM_THICK	Open Set Machining Thicks Form	X	
CAM_TLAXIS_ATANGL	Tool Axis At Angle to Part Surface	X	
CAM_TLAXIS_CURVE	Tool Axis Through a Curve	X	
CAM_TLAXIS_FIXED	Tool Axis Fixed	X	
CAM_TLAXIS_GOUGE	Tool Axis Gouge Avoid Modifier	X	
CAM_TLAXIS_GUIDE	Tool Axis Guide Curve Modifier	X	
CAM_TLAXIS_INTERP	Tool Axis Interpolation	X	
CAM_TLAXIS_LOCK	Tool Axis Lock Mode ON	X	
CAM_TLAXIS_LOCK_END	Tool Axis Lock Mode ON at beginning of next move then OFF	X	
CAM_TLAXIS_LOCK_OFF	Tool Axis Lock Mode OFF.	X	
CAM_TLAXIS MODIFY	Tool Axis Modify	X	
CAM_TLAXIS_NORMAL	Tool Axis Normal to Part Surface	X	
CAM_TLAXIS_POINT	Tool Axis Through a Point	X	
CAM_TLAXIS_TANTO	Tool Axis Tangent to Drive Surface	X	
CAM_TLAXIS_TILT	Tool Axis Tilt Angle Modifier	X	
CAM_TLAXIS_VIEW	Output a TLAXIS Command With the Current View Normal Vector as the Tool Axis Vector	X	
CAM_TOLER	Open Set Machining Tolerance Form	X	
CAM_TOOL_COND	Tool Condition Menu, i.e. TLLFT, TLRGT, TLON, TLOFPS, TLONPS	X	
CAM_TRACUT	Enable TRACUT Matrix	X	
CAM_TRACUT_NM	Disable TRACUT Matrix	X	

Function	Description	CAM	CADD
CAM_UBFN_CLOSE	Close Secondary Unibase	X	
CAM_UBFN_INVIS_ALL	Invisible All UBFN Geometry	X	
CAM_UBFN_INVIS_GEO	Invisible UBFN Geometry	X	
CAM_UBFN_OPEN	Open Secondary Unibase	X	
CAM_UBFN_VIS_ALL	Visible All UBFN Geometry	X	
CAM_UBFN_VIS_GEO	Visible UBFN Geometry	X	
CAM_UNDO	UNDO Statement (Terminate a DO Loop)	X	
CAM_VOLUMILL	Open the VoluMill Pocket Form	X	
CAM_ZSURF	ZSURF (Zplane for Point Definition)	X	
IPV_CHANGE_VIEW	Display the NCL/IPV Change View Form	X	
IPV_COMPARE	Compare the NCL/IPV Cut Model with NCL Surfaces	X	
IPV_CUT_COLORS	Define the NCL/IPV Simulation Colors	X	
IPV_DIAGNOSTICS	Activate the NCL/IPV Crash Diagnostics Handler Form	X	
IPV_DYNAMIC_MOUSE	NCL/IPV Dynamic Viewing	X	
IPV_EXPORT_SESSION	Save the NCL/IPV Current Machining Session To a Disk File	X	
IPV_FIT_VIEW	Fit the Current NCL/IPV View	X	
IPV_FIXT_ATTRIBS	Modify the Attributes of an NCL/IPV Fixture Solid	X	
IPV_FIXT_BXLEN	Create an NCL/IPV Fixture Box Using a Center Location, Length, Width, and Height	X	
IPV_FIXT_BXPT	Create an NCL/IPV Fixture Box Using Two Corners	X	
IPV_FIXT_CONCE	Create an NCL/IPV Fixture Cone Using a Center Point, Axis Vector, Height, and an Upper and Lower Radius	X	
IPV_FIXT_CONPT	Create an NCL/IPV Fixture Cone Using Two Points and an Upper and Lower Radius	X	
IPV_FIXT_CONTOUR	Create an NCL/IPV Contour Fixture	X	
IPV_FIXT_CYLCE	Create an NCL/IPV Fixture Cylinder Using a Center Point, Axis Vector, Height, and Radius	X	
IPV_FIXT_CYLCIR	Create an NCL/IPV Fixture Cylinder Using a Circle and Length	X	
IPV_FIXT_CYLPT	Create an NCL/IPV Fixture Cylinder Using Two Points and a Radius	X	
IPV_FIXT_MODALS	Display the NCL/IPV Fixture Modals Form	X	

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
IPV_FIXT_REVOLVE	Create an NCL/IPV Fixture Surface of Revolution Using Either an NCL Surface of Revolution or a Curve and an Axis with Options Starting and Ending Angles	X	
IPV_FIXT_SOLID	Create an NCL/IPV Fixture Using an Existing Visual Solid	X	
IPV_FIXT_SPHCE	Create an NCL/IPV Fixture Sphere Using a Center Point and a Radius	X	
IPV_FIXT_PHCI	Create an NCL/IPV Fixture Sphere using the Center Point and Radius of a Defined Circle	X	
IPV_FIXT_SWEEP	Create an NCL/IPV Fixture Extrusion Using a Curve and Vector	X	
IPV_FIXT_TORCE	Create an NCL/IPV Fixture Torus Using a Center Point, Axis Vector, Axial Radius, and Circular Radius	X	
IPV_FIXT_TORCI	Create an NCL/IPV Fixture Torus Using a Circle to Define Its Center, Axis, and Inner Radius and a Second Circle to Define the Outer Radius	X	
IPV_FIXT_TORRA	Create an NCL/IPV Fixture Torus Using a circle to Define Its Center, Axis, and Axial Radius and a Value to Specify Its Circular Radius	X	
IPV_FLIP_VIEW	Flip the NCL/IPV View	X	
IPV_IMPORT_SESSION	Load a Previously Saved NCL/IPV Machining Session From a Disk File	X	
IPV_INTERACTIVE	Open the Interactive NCL/IPV Window	X	
IPV_LOAD_FIXT_STL	Load an External STL File and Use It as a Fixture Definition	X	
IPV_LOAD_MACHINE	Activate the NCL/IPV Machine Form	X	
IPV_LOAD_STOCK	Load Stock and Fixture Definitions From an External File	X	
IPV_LOAD_STOCK_STL	Load an External STL File and Use It as a Stock Definition	X	
IPV_MEASURE	Perform Measurements which Include Attributes, Point, Geometry, Thickness, Gap, Distance, and Volume	X	
IPV_MODALS	Display the NCL/IPV Modal Form	X	
IPV_PAN	Define a New Center Location for the NCL/IPV View	X	
IPV_PLAYBACK	Display the NCL/IPV Simulation Form	X	
IPV_PREVIOUS_VIEW	Restore the Previous Modals Form	X	
IPV_PRINT_SCREEN	Print Current NCL/IPV Screen to the Windows Printer	X	
IPV_REMOVE_CHIPS	Remove Stock Pieces From the NCL/IPV Machining Session	X	
IPV_REPAINT_VIEW	Redraw the NCL/IPV View	X	

Function	Description	CAM	CADD
IPV_RESET_SESSION	Restore NCL/IPV Fixtures and Stocks to Their Original Precut State	X	
IPV_RESTORE_SESSION	Restore a Previously Saved NCL/IPV Machining Session From Memory	X	
IPV_SAVE_FIXT_STL	Save Fixture Definitions in External STL File(s)	X	
IPV_SAVE_SESSION	Save a Copy of the NCL/IPV Machining Session in Memory	X	
IPV_SAVE_STOCK	Save Stock and Fixture Definitions to an External file	X	
IPV_SAVE_STOCK_STL	Save Stock Definitions in External STL File(s)	X	
IPV_SCALE_ZOOM	Scale the NCL/IPV View	X	
IPV_SECTION	Create a Sectioned View of the Part	X	
IPV_STANDALONE	Stand-alone NCL/IPV	X	
IPV_STEP_BACKWARD [,"n"]	Step Tool Backward "n" Cut. Default to One Cut if "n" not specified	X	
IPV_STEP_BEGIN	Step Tool Backward to Start of Undo Stack	X	
IPV_STEP_END	Step Tool Forward to Current Motion	X	
IPV_STEP_FACE	Allow the User to Select a Face on the Model and Steps the Tool Back to When the Face Was Cut	X	
IPV_STEP_FORWARD [,"n"]	Step Tool Forward "n" Cut. Default to One Cut if "n" not specified.	X	
IPV_STEP_MODAL	Open the NCL/IPV Undo Stack Modals Form	X	
IPV_STEP_RESET	Reset (Empty) the NCL/IPV Undo Stack	X	
IPV_STOCK_ATTRIBS	Modify the Attributes of an NCL/IPV Stock Solid	X	
IPV_STOCK_BOUND	Create an NCL/IPV Stock Box Using the Extremities of the Tool Motion	X	
IPV_STOCK_BXLEN	Create an NCL/IPV Stock Box Using a Center Location, Length, Width, and Height	X	
IPV_STOCK_BXPT	Create an NCL/IPV Stock Box Using Two Corners	X	
IPV_STOCK_CONCE	Create an NCL/IPV Stock Cone Using a Center Point, Axis Vector, Height, and an Upper and Lower Radius	X	
IPV_STOCK_CONPT	Create an NCL/IPV Stock Cone Using Two Points and an Upper and Lower Radius	X	
IPV_STOCK_CONTOUR	Create an NCL/IPV Contour Stock.	X	
IPV_STOCK_CYLCE	Create an NCL/IPV Stock Cylinder Using a Center Point, Axis Vector, Height, and Radius	X	

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
IPV_STOCK_CYLCIR	Create an NCL/IPV Stock Cylinder Using a Circle and Length	X	
IPV_STOCK_CYLPT	Create an NCL/IPV Stock Cylinder Using Two Points and a Radius	X	
IPV_STOCK_SOLID	Create an NCL/IPV Stock Using an Existing Visual Solid	X	
IPV_STOCK_SPHCE	Create an NCL/IPV Stock Sphere Using a Center Point and a Radius	X	
IPV_STOCK_SPHCI	Create an NCL/IPV Stock Sphere using the Center Point and Radius of a Defined Circle	X	
IPV_STOCK_SWEEP	Create an NCL/IPV Stock Extrusion Using a Curve and Vector	X	
IPV_STOCK_TORCE	Create an NCL/IPV Stock Torus Using a Center Point, Axis Vector, Axial Radius, and Circular Radius	X	
IPV_STOCK_TORCI	Create an NCL/IPV Stock Torus Using a Circle to Define Its Center, Axis, and Inner Radius and a Second Circle to Define the Outer Radius	X	
IPV_STOCK_TORRA	Create an NCL/IPV Stock Torus Using a circle to Define Its Center, Axis, and Axial Radius and a Value to Specify Its Circular Radius	X	
IPV_SWAP_SCREEN	Swap the NCL and NCL/IPV Windows	X	
IPV_TOOL_LIST	Display and Modify the Active Tool List	X	
IPV_TOOL_MODALS	Display the NCL/IPV Tool Modals Form	X	
IPV_TOOL_OFF	End Machining Sequence	X	
IPV_TOOL_ON	Start Machining Sequence	X	
IPV_VIEW_CENTER	Change the NCL/IPV Dynamic Viewing Center Location	X	
IPV_VIEW_NEGX	Set the NCL/IPV View Normal Vector to Negative X	X	
IPV_VIEW_NEGY	Set the NCL/IPV View Normal Vector to Negative Y	X	
IPV_VIEW_NEGZ	Set the NCL/IPV View Normal Vector to Negative Z	X	
IPV_VIEW_REFSYS	Change the NCL/IPV View to Match the Active REFSYS Matrix	X	
IPV_VIEW_SAME	Match the NCL/IPV View to the NCL View	X	
IPV_VIEW_TOOL	The Last Output Tool Axis Becomes the NCL/IPV View Normal	X	
IPV_VIEW_TRACUT	Change the NCL/IPV View to Match the Active TRACUT Matrix	X	
IPV_VIEW_X	Set the NCL/IPV View Normal Vector to Positive Y	X	

Function	Description	CAM	CADD
IPV_VIEW_Y	Set the NCL/IPV View Normal Vector to Positive Y	X	
IPV_VIEW_Z	Set the NCL/IPV View Normal Vector to Positive Y	X	
IPV_WINDOW_ZOOM	Window Zoom the NCL/IPV View	X	
CAD_CALCULATOR	NCL/CADD Calculator Mode		X
CAD_COPY_DRAG	Copy Geometries by Dragging		X
CAD_COPY_MIRROR	Copy Geometries Using Mirror Plane		X
CAD_COPY_PTPT	Copy Geometries Point to Point		X
CAD_COPY_ROTATE	Copy Geometries Using Rotation		X
CAD_COPY_SCALE	Copy Geometries by Scale Factor		X
CAD_COPY_VECTOR	Copy Geometries Using Vector		X
CAD_CREATE_FEATURE	Create Feature Labels		X
CAD_DECOMP_SYM	Decompose Symbol		X
CAD_DELETE_FEATURE	Delete Features		X
CAD_DRAFT_ARROW	Create Arrowhead on Entity		X
CAD_DRAFT_ABALL	Create Balloon		X
CAD_DRAFT_AHATCH	Crosshatch Area		X
CAD_DRAFT_ALABEL	Create Label		X
CAD_DRAFT_ANOTE	Create Notes		X
CAD_DRAFT_APOSTOL	Set Geometric Tolerances		X
CAD_DRAFT_ATEXT	Create Text		X
CAD_DRAFT_DANG	Angular Dimension		X
CAD_DRAFT_DARCL	Arc Length Dimension		X
CAD_DRAFT_DCL	Create Center Line		X
CAD_DRAFT_DDIA	Diameter Dimension		X
CAD_DRAFT_DHORIZ	Horizontal Dimension		X
CAD_DRAFT_DPTRL	Parallel Dimension		X
CAD_DRAFT_DPERP	Perpendicular Dimension		X
CAD_DRAFT_DPOL	Polar Coordinate Dimension		X
CAD_DRAFT_DRAD	Radius Dimension		X
CAD_DRAFT_DTHICK	Thickness Dimension		X
CAD_DRAFT_DVERT	Vertical dimension		X

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
CAD_DRAFT_ETEXT	Edit Text		X
CAD_DRAFT_EXIT	Exit Drafting Mode and Take Down Drafting Menu		X
CAD_DRAFT_LOAD	Load Drafting Standard		X
CAD_DRAFT_MARROW	Modify Arrowhead Attributes		X
CAD_DRAFT_MCLINE	Modify Center Line Attributes		X
CAD_DRAFT_MDATT	Modify Dimension Attributes		X
CAD_DRAFT_MDLINE	Modify Dimension Line Attributes		X
CAD_DRAFT_MDTEXT	Modify Dimension Text Attributes		X
CAD_DRAFT_MELINE	Modify Extension Line Attributes		X
CAD_DRAFT_MHATCH	Modify Crosshatch Attributes		X
CAD_DRAFT_MLEADER	Modify Leader Line Attributes		X
CAD_DRAFT_MNTEXT	Modify Note Text Attributes		X
CAD_DRAFT_MTOLER	Modify Tolerance		X
CAD_DRAFT_OARROW	Arrowhead Modals		X
CAD_DRAFT_ODIM	Dimension Modals		X
CAD_DRAFT_ODLINE	Dimension Line Modals		X
CAD_DRAFT_ODTEXT	Dimension Text Modals		X
CAD_DRAFT_OELINE	Extension Line Modals		X
CAD_DRAFT_OLEADER	Leader Line Modals		X
CAD_DRAFT_ONTEXT	Note Text Modals		X
CAD_DRAFT_OSPEC	Special Drafting Modals		X
CAD_DRAFT_OTOL	Tolerance Modals		X
CAD_DRAFT_REGEN	Regenerate Dimension		X
CAD_DRAFT_REPOSIT	Reposition Dimension		X
CAD_DRAFT_SAVE	Save Drafting Standard		X
CAD_DRAFT_SCALE	Set Drafting Scale		X
CAD_DRAFT_VIEW	Drafting Mode Viewing		X
CAD_DRAFTING	Enter Drafting Mode and Display Drafting Menu		X
CAD_DRAW_ARCHIVE	Archive Drawing		X
CAD_DRAW_CREATE	Create Drawing		X
CAD_DRAW_DELETE	Delete Drawing		X

Function	Description	CAM	CADD
CAD_DRAW_ENTER	Enter Drawing Mode		X
CAD_DRAW_EXIT	Exit Drawing Mode		X
CAD_DRAW_PLACEVW	Place View on Drawing		X
CAD_DRAW_RENAME	Rename Drawing		X
CAD_DRAW_RETRIEVE	Retrieve Drawing		X
CAD_DRAW_VIEW	Display Drawing		X
CAD_GEO_ARC_CEPTPT	Arc Center and 2 Points		X
CAD_GEO_ARC_CERA	Arc Center and Radius		X
CAD_GEO_ARC_FILLET	Single Arc Fillet		X
CAD_GEO_ARC_MFILLET	Multiple Arc Fillets		X
CAD_GEO_ARC_PTPTPT	Arc Through 3 Points		X
CAD_GEO_ARC_TANTO	Arc Tangent to 3 Lines		X
CAD_GEO_ARC_TTRA	Arc Tangent to 2 Entities and Radius		X
CAD_GEO_CI_CEPT	Circle Center and Point on Circle		X
CAD_GEO_CI_CERA	Circle Center and Radius		X
CAD_GEO_CI_CETT	Circle Center Tangent to Entity		X
CAD_GEO_CI_NCIRC	Multiple Circles with Radius		X
CAD_GEO_CI_PTPTDI	Circle Through 2 Diameter Points		X
CAD_GEO_CI_PTPTPT	Circle Through 3 Points		X
CAD_GEO_CI_TANTO	Circle Tangent to 3 Lines		X
CAD_GEO_CI_TTRA	Circle Tangent to 2 Entities and Radius		X
CAD_GEO_CM_DISOLV	Dissolve Composite Curve		X
CAD_GEO_CM_INTERP	Composite Curve Interpolate		X
CAD_GEO_CM_MERGE	Composite Curve Merge		X
CAD_GEO_CM_SPLIT	Split Composite Curve		X
CAD_GEO_CV_ABCDEF	Conic Using Coefficients		X
CAD_GEO_CV_CONTAN	Conic Through Multiple Points/Tangent-Vectors		X
CAD_GEO_CV_ELPCEN	Ellipse by Center and Axes		X
CAD_GEO_CV_ELPLIN	Ellipse Tangent to 2 Lines		X
CAD_GEO_CV_FIT	Curve Fit From Multiple Points		X
CAD_GEO_CV_HYPERB	Hyperbola		X

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
CAD_GEO_CV_INTERP	Curve From Multiple Points		X
CAD_GEO_CV_PB4PTS	Parabola by 4 Points		X
CAD_GEO_CV_PBFOCUS	Parabola by Vertex and Focus		X
CAD_GEO_CV_PBTANTO	Parabola by End Point and Tangent Vector		X
CAD_GEO_EXTR_CIRC	Rotational Extrusion		X
CAD_GEO_EXTR_LINEAR	Linear Extrusion		X
CAD_GEO_EXTR_PROJECT	Planar Projection		X
CAD_GEO_LIGHTS_MODALS	Light Modals		
CAD_GEO_LN_ATANGL	Line Through Point at Angle to a Line		X
CAD_GEO_LN_AXIS	Line Parallel to Vector		X
CAD_GEO_LN_CHAMF	Line Defined as a Chamfer		X
CAD_GEO_LN_CON	Series of Connected Lines		X
CAD_GEO_LN_MCHAMF	Line Defined as Multiple Chamfers		X
CAD_GEO_LN_PALN	Line Parallel to Line		X
CAD_GEO_LN_PTPALN	Line Through Point Parallel to a Line		X
CAD_GEO_LN_PTPE	Line Through Point Perpendicular to Entity		X
CAD_GEO_LN_PTPECV	Line Through Point Perpendicular to a Curve		X
CAD_GEO_LN_PTPT	Line Through 2 Points		X
CAD_GEO_LN_PTTA	Line Through Point Tangent to Entity		X
CAD_GEO_LN_PTTACV	Line Through Point Tangent to Curve		X
CAD_GEO_LN_TANTO	Line Tangent to 2 Entities		X
CAD_GEO_PN_CIRC	Circular Pattern of Points		X
CAD_GEO_PN_LINEAR	Linear Pattern of Points		X
CAD_GEO_PT_DELTA	Point Offset From Point		X
CAD_GEO_PT_IO	Point Intersection of 2 Entities		X
CAD_GEO_PT_ONCV	Point at U-value on Curve		X
CAD_GEO_PT_ONENT	Point on Geometry Entity		X
CAD_GEO_PT_ONSF	Point on Surface		X
CAD_GEO_PT_XY	Point XY Coordinates		X
CAD_GEO_SF_4CVC	Surface by 4 Curves Cublic Interpolation		X
CAD_GEO_SF_4CVL	Surface by 4 Curves Linear Interpolation		X

Function	Description	CAM	CADD
CAD_GEO_SF_ATTRIB	Modify Surface Attributes		X
CAD_GEO_SF_CONE	Cone Surface		X
CAD_GEO_SF_CYL	Cylindrical Surface		X
CAD_GEO_SF_FIT	Surface Fit Through Entities		X
CAD_GEO_SF_LIGHTS	Open the Light Sources Form	X	X
CAD_GEO_SF_MATERIAL	Opens the Material Properties Form	X	X
CAD_GEO_SF_MODALS	Set Surface Modals		X
CAD_GEO_SF_PLANE	Planar Surface		X
CAD_GEO_SF_REVERSE	Reverse Surface Normal		X
CAD_GEO_SF_REVOLVE	Surface of Revolution		X
CAD_GEO_SF RULED	Surface Through 2 Entities		X
CAD_GEO_SF_SPHERE	Sphere Surface		X
CAD_GEO_SF_TABCYL	Tabulated Cylinder Surface		X
CAD_GEO_SF_TORUS	Torus Surface		X
CAD_LOAD_LIGHTS	Load Light Setting Modal File	X	X
CAD_LOAD_MATERIALS	Load Materials Setting Modal File	X	X
CAD_LOAD_UNIBASE	Load Unibase File	X	X
CAD_MERGE_UNIBASE	Merge Unibase into Existing Unibase	X	X
CAD_MOD_ENTITY	Modify an Entity		X
CAD_MOVE_DRAG	Move Geometries by Dragging		X
CAD_MOVE_MIRROR	Move Geometries Using Mirror Plane		X
CAD_MOVE_PTPT	Move Geometries Point to Point		X
CAD_MOVE_ROTATE	Move Geometries Using Rotation		X
CAD_MOVE_VECTOR	Move Geometries Using Vector		X
CAD_PLACE_SYM	Place Symbol		X
CAD_PLAYBACK	Start Playback	X	X
CAD_PLOT_LOCAL	Create Drawing Plot	X	X
CAD_PLOT_PENTB	Modify Plot Pen Table File	X	X
CAD_PLOT_SETFL	Modify Plot Setup File	X	X
CAD_REMOVE_ALL	Remove All Displayed Entities		X
CAD_REMOVE_GEO	Remove Entities		X

APPENDIX E**PROGRAMMABLE FUNCTIONS**

Function	Description	CAM	CADD
CAD_RENAME_GEO	Rename Geometry		X
CAD_REVERSE_CV	Reverse Direction of Wireframe Entities		X
CAD_SAVE_UNIBASE	Save Unibase File	X	X
CAD_SCALE_GEO	Apply Scale to Geometries		X
CAD_SEL_SAVE	Selective Save of Unibase	X	X
CAD_SPLIT_CV	Split Entity into 2 Entities		X
CAD_SYM_ATTRIB	Set Display Attributes for a Symbol Instance		X
CAD_SYM_CRELIB	Create Symbol Library		X
CAD_SYM_CREMASTER	Create Symbol		X
CAD_SYM_DELLIB	Delete Symbol Library		X
CAD_SYM_DELMASTER	Delete a Master Symbol from the Master Symbol Library		X
CAD_SYM_LOADLIB	Load Symbol Library		X
CAD_SYM_RENAME	Rename Symbol		X
CAD_SYM_RENLIB	Rename Symbol Library		X
CAD_SYM_RENMASTER	Rename a Master Symbol in the Master Symbol Library		X
CAD_SYM_SAVMASTER	Archive Symbol		X
CAD_SYM_TEXTATT	Set Symbol Text Attributes		X
CAD_SYM_TEXTVIS	Set Symbol Text Visibility		X
CAD_SYM_UNLOADMAS	Remove Symbol		X
CAD_2D_MODALS	2D Geometry Analysis Modals		X
MSL_ABOUT	Display MSLITE Version Number		
0	0 Character	X	X
1	1 Character	X	X
2	2 Character	X	X
3	3 Character	X	X
4	4 Character	X	X
5	5 Character	X	X
6	6 Character	X	X
7	7 Character	X	X
8	8 Character	X	X
9	9 Character	X	X

Function	Description	CAM	CADD
.	Period Character	X	X
,	Comma Character	X	X
+	Plus Character	X	X
-	Dash Character	X	X
*	Asterisk Character	X	X
/	Slash Character	X	X
<~	Delete Character	X	X
ENTER	Enter Character	X	X

APPENDIX F: PROGRAMMABLE KEYS

NCL allows the use of keyboard hot keys or a SpaceMouse during interactive programming. The hot keys and the SpaceMouse functions are defined inside a key definition file. Following sections give an explanation of how to define these features in the key definition file.

F.1 Hot Keys Definition File

The name of this key definition file is:

C:\NCCS\NCL101\interface\nclkeys.kdf

The syntax for defining a hot key or a SpaceMouse' function in this hot key definition file is as shown below:

[^]KeyName = Function

Where:

^ - Optional parameter. If specified before the KeyName, the CTRL key must be pressed at the same time in order to activate the specified function, i.e. CTRL + Key. About half of the keys will allow this combination except those marked with a “▲”.

KeyName - The name of the actual physical key as shown in the “Key Name” column of the following table. “Upper Case” and “lower case” characters are treated as different names except when using with the CTRL key.

Function - The name of the programmable function as described in Appendix E: Programmable Functions.

Examples:

pause = CAM_PLAYBACK

Whenever the <Pause> key is pressed, the “Motion Playback” Form will be activated.

kp_enter = CAM_PLAYFILE

Whenever the KeyPad <Enter> key is pressed, the “Motion Playback File” Form will be activated.

^u = KEY_CLEAR

Whenever the <Ctrl> and the <u> keys are pressed together, the entry in the command line will be cleared.

G = NCL_GRID_OFF

Whenever the Upper Case “G” (i.e. the “G” + “SHIFT” keys if the “Caps Lock” key is off, or just the “G” key by itself if the “Caps Lock” key is on) is pressed, the Grid Pattern will be turned off.

g = NCL_GRID_ON

Whenever the lower case “g” (i.e the “G” key by itself if the “Caps Lock” key is off, or the “G” + “SHIFT” keys if the “Cap Lock” key is on) is pressed, the Grid Definition Form will be activated.

F.2

Key Names For Keyboard Hot Keys Definition

The following is a list of keyboard keys which can be programmed as hot keys for different platforms. “Key Name” is the name used to represent the actual physical key in the hot key definition file.

Key Name	Windows XP, Vista, 7, 8
f1	F1
f2	F2
f3	F3
f4	F4
f5	F5
f6	F6
f7	F7
f8	F8
f9	F9

APPENDIX F**PROGRAMMABLE KEYS**

Key Name	Windows XP, Vista, 7, 8
f10	F10
f11	F11
f12	F12
end	End
home	Home
insert	Insert
next	Page Down
pause	Pause ▲
prior	Page Up
↖ ▲	#
↙ ▲	\$
A ▲	A
B ▲	B
C ▲	C
D ▲	D
E ▲	E
F ▲	F
G ▲	G
H ▲	H
I ▲	I
J ▲	J
K ▲	K
L ▲	L
M ▲	M
N ▲	N
O ▲	O
P ▲	P
Q ▲	Q
R ▲	R
S ▲	S

Key Name	Windows XP, Vista, 7, 8
T ▲	T
U ▲	U
V ▲	V
W ▲	W
X ▲	X
Y ▲	Y
Z ▲	Z
a	a
b	b
c	c
d	d
e	e
f	f
g	g
h ▲	h
i	i
j ▲	j
k	k
l	l
m ▲	m
n	n
o	o
p	p
q	q
r	r
s	s
t	t
u	u
v	v
w	w

APPENDIX F**PROGRAMMABLE KEYS**

Key Name	Windows XP, Vista, 7, 8
x	x
y	y
z ▲	z
0	0
1	1
2 ▲	2
3	3
4	4
5	5
6 ▲	6
7 ▲	7
8 ▲	8
9	9
~ ▲	~
! ▲	!
@ ▲	@
# ▲	#
\$ ▲	\$
% ▲	%
^ ▲	^
& ▲	&
* ▲	*
(▲	(
) ▲)
_ ▲	_
+ ▲	+
{ ▲	{
} ▲	}
▲	
: ▲	:

Key Name	Windows XP, Vista, 7, 8
" ▲	"
< ▲	<
> ▲	>
? ▲	?
' ▲	'
-	-
=	=
[[
]]
\	\
;	;
,	,
.	.
/ ▲	/
kp_enter	KP Enter
kp_multiply	KP *
kp_add	KP +
kp_subtract	KP -
kp_divide	KP /
The Num Lock Key must be ON before the following keys can be used.	
kp_decimal	KP .
kp_0	KP 0
kp_1	KP 1
kp_2	KP 2
kp_3	KP 3
kp_4	KP 4
kp_5	KP 5
kp_6	KP 6
kp_7	KP 7

Key Name	Windows XP, Vista, 7, 8
kp_8	KP 8
kp_9	KP 9

F.3**Key Names For SpaceMouse's Dials And Buttons Definition**

Use the following key names in the key definition file when defining SpaceMouse functions.

dial_1_left	dial_1_right	dial_2_left	dial_2_right
dial_3_left	dial_3_right	dial_4_left	dial_4_right
dial_5_left	dial_5_right	dial_6_left	dial_6_right
button_1	button_2	button_3	button_4
button_5	button_6	button_7	button_8
button_9	button_10	button_11	

APPENDIX G: MISCELLANEOUS

G1 Save And Load Session

NCL has the ability to save an interactive session to disk so that it can be restarted at another time. All appropriate settings are saved, as well as the current Unibase, Part Program, Clfile, Layout, and **NCL/IPV** simulation.

Saving an **NCL** session requires many external files, requiring it to be saved in its own directory. The user will be prompted for the directory name when saving a session and if the directory already exists, then it will be checked to make sure that it already contains an **NCL** Session. If it does not, then an error message will be output.

All settings and files that are important to the processing of the **NCL** program are saved and restored with the session, but not all “attribute” style settings are saved. These settings mainly include default display settings that are used for visualization.

The “Save Session” task can be accomplished by using the following interface menu sequence.

File > Save Session

The “Load Session” task can be accomplished by using the following interface menu sequence.

File > Load Session

Note: Saved session from 64 bits version of **NCL** cannot be loaded back in 32 bits version of **NCL** or vice versa.

G2 Autosave Part Program or Unibase

NCL has the ability to autosave a part program or the unibase at specified times/changes during an **NCL** session. The part program will be saved to a new to a new temporary file with each automatic save and will use the following name convention:

%ind_filename.ext

“ind” refers to the version number of the saved file, which starts at the lowest available number and increments with each automatic save. Note that existing files will not be overwritten so the version number may be incremented multiple times to find the next available file name.

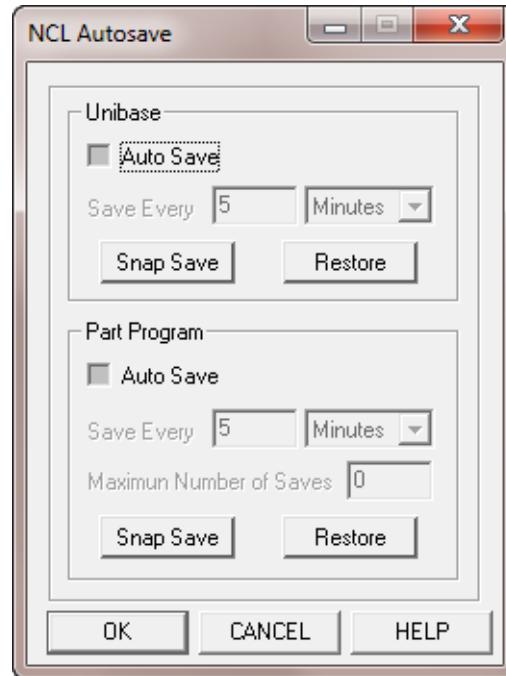
“filename” refers to the name of the part program currently loaded in **NCL** and will be the same used for the temporary work file.

“ext” refers to the file extension of the part program currently loaded in **NCL** and will be the same used for the temporary work file

This function can be activated/deactivated by using the following interface menu sequence

File > Autosave

to bring up the form as shown below.



This form has two sections: Unibase and Part Program.

Unibase**Auto Save**

When selected, **NCL** will automatically save the Unibase when the Autosave condition has been met. Selecting this box also enables the Autosave condition fields. The condition can be:

- Minutes: Autosave the Unibase after the given number of minutes have passed.
- Changes: Autosave the Unibase after the given number of changes have been made.

Snap Save

Press this button to manually generate an Autosave Unibase file. Autosave does not need to be enabled to perform a Snap Save.

Restore

Press this button to restore the Autosave Unibase as the active model.

Part Program**Auto Save**

When selected, **NCL** will automatically save the part program when the Autosave condition has been met. Selecting this box also enables the Autosave condition fields. The condition can be:

- Minutes: Autosave the part program after the given number of minutes have passed.
- Changes: Autosave the part program after the given number of changes have been made.

Maximum Number of Saves

Enter the maximum number of Autosave part program files that can be saved in the current **NCL** session. Once the maximum is reached, the oldest Autosave file will be deleted and the new Autosave file will be created. Valid responses are 1 to 100.

Snap Save

Press this button to manually generate an Autosave part program file. Autosave does not need to be enabled to perform a Snap Save.

Restore

Press this button to restore an Autosave part program as the active part program. A file browser will be displayed requesting the name (version) of the saved part program file to load.

OK:

Click this button to accept the changes and close the form.

Cancel:

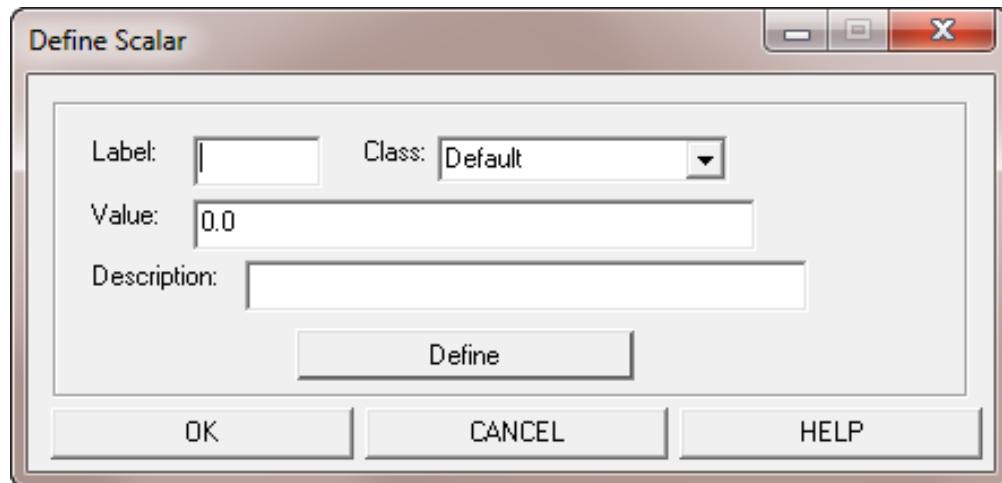
Click this button to cancel all the changes and close the form.

Help:

Click this button to open the online help for this form.

G.3**Define Scalar Variables With Interface Form**

NCL users can define scalar variables with an interface form. The following shows the graphical interactive interface to define a scalar variable and a description of how to use this interface. This interface can be activated by using the following icon sequence:



Label:

Enter the name of the scalar variable to be defined or modified. If a defined scalar variable label is entered, all the fields will be updated with the current settings automatically.

Class:

Enter the name of the class to be assigned to this scalar variable if the class has not been defined. Otherwise, click on the down arrow to pick an existing class. The “Default” class is same as not assigning any class to the corresponding scalar variable. The class string can be up to 20 characters.

Value:

Enter the scalar value or number assigned to the scalar variable. Arithmetic expression and previously defined scalar variable are allowed.

Description:

Enter an optional description string assigned to the scalar variable. The description string can be up to 64 characters.

Define:

Click this button to accept the form and output the scalar definition and optional [PROMPT/SCALAR](#) statement without closing the form.

OK:

This button is similar to the Define button except the Define Scalar form will be closed.

Cancel:

Click this button to cancel all the changes and close the Define Scalar form.

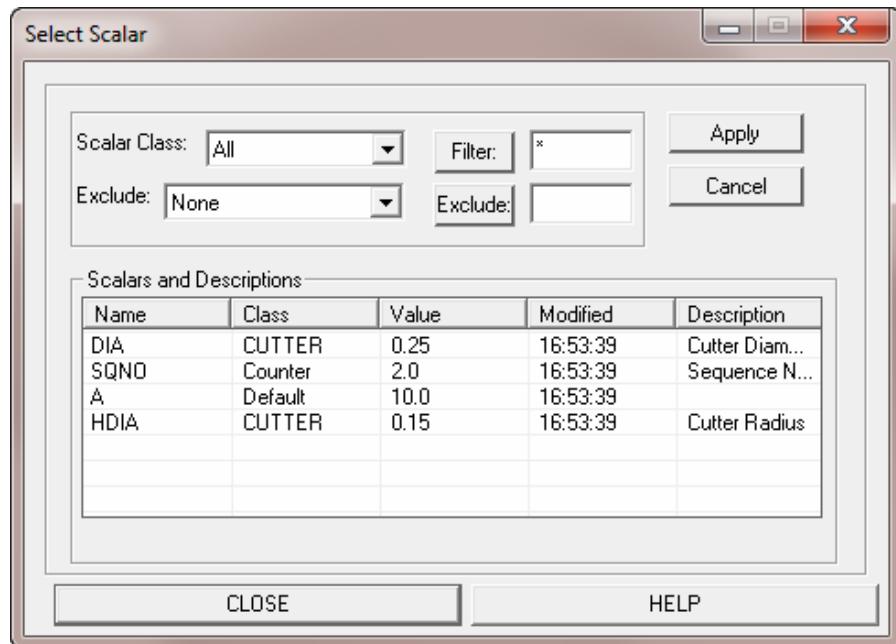
Help:

Click this button to open the online help for this form.

G.4

Select A Scalar Variable With Interface Form

NCL users can select a defined scalar variable with interface form. This form can be activated anytime by using the hot key “**Cntl + s**”. Unlike other forms, this form can be accessed while a form is active. Following shows an example of selecting a scalar variable by highlighting it from a list of all defined scalar variables displayed in the Scalars and Descriptions window of the Select Scalar interface form.



The Apply button will copy the scalar selected in the list to the active text field (form field, command prompt) without closing the form. This is similar to the Close button, except that the Select Scalar form will remain active.

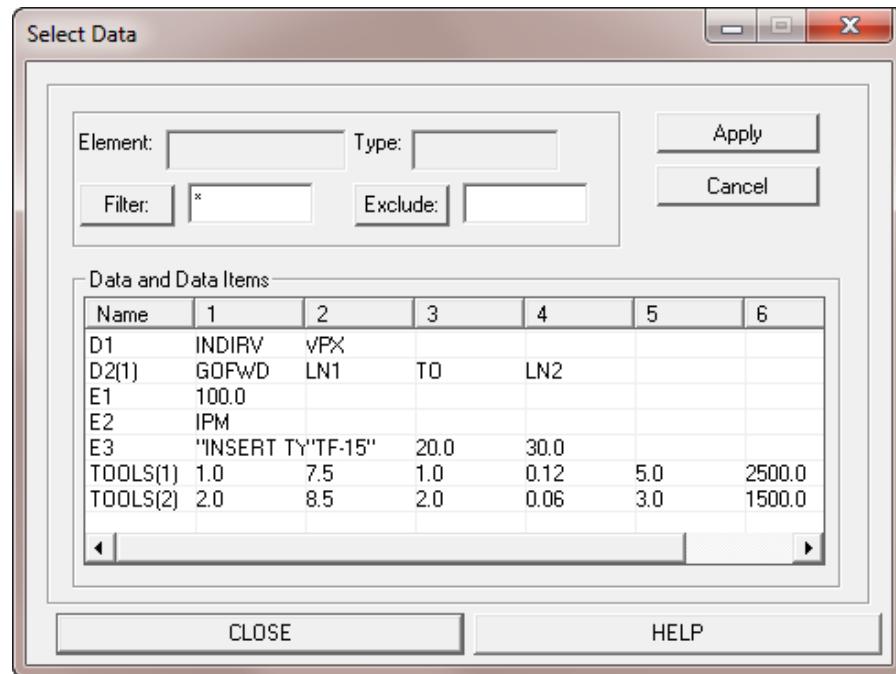
The Scalar Class and associated Filter fields determine which scalars are displayed in the list. The Exclude and associated Filter fields determine which scalars are not displayed in the list.

The Scalars and Descriptions list is used to both display the defined scalars and as a selection field to pass a scalar name into a form field or command prompt. Pressing any of the column headings will sort the list by the contents of the associated column. Pressing the column button a second time will sort the list in reversed order. The size of the columns can be changed by simply clicking on the column separator bars and dragging them to their new size.

When this form is accepted (closed by clicking the *CLOSE* button), the selected scalar will be appended to the current command line if the command prompt is opened, or it will be appended to or replaced the active form field text depending on the form field data type.

G.5**Select A Data Statement Element With Interface Form**

NCL users can select a defined DATA statement element with interface form. This form can be activated anytime by using the hot key “***Cntl + D***”. “**D**” must be in upper case. Unlike other forms, this form can be accessed while a form is active. Following shows an example of selecting a DATA statement element by highlighting it from a list of all defined DATA statements displayed in the Data and Data Items window of the Select Data interface form.

**Element:**

Displays the currently selected data element's label as it will be output to the command line or a form field. This is a Read-Only field.

Type:

Displays the entity type of the data element. The data element type can be a geometric entity (Point, Line, etc.), Vocabulary word, Value, or Text string. This is a Read-Only field.

Filter:

This field is used to filter the data displayed in the 'Data and Data Items' field. Press the Filter button to filter the list.

Exclude:

This field is used to filter the data displayed in the 'Data and Data Items' field. This is an opposite filter to the "Filter", in such that any Data belonging to the filter in this field will NOT be displayed. Press the Exclude button to filter the list.

Apply

Apply will send the selected data element to the active text field without closing the form.

Cancel

Cancels the form without selecting a data element.

Data and Data Items

This list can be used to simply view the defined Data statements or you can select an individual Data element from the list and send it to the active command line or form field by pressing the "*Apply*" or "*Close*" button.

When the Apply or Close button is pressed, the selected data will be sent to the current active text field, which is a command window or an active form field. In the case of the Command Window, the reference to the DATA statement "*D1[I]*" will be inserted in the line and not the value shown in the form. In the case of sending the data to the active form field, it depends on the form field's type, if the form field type is a text string, then the reference to the DATA statement element "*D1[I]*" will be inserted in the form field, otherwise the data item value will be used.

Close

Selects the data element and closes the form.

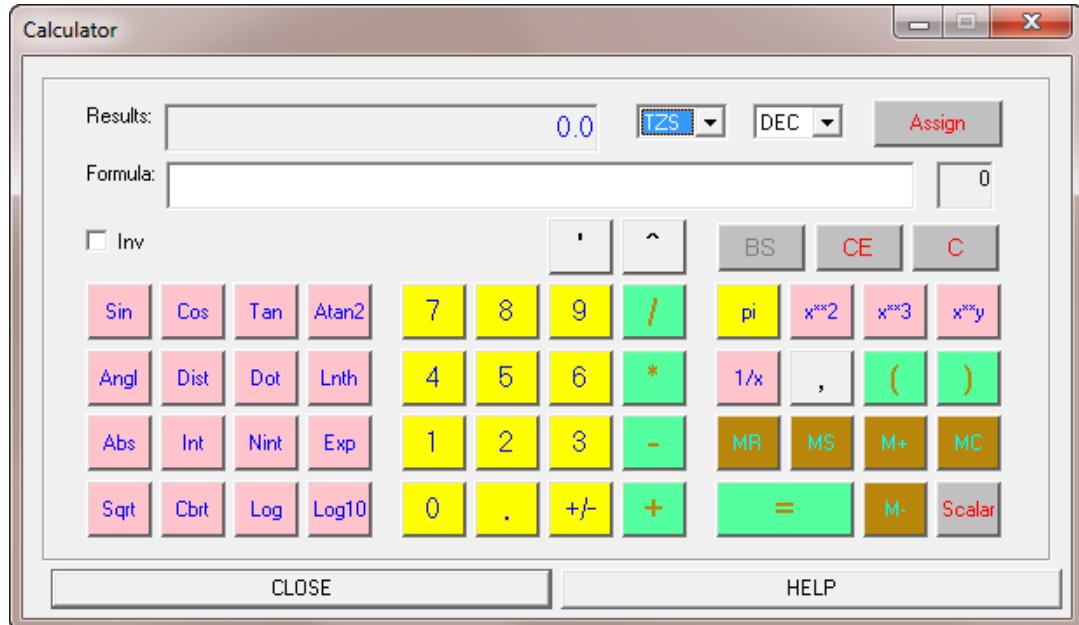
Help

Click this button to open the online help of this form.

G.6 NCL Visual Calculator

The **NCL** Visual Calculator can be activated by clicking the icon menu  .

Following shows the graphical interactive interface of this Visual Calculator, its functionality and a description of how to use this calculator.

**Results**

The *Results window* is the standard display of a calculator. It contains the numeric input/calculation of the calculator functions. This is a display only field and cannot be modified directly by the user. The result will always be a

current result or current input value. When multiple values are entered, it always displays the last input value.

Precision Choice Box

This *choice box* determines the accuracy of the value displayed in the *Results window*. TZS will display the values using trailing zero suppression. Up to 8 digits after the decimal point can be displayed, but trailing zeros will be removed. For example, 1.0000 will display as 1.0, 91.000012348 will display as 91.00001235. The other choices (1 to 8) will display a fixed number of digits after the decimal point including trailing zeros. For example, when ‘4’ is selected as the choice, then (1.0 will be displayed as 1.0000, 91.000012348 will be displayed as 91.0000).

Display Format Choice Box

This *choice box* contains the choices of *DEC* and *DMS*. *DEC* will display the standard decimal notation for the results field while *DMS* will display the results in degrees, minutes, seconds using the following format: deg’m^sec. In *DMS* format, there are 60 seconds per minute and 60 minutes per degree. The only value that can have a decimal point is the seconds value.

Assign

When in *choice mode*, the *Assign* button will assign the generated formula to a scalar by issuing a standard **NCL** assignment command, such as $A=3/12*\sin(15)$. When the *Assign* button is pressed, then the formula will be finalized as if the = key was pressed and the *Define Scalar form* will be activated with the *Value* field defaulting to the *Formula window*. The scalar will actually be defined from the *Define Scalar form*.

When in command/prompt mode or when another form is active, the *Assign* key will act as an accept button, it will assign the formula/result to the *command window* or a form field and then automatically close the form.

Formula Window

The *Formula window* contains the formula created by the user interaction with the calculator. The user may also manually type data into this window and when this field is activated either by using the <Enter> key or by exiting the field, the formula will be checked for validity, processed, and the answer will be displayed in the *Results window*.

The following special rules must be followed when building the actual formula.

The *Trig* (*Sin*, *Cos*, *Tan*, *Atan2*) and *Numeric* (*Abs*, *Int*, *Nint*, *Exp*, *Sqrt*, *Cbrt*, *Log*, *Log10*) functions will be applied to the value that is currently displayed in the *Results window*, but will be output to the *Formula window* at the front of the appropriate part of the equation. For example, the following shows the window displays when if the corresponding sequences are entered.

Input: 3 * 2 + 4 Sin
Results: .069756
Formula: 3*2+SIN(4)

Input: 3 * 2 + 4 = Sin
Results: .173648
Formula: SIN(3*2+4)

The *Geometry functions* (*Angl*, *Dist*, *Dot*, *Lnth*) do not use the value in the *Results window*, but rather require geometry to be selected, therefore the output of these functions to the *Formula window* is sequential in nature.

Input: 3 * 2 + Dist (Pick PT1 and PT2) =
Results: 7.0
Formula: (3*2+DIST(PT1,PT2))

The +/- key will change the sign of the value in the *Results window*, but will act differently in the *Formula window*, depending on the key sequence entered prior to the +/- key. The 1/x key works in a similar manner.

Input: 3 * 2 +/ - + 4 =
Results: -2
Formula: (3*-2+4)

Input: 3 + 4 = +/1 *2 =
Results: -14
Formula: (-(3+4)*2)

Input: 45 1/x sin
Results: .000388
Formula: SIN(1/(45))

Status Window

The *Status Window* is used to display the open parenthesis count in the format (n), where n is the number of parenthesis levels currently open.

Inv

When the *Inv* box is checked, the *trig* keys (*Sin*, *Cos*, *Tan*) will be assigned to their respective *inverse functions* (*Asin*, *Acos*, *Atan*). The text of the trig function keys will also be changed to match the inverse function text. When this box is not checked, then the normal trig functions will be active.

In addition, the *Dist* key will be changed to a *Tdist* key when the *Inv* box is checked.

Backspace - BS

The *BS* key deletes one digit from the number currently being input. For example, if you input *123.45* and then press the *BS* key, it will become *123.4*. It does not apply to other operation/function input, for example, if you input *123.45+*, then the *BS* key will be disabled. If you want to undo “+”, then use the *CE* button.

Clear Entry - CE

The *Clear Entry* key will act as an Undo function and cancel/erase the last interaction. The *CE* key can only undo the last entered value/operation/function, pressing it multiple times will have no effect pass the first press. Following is a list of the procedures that will be taken upon pressing the *CE* key depending on the last key sequence entered.

Input:	Function key
Results:	Previous value prior to pressing the function key.
Formula:	The function is removed and the formula restored to its previous state.
Input:	Number, pi, Scalar
Results:	Previous value prior to entering the number.
Formula:	Previous format before entering "Number, pi, Scalar"
Input:	+/- or 1/x

Results:	Previous value prior to pressing this key.
Formula:	If the formula was changed, then it is reset to its previous state.
Input:	Operator
Results:	Previous value prior to entering the operator.
Formula:	The operator is removed.
Input:	x^2, x^3, x^y
Results:	Previous value prior to pressing this key.
Formula:	The exponent is removed from the formula.
Input:	Comma key
Results:	Comma is removed.
Formula:	The Comma is removed.
Input:	Open parenthesis
Results:	No change.
Formula:	Open parenthesis is removed.
Input:	Close parenthesis
Results:	Previous value prior to pressing this key.
Formula:	Close parenthesis is removed.
Input:	Equals
Results:	Previous value prior to pressing this key.
Formula:	Previous format.

Clear - C

The *Clear* key will reset the *Results window* back to 0 and clear the *Formula window*. It will not clear the memory value. Use the *MC* key to clear the memory value.

Trig Functions - Sin, Cos, Tan, Atan2

The trig function keys perform the standard trig function on the value currently displayed in the *Results* field. If the Inv box is checked, then the *Sin, Cos, Tan*

buttons will display *Asin*, *Acos*, *Atan* respectively and will perform the inverse trig functions.

The *Atan2* function requires two numeric values. Therefore, the input will be two values separated by a comma. The Results window only display the last entered number. For example:

Input:	-5, 1.0 Atan2
Results:	"-5.0", then "0.0" (after enter comma), then "1.0", then "-78.6900675"
Formula:	ATAN2 (-5,1.0)

Geometry Functions - Angl, Dist, Dot, Lnth, Tdist

The geometry functions require the user to select one or more geometric entities from the screen. Pressing one of these keys causes the calculator form to be hidden and the user will be prompted to select one or 2 entities. Once the entities are selected, the calculator form will be redisplayed and the result of the function (with the input as selected) will be displayed in the *Results window* and the geometry function will be added to the formula.

Input:	Lnth, the select LN1
Results:	3.915806
Formula:	LNTHF (LN1)

When prompted for selection, only the correct types of geometry can be selected. For example, many types of geometry can be selected as the first parameter of the *Dist* function, but if a Line is selected as the first entity, then a Line must be selected as the second entity.

The *Tdist* function can be used to find the distance between a point-vector a line, circle, curve, plane, surface or solid. After selecting the geometry a popup menu will be displayed requesting which distance to report. The following distances can be reported.

- | | |
|----------|---|
| Closest | - All intersections between the point-vector and entity will be evaluated. The distance between the starting point of the point-vector and the closest intersection of the entity will be returned. |
| Farthest | - The distance between the starting point of the point-vector and the farthest intersection of the entity will be returned. |

Near Point - The distance between the starting point of the point-vector and the intersection of the entity closest to the selected near point will be returned.

The first entity selected for the *Tdist* function must be a point-vector.

Numeric Functions - Abs, Int, Nint, Exp, Sqrt, Cbrt, Log, Log10

The *numeric function* buttons perform the standard numeric function on the value currently displayed in the *Results field*.

Input:	2 Exp
Results:	7.389056
Formula:	EXPF(2)

Numeric Keypad - 7, 8, 9, 4, 5, 6, 1, 2, 3, 0, ., ', ^

The *Numeric Keypad* is used for entering input values as on a standard calculator.

Plus / Minus Key - +/-

The *+/-* key will change the sign of the value currently in the *Results window*.

Operation Keys - /, *, -, +

The *operation keys* perform the following operations on the values entered prior to and after the *operation key*.

/	Division
*	Multiplication
-	Subtraction
+	Addition

pi

The *pi* key will simply enter the value of pi into the *Results window*, the same as if the digits were entered individually. If a number is currently being input (the last key pressed was a number key), then this number will be replaced with pi.

Exponent Keys - x^2 , x^3 , x^y

The *exponent keys* will add an exponent to the current value. The x^2 key will square the value, x^3 will cube the value, and x^y will use the next value as the exponent.

Inversion Key - $1/x$

The *inversion key* will divide 1.0 by the number displayed in the *Results window*. X=0 is not allowed.

Comma Key

Enters a comma in the *Results window* and is used for entering values for functions that require two values such as the *Atan2* function.

Parenthesis Keys - ()

The *Open parenthesis* key will start a new parenthesis level. The level number of open parenthesis will be displayed in the *Status Window*.

The *Close parenthesis* key will end the current parenthesis level. The parenthesis level count in the Status Window will be decremented by 1. This key will be disabled unless there is at least one open level of parenthesis.

When a parenthesis level is active the *Assign* and = keys will be disabled.

When *Close parenthesis* key is entered, it will calculate the sub-total and display it in the *Results window*. For example:

Input:	(1+(2+3)+4)
Results:	'1' '2' '3' '5' (which is 2+3) '4' '10' (which is 1+(2+3)+4)
Status:	'0' '2' '1' '0'
Formula:	(1+(2+3)+4)

Memory Keys - MR, MS, M+, MC, M-

The *Memory keys* are used to access the single memory location of the calculator. The *MS* key will store the value currently in the *Results window* in memory. *M+* will add the current value to the memory value and *M-* will subtract the current value from the memory value.

The *MR* key will enter the memory value into the *Results window*, the same as if the digits were entered individually. If a number is currently being input (the last key pressed was a number key), then this number will be replaced with the memory value.

Equals Key - =

The *Equals key* will finalize the formula and display the final answer in the *Results field*. The formula can still be built on after pressing the *=* key, just like on a physical calculator. If the formula is continued after the *=* key is pressed, then the currently defined formula will be encapsulated within parenthesis.

Scalar Key

The *Scalar key* will bring up the *Select Scalar form*. Once a scalar is selected, its value will be displayed in the *Results window* and the scalar added to the formula.

Operation

The **NCL** Visual Calculator can be brought up during *Choice mode, Command mode, Prompt mode, and while in a form field*. The followings define how the calculator acts during the various modes.

Choice Mode

The Calculator will behave exactly as described as above. The *OK* and *CANCEL* buttons will cancel the form without any other output.

Command Mode

The *Assign* key here acts as an “Accept” button. Pressing the *Assign* key will place the formula as defined into the *command line* at the current cursor location. The *Results* value will be ignored. The *OK* and *CANCEL* buttons will close the calculator without outputting anything to the *command line*.

Prompt Mode

The *Assign* key here acts as an “Accept” button. Pressing the *Assign* key will place the *Results* value in the *prompt field* overriding any input that is currently in the field. The *OK* and *CANCEL* buttons will close the calculator without outputting anything to the *prompt field*.

Form Field

Bringing up the Calculator while in a *form field* will behave exactly the same way as in *Prompt Mode*.

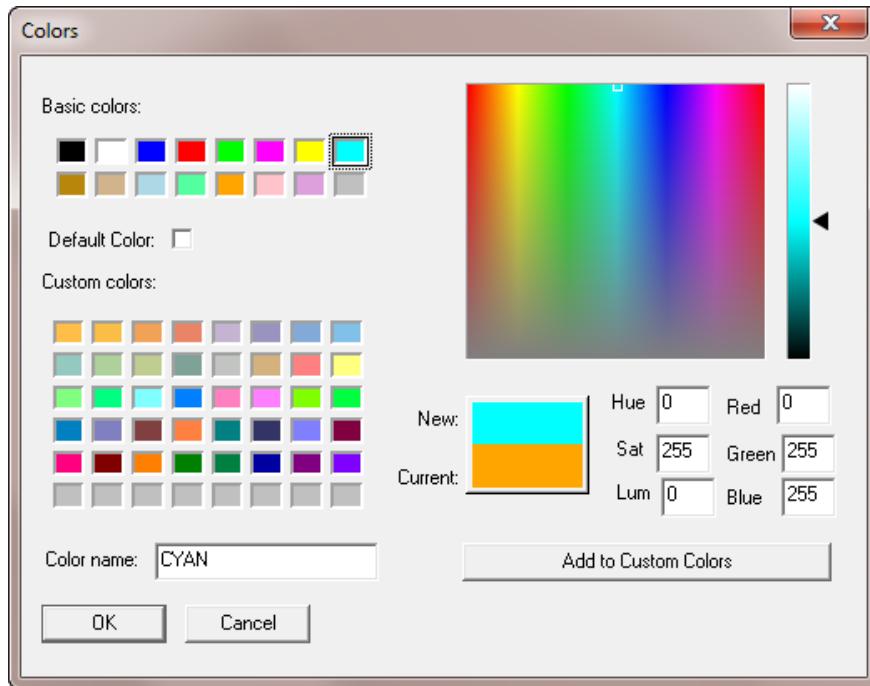
Shortcut Keys

The following shortcut keys are available while in the calculator.

<code>kp_1 -> 1</code>	<code>kp_2 -> 2</code>	<code>kp_3 -> 3</code>	<code>kp_4 -> 4</code>
<code>kp_5 -> 5</code>	<code>kp_6 -> 6</code>	<code>kp_7 -> 7</code>	<code>kp_8 -> 8</code>
<code>kp_9 -> 9</code>	<code>kp_0 -> 0</code>	<code>kp_divide -> /</code>	<code>kp_multiply -> *</code>
<code>kp_subtract -> -</code>	<code>kp_add -> +</code>	<code>kp_enter -> =</code>	

G.7 Color Form

The Color Form allows the user to change the displayed color. Clicking the color button of any form will bring up the following Color Form.



The user can define a new custom color by setting the desired RGB values or by choosing a color from the palette, typing in the name of the color in the “Color name” field and then clicking the “Add to Custom Colors” button. This will display the newly defined color in the “Custom colors” section and add the newly

defined color to the “ncl_color.mod” file in the folder defined by the “UU_USER_SETTINGS” in the “nccs.init” file.

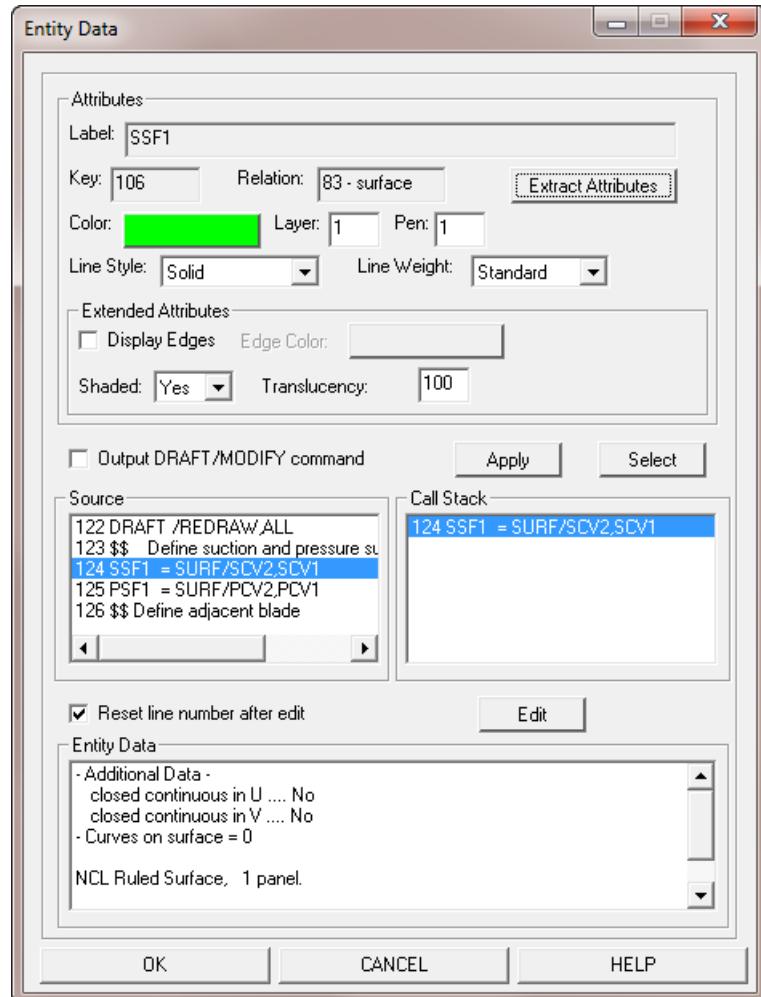
The “Default Color” check box will not appear on the color form if there is no “Default Color Setting” for the corresponding item.

G8 Geometry Entity Data Query And Edit Tool

The Entity Data Query and Edit Tool form can be activated by clicking the following menu sequence.

Status > Entity Data > Pick the Geometry Entity

Following shows the graphical interactive interface of this Entity Data form, its functionality and a description of how to use this form.



Label

This is a read only field that displays the label of the geometric entity.

Key

This is a read only field that displays the Unibase key of the geometric entity.

Relation

This is a read only field that displays the geometry type of the entity. Defines the type of light being defined.

Extract Attributes

Pressing the Extract Attributes button allows the attributes of a picked entity to be used to replace the attributes of the current entity. The color, layer, pen, line style and line weight will be extracted. Further, additional surface attributes can be extracted from a surface or solid and applied if the current entity is a surface or solid. In addition, point or pattern marker types can also be extracted and assigned to the current entity.

Color

The entity is displayed using the color appears here. The displayed color can be changed by clicking this Color button to bring up the Color form. This button is disabled if the picked entity is of the type Drafting or Symbol Instance.

Layer

Defines the layer that the geometric entity resides on. This field is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Pen

Defines the logical pen used for the entity when a plot is created. This field is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Line Style

Defines the line style (solid, dashed, etc.) to display the entity with. This field is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Line Weight

Defines the line weight (standard, heavy, etc.) to display the entity with. This field is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Display Edges

Check this box to display edges for Surface and Solid type entities.

Edge Color

The Edges is displayed using the color appears here. The displayed color can be changed by clicking this Color button to bring up the Color form.

Shaded

Toggles the display of the selected surface or solid between shaded and wireframe modes.

Translucency

Sets the translucency (opacity) of the selected surface or solid. A value of 100 will display a solid surface with lesser values displaying a more see through surface or solid.

Output DRAFT/MODIFY command

When this box is checked a DRAFT/MODIFY command will be output to the part program containing the attribute settings that have changed when the Apply or OK button is pressed. Unchecking this box will update the attributes of the entity without outputting a command.

This box is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Apply

Pressing the Apply button will assign any attributes that have been changed in this form to the geometric entity without closing the form. This button is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Select

Selects a new geometric entity display the attributes for. If any changes were made to the current entity's attributes, they will NOT be set prior to selecting a new entity, you must first press the Apply button and then the Select button for these changes to take effect.

Source

Displays the source code that generated the geometric entity. If the entity was not created with a source command, then no source will be displayed.

Call Stack

Displays the active call/loop stack when the geometric entity was created. The call/loop stack consists of the actual command that generated the entity and any Macro calls or DO loops that were active at the time of the entity's creation.

Macro calls will display simply as CALL/macro and DO loops will be displayed as DO/lable,index=n, where 'index' is the name of the controlling DO loop variable and 'n' is its value at the time that the geometric entity was created.

Selecting a line from the Call Stack list will display the associated source lines in the Source list.

Reset the line number after edit

Checking this box causes **NCL** to reset the current source line number to the line that was active prior to displaying the Entity Data form. If this box is not checked, then the current source line will be the line that was active when Command Mode was exited after pressing the Edit button.

Edit

Pressing the Edit button when there is source code displayed will enter Command Mode at the line that is active in the Source list.

Entity Data

Displays the canonical data associated with the selected geometric entity.

OK

Click this button to accept the changes, close the form and output the corresponding DRAFT statement if the “Output DRAFT/MODIFY command” button is checked and at least one of the fields has been changed.

CANCEL

Click this button to cancel the changes and close the form.

HELP

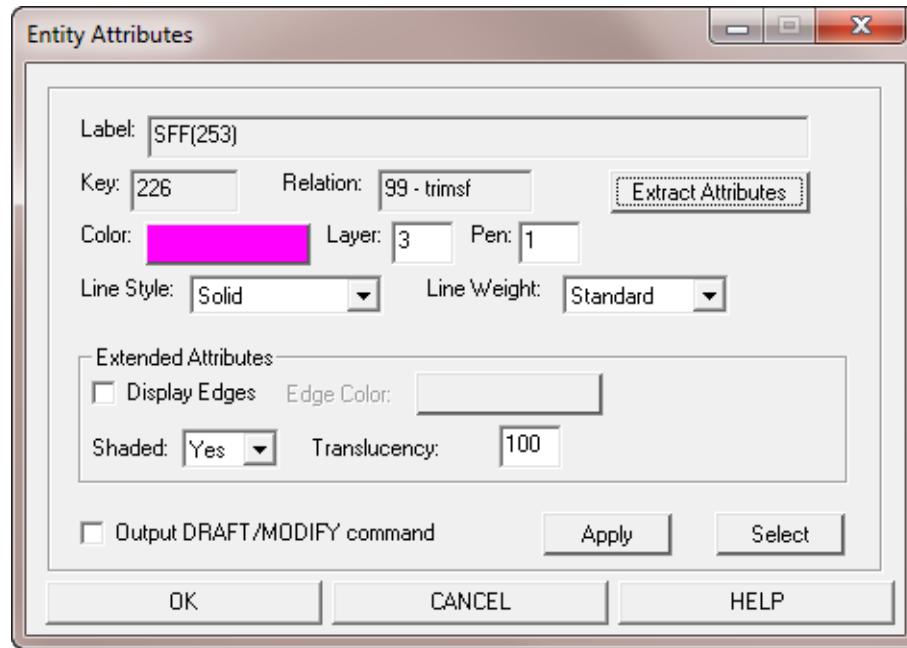
Click this button to open the online help of this form.

G9**Geometry Entity Attributes Query And Edit Tool**

The Entity Attributes Query and Edit Tool form can be activated by clicking the following menu sequence.

Status > Entity Attrib > Pick the Geometry Entity

Following shows the graphical interactive interface of this Entity Attributes form, its functionality and a description of how to use this form.



Label

This is a read only field that displays the label of the geometric entity.

Key

This is a read only field that displays the Unibase key of the geometric entity.

Relation

This is a read only field that displays the geometry type of the entity. Defines the type of light being defined.

Extract Attributes

Pressing the Extract Attributes button allows the attributes of a picked entity to be used to replace the attributes of the current entity. The color, layer, pen, line style and line weight will be extracted. Further, additional surface attributes can be extracted from a surface or solid and applied if the current entity is a surface or solid. In addition, point or pattern marker types can also be extracted and assigned to the current entity.

Color

The entity is displayed using the color appears here. The displayed color can be changed by clicking this Color button to bring up the Color form. This button is disabled if the picked entity is of the type Drafting or Symbol Instance.

Layer

Defines the layer that the geometric entity resides on. This field is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Pen

Defines the logical pen used for the entity when a plot is created. This field is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Line Style

Defines the line style (solid, dashed, etc.) to display the entity with. This field is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Line Weight

Defines the line weight (standard, heavy, etc.) to display the entity with. This field is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Display Edges

Check this box to display edges for Surface and Solid type entities.

Edge Color

The Edges is displayed using the color appears here. The displayed color can be changed by clicking this Color button to bring up the Color form.

Shaded

Toggles the display of the selected surface or solid between shaded and wireframe modes.

Translucency

Sets the translucency (opacity) of the selected surface or solid. A value of 100 will display a solid surface with lesser values displaying a more see through surface or solid.

Output DRAFT/MODIFY command

When this box is checked a DRAFT/MODIFY command will be output to the part program containing the attribute settings that have changed when the Apply or OK button is pressed. Unchecking this box will update the attributes of the entity without outputting a command.

This box is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Apply

Pressing the Apply button will assign any attributes that have been changed in this form to the geometric entity without closing the form. This button is grayed out if the picked entity is of the type Drafting or Symbol Instance.

Select

Selects a new geometric entity display the attributes for. If any changes were made to the current entity's attributes, they will NOT be set prior to selecting a new entity, you must first press the Apply button and then the Select button for these changes to take effect.

OK

Click this button to accept the changes, close the form and output the corresponding DRAFT statement if the “Output DRAFT/MODIFY command” button is checked and at least one of the fields has been changed.

CANCEL

Click this button to cancel the changes and close the form.

HELP

Click this button to open the online help of this form.

G.10**Geometry Entity Measuring Tool**

The Geometry Entity Measurement form can be activated by clicking either the following menu sequence.

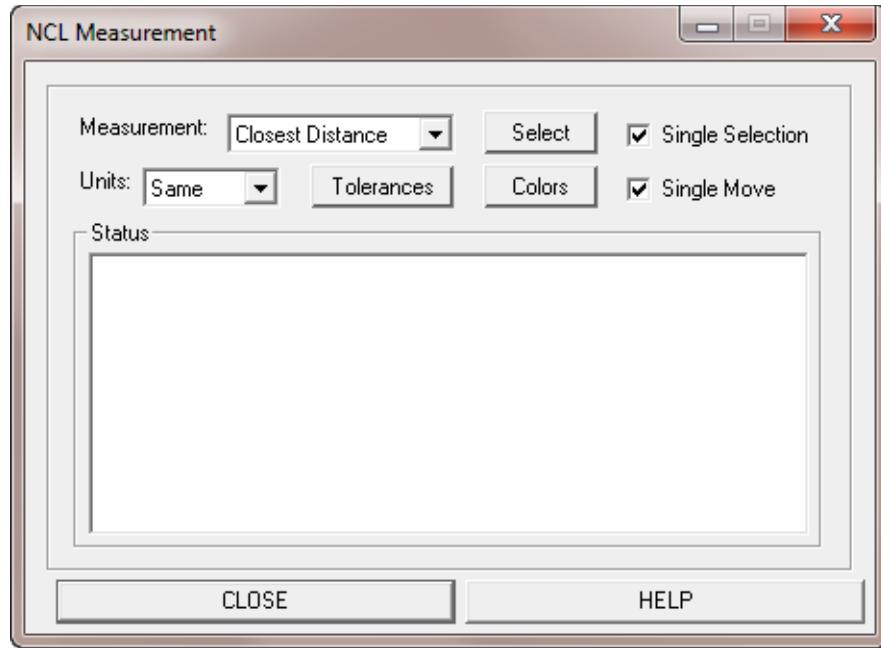
Status > Measure

or



> **Measure**

Following shows the graphical interactive interface of this Measurement form, its functionality and a description of how to use this form.

**Measurement:**

To measure the closest distance between two entities, select "Closest Distance", press the Select button, and then pick any two geometric entities. The minimum distance between the two entities and the angle between the normal/tangent vectors will be displayed in the Status window. Picking a curve will use the tangent vector at the location and picking a surface will use the normal vector.

Selecting "Picked Distance" will measure the entities between at the locations that they were picked. The distance between the two locations and the angle between the normal/tangent vectors at these locations will be displayed in the Status window.

For distance measurements, the tangent vector of a curve at the selected location or the normal vector of a surface will be used. The distance between the two locations will be displayed, unless one of the entities is a vector. If one of the selections is a point, then only the distance will be displayed.

To measure the radius of curvature of an entity, select "Radius", press the Select button, and then pick any geometric entity. The radius of curvature of the entity and the arc length used in the calculation will be displayed in

the Status window. For a surface, the values in both the U and V direction will be displayed.

Select

The Select button enters picking mode and allows you to select the geometric entity(ies) or motion to measure. The results will be displayed in the Status window after the selections are made and a visual representation of the geometry used to make the measurements will be displayed in the graphics window.

Note that the default picking type will be geometric entities and so to pick motion the **Alt-Act** button must be pressed. To switch back to geometric entity picking, press the **Alt-Act** button again.

Single Selection

Normally, the **NCL** Measurement form will remain displayed after pressing the *Select* button and the selection process will be modal, which means after a selection is made, the results will be displayed and you will remain in selection mode until you hit either **Done** or **Reject**.

If the **Single Selection** box is checked, then the form will go away when selection mode is entered and will be redisplayed after the selection is made.

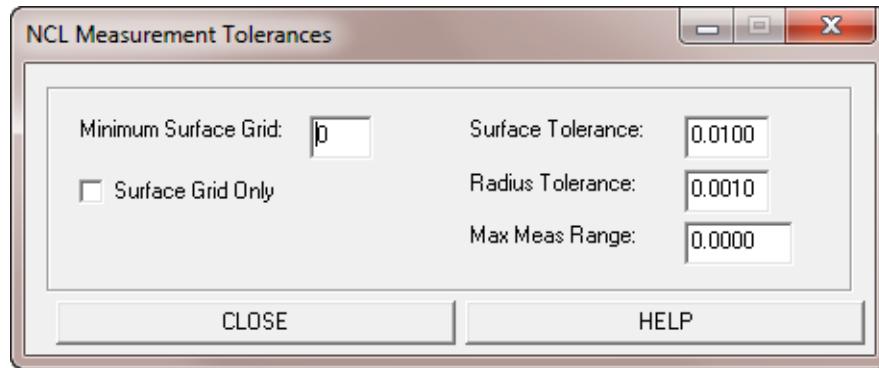
Selection mode will automatically be ended after a valid selection is made.

Units:

Normally the measurements output to the Status window will be in the same units as the model, but you can change the measurements to be output explicitly in Inches or Millimeters by selecting the desired output units in this field.

Tolerances

The Tolerances button brings up the Tolerances form as shown on next page. This form allows you to change various tolerances/settings used for measurements.

**Minimum Surface Grid:**

Specifies the minimum number of points in the U and V directions used for comparing distances to a surface. Specifying a value of zero will not generate any grid points when measuring surfaces.

Surface Tolerance:

Specifies the tolerance used to create points in the U and V directions for comparing distances to a surface.

Surface Grid Only:

When this box is checked, then only the surface grid will be used to compare distances to a surface. Measurement points within a tolerance of the surface will not be used. Uncheck this box if you would like to use both the grid and tolerance generated points for surface measurements.

Radius Tolerance:

Specifies the tolerance used to create points in the U and V directions for comparing radii to a surface.

Max Meas Range:

Surface distances will only be compared within this distance of the point picked on the surface.

CLOSE

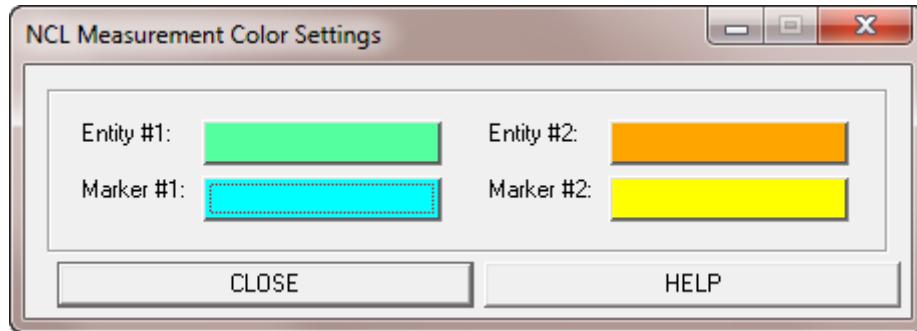
Click this button to close the form.

HELP

Click this button to open the online help of this form.

Colors

The Colors button brings up the Measurement Color Settings form as shown below. This form allows you to change various color settings used for displaying the measurement input and results.

**Entity #1:**

Click this button to open the Color form to select the highlight color of the first entity picked.

Entity #2:

Click this button to open the Color form to select the highlight color of the second entity picked.

Marker #1:

Click this button to open the Color form to select the highlight color of the first point from which the measurement was taken.

Marker #2:

Click this button to open the Color form to select the highlight color of the second point from which the measurement was taken.

CLOSE

Click this button to close the form.

HELP

Click this button to open the online help of this form.

Single Moves

If the Single Move box is checked, then single motion segments will be available for selection when picking mode is entered. If the Single Move box is left unchecked, then the entire movement generated by a motion command will be available for selection when picking mode is entered.

Status

Displays the results of the measurement.

CLOSE

Click this button to close the form.

HELP

Click this button to open the online help of this form.

G11**Motion Query And Edit Tool**

NCL has the ability to allow their users to pick motion off the screen and display the associated machining attributes and source code that generates the motion.

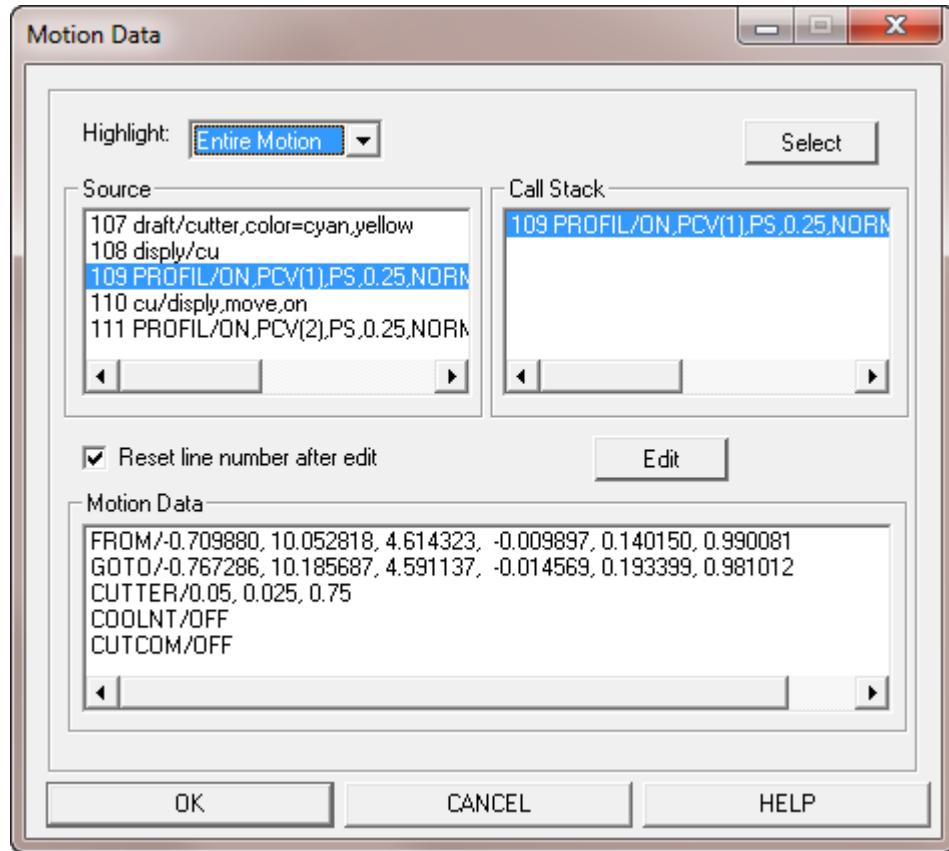
The machining attributes consist the following items:

- FROM Position
- GOTO Position
- CUTTER Parameters
- Loaded Tool
- Feed Rate
- Spindle Speed and Direction
- Coolant Setting
- Cutter Compensation Mode

The source code that created the motion will be displayed with a few lines prior to and after this line.

The following shows the graphical interactive interface of the motion query and a description of how to use this interface. This interface as shown below can be activated by using the following menu sequence:

Status > Motion Data > Pick the motion



Highlight

In verify mode, either the Entire Motion that was generated by a single command can be highlighted when the cursor is placed over any section of that motion, or only a Single Move within the displayed motion can be highlighted. Either method will still display the attributes for the move and the source that generated the entire motion.

Select

Selects a new motion segment to display the attributes for.

Source

Displays the source code that generated the displayed motion.

Call Stack

Displays the active call/loop stack when the displayed motion was generated. The call/loop stack consists of the actual command that generated the motion and any Macro calls or DO loops that were active at the time of the motion's creation.

Macro calls will display simply as CALL/macro and DO loops will be displayed as DO/lable,index=n, where 'index' is the name of the controlling DO loop variable and 'n' is its value at the time that the motion was generated.

Selecting a line from the Call Stack list will display the associated source lines in the Source list.

Reset the line number after edit

Checking this box causes **NCL** to reset the current source line number to the line that was active prior to displaying the Entity Data form. If this box is not checked, then the current source line will be the line that was active when Command Mode was exited after pressing the Edit button.

Edit

Pressing the Edit button when there is source code displayed will enter Command Mode at the line that is active in the Source list.

Motion Data

Displays the from and to position of the selected motion along with its cutting attributes.

Caveats

- Playing back an external clfile will not have any source lines associated with it.
- Playing back a simulation file created using the Playback File form will have associated source lines, but a Posted file created from this form will not.

G.12 Customized Screen Layout With Customized View

NCL has the ability to allow their users to define their own default screen layouts and individual views. This is accomplished by creating a data file with the screen definition in it. This screen definition file is referenced by the UV_SCREEN_LAYOUT environmental variable in the ncl.init file and is formatted as follows:

```
#VIEW#
/NAME/ view_name
/CENTER/ x,y,z
/NORMAL/ i,j,k
/UP-AXIS/ i,j,k
/SCALE/ scale_factor

#VIEW#
...other view definition
...
#SCREEN#
/NAME/ screen-name

/VIEWPORT/ viewport-name
/POSITION/ x,y
/SIZE/ x,y
/VIEW/ view-name
/MOTION/ *YES
/BORDER/*YES
/AXIS/ *YES
/NAME/ *YES
/APERTURE/ *YES
/DISPLAY/ *SHADED

/VIEWPORT/ viewport-name
....other viewport definition within the same screen
...
#SCREEN#
...other screen definition
...
```

where:

#VIEW#	Denotes the beginning of a new view definition.
--------	---

/NAME/	Specifies the name of the defined view. Any view name will be allowed except the NCL internal defined view names which are: "Front", "Back", "Top", "Bottom", "Left", "Right", "Isometric" "Left ISO", "Dimetric", "Left Dimetric", "Invisible", "Extern Unibase" and "Layer1".
/CENTER/	Defines the center location of the view.
/NORMAL/	Defines the normal vector of the view.
/UP-AXIS/	Defines the Y-axis vector of the view.
/SCALE/	Defines the scale factor of the view and must be a positive value.
#SCREEN#	Denotes the beginning of a screen layout definition with up to 20 viewport definitions for each screen definition.
/NAME/	Specifies the name of the screen layout. Any screen name will be allowed except the NCL internal defined screen names which are: "Single", "Horiz Dual", "Vert Dual", "Quad", "Six Equal", and "Five and One".
/VIEWPORT/	Specifies the name of the viewport and must be unique within this file. The same viewport name cannot be used for different screen definition.
/POSITION/	Defines the lower left corner of the viewport using a percentage of the graphics area in X & Y (0 through 1).
/SIZE/	Defines the size of the viewport using a percentage of the graphics area in X & Y (0 through 1).
/VIEW/	Defines the default view that will be used for this viewport and must already be defined either internally by NCL or by a preceding #VIEW# block. If it is not defined, then Front will be used.
/MOTION/	Defines to display or not to display motion in this viewport. This can be either *YES or *NO.
/BORDER/	Defines to display or not to display border around this viewport. This can be either *YES or *NO.

/AXIS/	Defines to display or not to display an axis image on the lower left corner of this viewport. This can be either *YES or *NO.
/NAME/	Defines to display or not to display the View Name (not the Viewport Name) on the lower left corner of this viewport. This can be either *YES or *NO.
/APERTURE/	Defines to display or not to display the aperture size on the lower left corner of this viewport. This can be either *YES or *NO.
/DISPLAY/	Defines the default display mode of this viewport. This can be either *SHADEDED (displayed both wire and shaded mode), *WIRE (only displayed in wire frame mode), *HIDDEN (displayed in wire frame hidden mode), or *SHADEDED_ONLY (displayed in shaded mode only, no wire entity will be displayed).

This user defined default screen layout is different than the **NCL** default screen layout in that it will not be loaded automatically when starting a new interactive session. **NCL** will automatically load the **NCL** default screen layout which has a single viewport with the view name “FRONT”. The user needs to use the following menu sequence to load the user defined screen layout into the new interactive session.

View > View Manage > Select Screen

The user can also accomplish the same task by using a playback record file which recorded the above menu sequences in the file.

G13 Customizing Macro Form

There are two ways to customize a macro form: External File and Interactive

G13.1 Customizing Macro Form With External File

NCL has the ability to allow the user to create an external file that describes the look and feel of Dynamic Macro Call forms. The form files give the user greater control over the form appearance.

The external file should be named macro.frm and be stored in the NCL_INCDIR directory.

The form file will consist of a header description.

```
#HEADER#
/TITLE/ title bar text
/POSITION/ 50,50
/SIZE/ x,y
```

Where:

- The title bar text will be displayed along the title bar of the form. This is the same as the description field on the PROMPT/macro command.
- The POSITION field is ignored. The SIZE field determines the initial size of the form in screen pixels. The default position is in the center of the screen the first time the Macro form is displayed. The position and size will be remembered on subsequent calls.

Each Macro parameter must have its own field descriptor in the form. The form field descriptors must also be defined in the same order as the parameters on the Macro definition. The Macro parameter field definitions have the following syntax.

```
#field-type#
/LABEL/ prompt
/POSITION/ x1,y1 [,x2,y2]
/SIZE/ x,y
/TYPE/ input-type
/INPUT/ pick-mode
/LIMIT/ pick-mask
/CHOICES/ word-list
/LEN/ length
/PREC/ precision
/RANGE/ min,max
/ACTIVE/ YES , NO
```

Where:

- “#field-type#” defines the type of field to be displayed in the form and can be one of the following.
 - #EDIT# Defines a text input field. The expected input can be a geometry label, vocabulary word, scalar, numeric value, etc.

#CHOICEBOX# Defines a choice field that is typically populated by vocabulary words.

When a *CHOICEBOX* field is defined in the form file, then a corresponding PROMPT command in the Macro definition should also be used to define this Macro parameter as a Choice box by specifying a list of vocabulary words on the PROMPT command. The word-list defined in the form field should match the word-list for the corresponding PROMPT command. If it does not match, then the default setting when the form is first displayed will be undefined and the choice text selected from the form will be used as the parameter setting when the Macro is called.

#CHECKBOX# Defines a checkbox field. Checkbox fields will return a value of 0 when the box is not checked or a value of 1 if the box is checked. These are the only two values that can be returned.

When a *CHECKBOX* style field is defined in the form file, then a corresponding PROMPT command in the Macro definition should all be used to define the Macro parameters as a Check box by specifying a numeric range of 0,0. This is a new feature with the Macro form file enhancement and allows Check box fields to be defined using this method without using a form file.

- The “**/LABEL/**” entry defines the prompt for the field.
- The “**/POSITION/**” field determines the position at the upper left of the field in device pixels. The first field is typically started at X=10 and Y=8. Each line in the form is typically 17 pixels (including the space between the lines), so a field starting on the second line will be at X=10 and Y=25.
- The “**/SIZE/**” field defines the size of the form field in device pixels. The X-value is determined by the length of the prompt and corresponding entry portion of the field (text box, choice box, check box). Since a variable width font is typically used with forms, there is no set way of determining the field length, it should be done by trial and error. The height of the field is usually Y=14.
- “**/TYPE/**” should be set to UD_DASSTRING for “**#EDIT#**” and “**#CHOICEBOX#**” fields, UD_DASUNITLESS for “**#EDIT#**” fields that require a real value (min,max specified on the corresponding PROMPT statement), and UD_DASINT for “**#CHECKBOX#**” fields.

- The “**/INPUT/**” entry is only used with “**#EDIT#**” fields and determines if the user is allowed to enter By pick mode to select geometry from the screen. It can have one of the following values.

FORM_STRING Only text input is allowed. A By pick button will not be enabled for this field. This is the default method if the “**/INPUT/**” entry is not specified.

FORM_PICK The user can pick geometry from the screen as input to this field. Then entire label and subscript of the selected geometry will be stored in the field.

FORM_LABEL The user can pick geometry from the screen as input to this field. Only the label of the selected geometry will be stored in this field.

FORM_SUBSCR The user can pick geometry from the screen as input to this field. Only the subscript of the selected geometry will be stored in this field.

- The “**/LIMIT/**” entry is only used when geometry picking is enabled in the “**/INPUT/**” field. It contains a list of geometry types that can be picked for this field. Valid geometry types are ANOTE, CIRCLE, CURVE, LINE, MATRIX, PATTERN, PLANE, PNTVEC, POINT, SHAPE, SURF, SYMBOL, and VECTOR. Multiple geometry types can be specified separated by commas.
- The “**/CHOICES/**” entry defines the acceptable words that can be entered for this field. This typically matches the vocabulary word list entered on the PROMPT command for this Macro parameter as described in the “**#CHOICEBOX#**” section. Commas should separate the acceptable words and multiple “**/CHOICE/**” lines can be used if there are more words than will fit on a single line.
- The “**/LEN/**” and “**/PREC/**” entries are only used with numeric input (UD_DASUNITLESS) and override the length,precision parameters on the corresponding PROMPT statement. The “**/RANGE/**” entry is also only used with numeric input fields and overrides the min,max parameters of the PROMPT statement.
- The “**/ACTIVE/**” entry determines if the field should be visible to the user when the form is displayed. Enabling the field is typical for almost every case when defining Macro parameter fields, though there are times when you may always want to use the default value of a certain Macro parameter. This entry is typically used with the Macro option fields that are described further down in this document. Since each Macro parameter must have a field description in the form file along with the fields for the Macro options, the “**/ACTIVE/**” entry allows individual fields to not be displayed

while keeping their position in the form definition. **YES** enables this field and is the default setting. **NO** will disable this field.

From the above descriptions you can see that a PROMPT command is sometimes required in the Macro definition depending on the field type used. The following sections describe how the PROMPT parameters interact with their corresponding form file entries.

```
PROMPT/macro[,class] [OUT ,] [,DEFALT],description
                      OMIT      RETAIN
```

The Macro description prompt does not have any corresponding fields in the form file except for the description which is overridden by the “**/TITLE/**” field in the form file. All other settings are made by the PROMPT command.

```
PROMPT/parameter[,len,prec],prompt          $
[ ,SUBSTR,RETAIN] [,NOW ] [.min,max ]
          LABEL      NEXT
          NUM        word-list
                      geo-list
```

While the Macro parameters are referenced by name in the PROMPT command, the form file must have a field assigned to each parameter in the order that the parameter is defined in the Macro definition.

- “*len,prec*” are overridden by the “**/LEN/**” and “**/PREC/**” entries in the form file.
- prompt is overridden by the form file “**/LABEL/**” entry. The *SUBSTR* parameter is overridden by the form file “**/INPUT/**” entry.
- *NOW* and *NEXT* do not have a corresponding entry in the form file, the PROMPT command must be used to define this setting.
- min,max when specified designates the form field as a numeric value field (UD_DASVAL) or a “**#CHECKBOX#**” field if the values are set to 0,0. These range values can be overridden by the “**/RANGE/**” entry in the form file when a numeric field is being defined. The min,max parameters must be specified if the form field type is set to UD_DASVAL.
- The word-list parameters must be specified if the field type is set to “**#CHOICEBOX#**” and the word list should match the choice list specified in the “**#CHOICES#**” entry in the form file. If the two word lists do not match, then the form file takes precedence and the default value when the form is first displayed will be incorrect.

- The geo-list parameters are overridden by the form “/**LIMIT/**” entry.

There are also five option fields that are automatically added to Dynamic Macro Call forms. These fields are not referenced in the PROMPT command (though the Macro definition PROMPT command does set some of the default values), but rather are added to the bottom of the form by **NCL**.

When a form file is used for the Dynamic Macro Call, then these fields must be part of the form file. They are specified at the end of the form file in the following order.

Field	Field Type
Place in Source	#CHECKBOX#
Remember Input Values	#CHECKBOX#
Output Default Values	#CHECKBOX#
View	#PUSHBUTTON#
Dynamic View	#PUSHBUTTON#

If these fields are not defined in the form file, then they will not be accessible to the user. If you wish to display any of these fields, then all of the fields should be defined in the correct order. If you do not want to display all of these fields, then use the “/**ACTIVE/ NO**” entry on the fields that you do not want to display.

The “#PUSHBUTTON#” field type is not available for the Macro parameter fields, only for the View and Dynamic View option fields. These field types have the following format.

```
#PUSHBUTTON#
/LABEL/ button-text
/POSITION/ x,y
/SIZE/ x,y
/TYPE/ UD_DASINT
/ACTIVE/ YES , NO
```

When a form file is used to define the Macro form layout you also have the option of displaying a frame around certain fields in the form. The following section shows how this is accomplished.

```
#FRAME#
/TITLE/ frame-text
/POSITION/ x,y
/SIZE/ x,y
```

- “frame-text” is the title that is displayed with the frame.

- “**/POSITION/**” determines the position at the upper left of the frame in device pixels. The X-value should be less than the starting X-value for input fields, for example X=5. The Y-value is typically 17 pixels lower than the previous line and 10 pixels above the next line.
- “**/SIZE/**” determines the size of the frame in device pixels. This is typically determined by trial and error.

One other benefit to defining form files for common Macros is the ability to add help text to the Macro. This is accomplished by including a help section in the form file defined using the “**#HELP#**” field type. For example,

```
#HELP#
(help text)
```

When a **#HELP#** section is defined in the form file, then the *HELP* button will displayed along with the *OK* and *CANCEL* buttons on the Dynamic Macro Call form. When the user presses this button another window will be displayed containing the defined help text, the same as with standard **NCL** forms.

The following pictures show the differences in a Dynamic Macro Call form appearance on a screen with resolution 1280 X 800 when defined as a simple Macro without PROMPT commands, with PROMPT commands, and with an external form file. The Macro and form file are defined as follows.

Macro Definition

```
TLCHG =Macro/Tlno=1,Dial=1.0,Cor1=0.,Hgt1=1.0,      $
        Rpm1=100,Dir1=CLW,Col1=FLOOD,Tcpt,      $
        1st=0,Last1=0
PROMPT/TLCHG,OUT,RETAIN,"Tool Change Macro"
PROMPT/Tlno,8,0,"Tool Number",1,9999
PROMPT/Dial,8,3,"Tool Diameter",0,10
PROMPT/Cor1,8,3,"Corner Radius",0,5
PROMPT/Hgt1,8,3,"Tool Height:",0,20
PROMPT/Rpm1,8,3,"Spindle Rpm:",40,12000
PROMPT/Dir1,"Spindle Direction:",CLW,CCLW,BOTH
PROMPT/Col1,"Coolant:",FLOOD,MIST,AIR,TOOL
PROMPT/Tcpt,"Tool Change Point",POINT,PNTVEC
PROMPT/1st,"Initial Call",0,0
PROMPT/Last1,"Last Call",0,0
```

Form File tlchg.frm

```
#HEADER#
/TITLE/ Tool Change Macro
/POSITION/ 50,50
/SIZE/240,215

#FRAME#
/TITLE/ Tool Data
/POSITION/ 5,8
/SIZE/ 225,50

#EDIT#
/LABEL/ Tool Number:
/POSITION/ 10,18, 60,18
/SIZE/ 100,14
/TYPE/ UD_DASUNITLESS
/LEN/ 8
/PREC/ 3
/RANGE/ 1,9999

#EDIT#
/LABEL/ Tool Diameter:
/POSITION/ 120,18, 170,18
/SIZE/ 100,14
/TYPE/ UD_DASUNITLESS
/LEN/ 8
/PREC/ 3
/RANGE/ 0,10

#EDIT#
/LABEL/ Corner Radius:
/POSITION/ 10,35, 60,35
/SIZE/ 100,14
/TYPE/ UD_DASUNITLESS
/LEN/ 8
/PREC/ 3
/RANGE/ 0,5

#EDIT#
/LABEL/ Tool Height:
/POSITION/ 120,35, 170,35
/SIZE/ 100,14
```

```
/TYPE/ UD_DASVALUE  
/LEN/ 8  
/PREC/ 3  
/RANGE/ 0,20

#FRAME#
/TITLE/ Spindle
/POSITION/ 5,62
/SIZE/ 225,50

#EDIT#
/LABEL/ SPINDL
/POSITION/ 10,72
/SIZE/ 90,14
/TYPE/ UD_DASVALUE
/LEN/ 8
/PREC/ 3
/RANGE/ 40,12000

#CHOICEBOX#
/LABEL/ ,
/POSITION/ 105,72
/SIZE/ 60,14
/TYPE/ UD_DASINT
/CHOICES/ "CLW", "CCLW", "BOTH"

#CHOICEBOX#
/LABEL/ COOLNT
/POSITION/ 10,89
/SIZE/ 80,14
/TYPE/ UD_DASINT
/CHOICES/ "FLOOD", "MIST"
/CHOICES/ "AIR", "TOOL"

#FRAME#
/TITLE/ Control
/POSITION/ 5,116
/SIZE/ 225,55

#EDIT#
/LABEL/ Tool Change Point
/POSITION/ 10,133
/SIZE/ 160,14
```

```
/LEN/ 12
/TYPE/ UD_DASSTRING
/INPUT/ FORM_PICK
/LIMIT/ POINT, PNTVEC

#FRAME#
/TITLE/ Call
/POSITION/ 160,121
/SIZE/ 60,45

#CHECKBOX#
/LABEL/ First
/POSITION/ 165,133
/SIZE/ 40,14
/TYPE/ UD_DASINT

#CHECKBOX#
/LABEL/ Last
/POSITION/ 165,150
/SIZE/ 40,14
/TYPE/ UD_DASINT

#CHECKBOX#
/LABEL/ Place in Source
/POSITION/ 5,190
/SIZE/ 100,4
/TYPE/ UD_DASINT
/ACTIVE/ NO

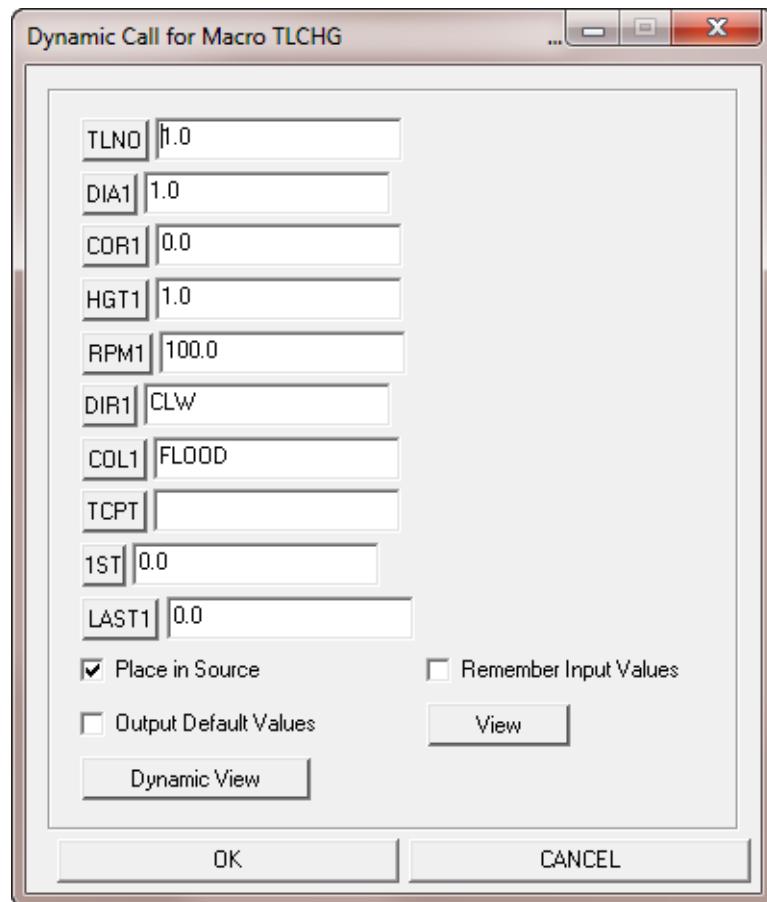
#CHECKBOX#
/LABEL/ Remember Vals
/POSITION/ 130,190
/SIZE/ 100,14
/TYPE/ UD_DASINT
/ACTIVE/ NO

#CHECKBOX#
/LABEL/ Output Default
/POSITION/ 10,150
/SIZE/ 85,14
/TYPE/ UD_DASINT
/ACTIVE/ YES
```

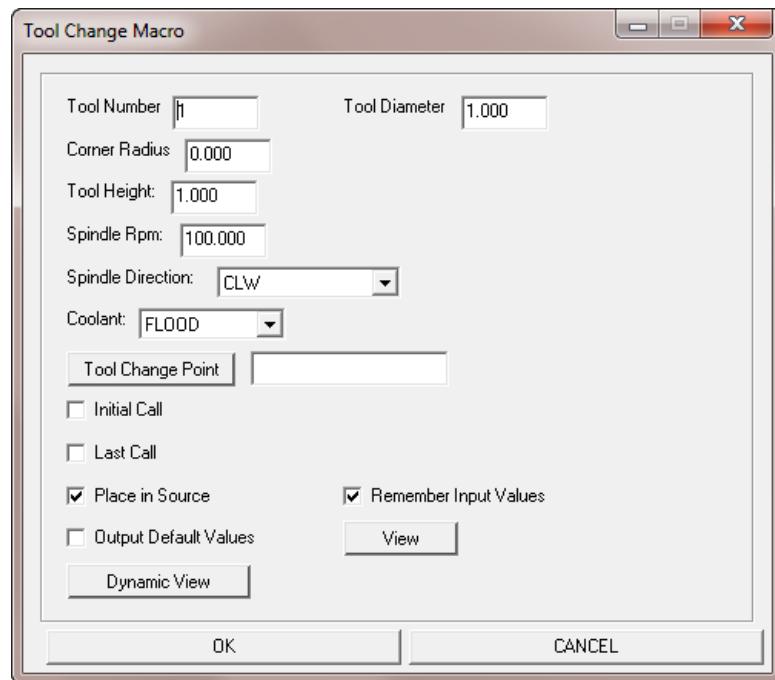
```
#PUSHBUTTON#
/LABEL/ View Macro
/POSITION/ 110,190
/SIZE/ 50,14
/TYPE/ UD_DASINT
/ACTIVE/ NO

#PUSHBUTTON#
/LABEL/ Dynamic View
/POSITION/ 90,175
/SIZE/ 60,14
/TYPE/ UD_DASINT
/ACTIVE/ YES

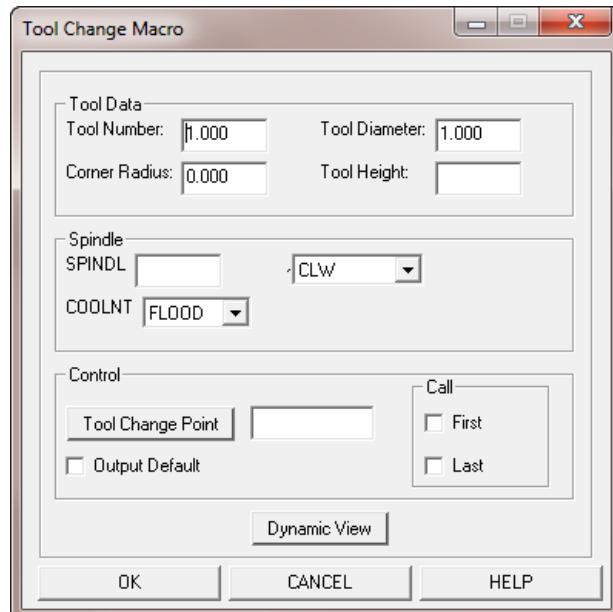
#HELP#
(Help Text)
```



TLCHG macro form without the PROMPT statements



TLCHG macro form with the PROMPT statements



TLCHG macro form with the tlchg.frm file

Note:

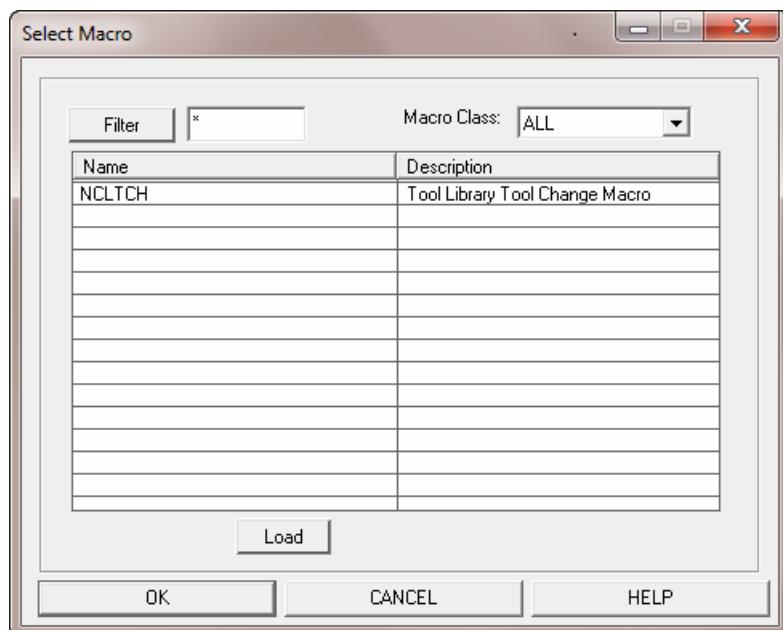
- When By Pick mode is entered and stayed in automatically (either by outputting the CALL command after each pick or by traversing to the next By Pick style field) the user will now see the form flash between picks.
 - When first defining the form, it is possible to repeatedly call the Macro interactively and make changes to the form between each call. The form will be loaded from the file each time the Macro is called.
 - The position and size of Dynamic Macro forms when defined using a form file will be remembered for subsequent calls by creating the macro.pos file in the local forms directory, the same as for standard **NCL** forms.
 - A local Dynamic Form file can be created in the local user forms directory that will override the form file in the NCL_INCDIR directory.

G.13.2 Customizing Macro Form Interactively

NCL allows the user to interactively create a custom interface form for a predefined macro. This function can be activated by using the following menu sequence:

Macros > Form Design

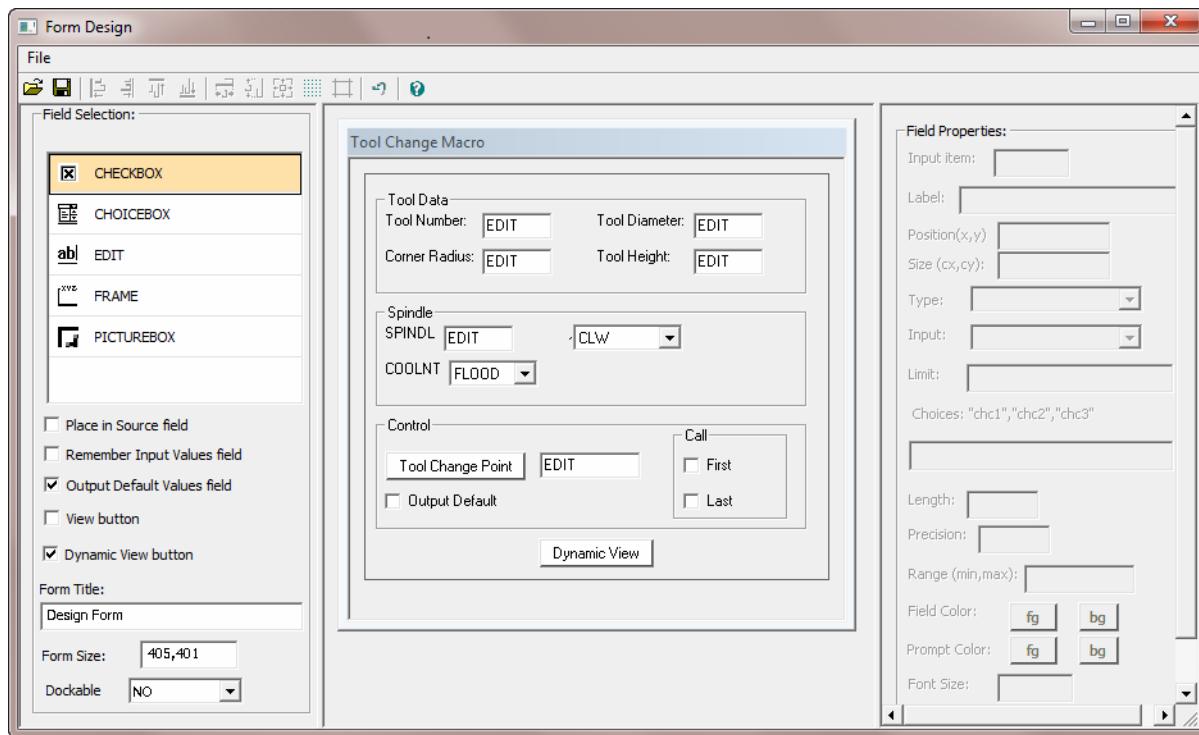
This will bring up the standard Select Macro form used for dynamically calling a macro as shown below:



After selecting a Macro the Form Design interface is displayed allowing the user to interactively design the form. The initial form displayed in the interface is determined in the same manner as when displaying the Macro form when the Macro is dynamically called using the interface.

1. If a form file exists with the same name as the Macro, then this form will be loaded.
2. If there are PROMPT statements defined within the Macro, then these will be used to define the form layout.
3. **NCL** will automatically create the initial form design if there is no form file or PROMPT statements.

The Macro Form Design interface has the following appearance. In this case a Macro with a predefined form file is loaded.



The Form Design interface has a menu bar and three other sections.

Menu Bar

The Menu Bar at the top of the interface allows you to open and save form files and control the placement and sizing of form fields.



Prompts the user to save the existing form if it has been changed and displays the Select Macro form allowing you to select another Macro to design a form for.



Displays a file browser allowing you to save the current Macro form being designed.



Left aligns the selected fields. The fields displayed using the hollow handles will be aligned on the left side with the field displayed using the black handles. Refer to the Form Display section for a description on how to select fields.



Right aligns the selected fields.



Aligns the selected fields along the top.



Aligns the selected fields along the bottom.



Makes the selected fields the same width.



Makes the selected fields the same height.



Makes the selected fields the same width and height.



Displays a Snap Grid. All fields placed in the form or moved within the form will have their position snap to the closest grid point.



Displays guide lines used for visually aligning fields. The guide lines are displayed in green and follow each edge of the selected field going all of the way to the edges of the form. The guide lines move with the selected field as it is dragged in the form.



Restores the last deleted field. The ^Z key performs the same function.



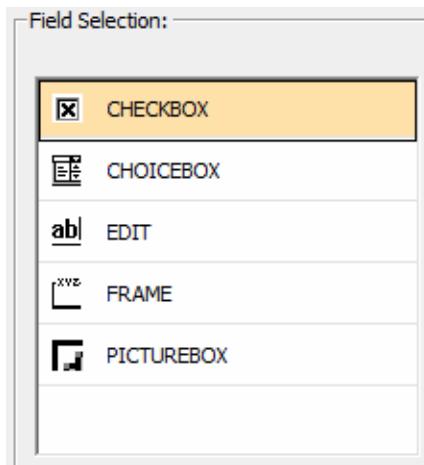
Opens a text editing window displaying the Help text associated with the form. This help text is created by the user and will be displayed when the HELP button is pressed on the Dynamic Macro Call form.

Field Selection

The Field Selection section is used to drag fields onto the form, enable/disable the fields automatically placed at the end of a Macro form, and define global settings for the form.

The valid field types are contained in the list at the top of this section. These fields can be dragged onto the form displayed in the Form Display section of the window. When changing the field type of an existing field you can simply drag the new field type over the existing field. This field will then be replaced with the new field type.

FRAME and PICTUREBOX fields do not correspond to a Macro parameter, therefore these fields should be dragged onto the form in a blank spot. They will not replace an existing field.

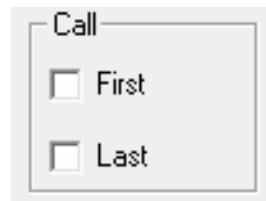


CHECKBOX fields display a check (or tick) box along with a prompt. These fields return a value of 0 when the box is blank and 1 when the box is checked.

CHOICEBOX fields contain a list of vocabulary words that the user can choose from. The valid vocabulary words to choose from are defined in the Field Property Choices field.

EDIT fields contain either a text string or numeric value depending on the type of input selected in the Field Properties Type field.

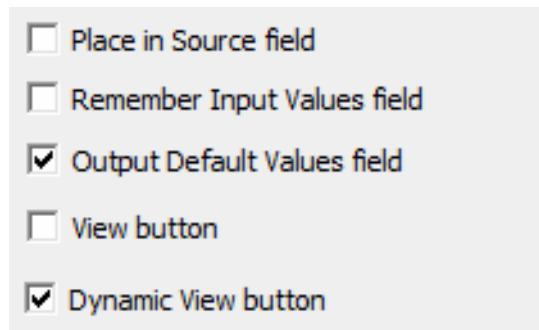
A FRAME causes a frame to be displayed around selected fields for the sole purpose of visual clarity. You can add a title to the top frame edge.



Sample Frame

A PICTUREBOX displays a picture in the form. These pictures can be just for display purposes or can be used interactively to activate certain fields in the form. Currently the Form Design function does not support the ability to make a PICTUREBOX interactive, you will need to manually edit the form in order to add the picture field controls. Refer to the [Adding Picture To A Macro Form](#) section for a complete description of Form Pictures.

Directly under the field types list are the controls for displaying and hiding the option fields that are automatically added to Dynamic Macro Call forms. These fields are not accessible to the Macro itself, but rather are controlled by **NCL**. The field types of these fields cannot be changed, but the appearance and prompts can.



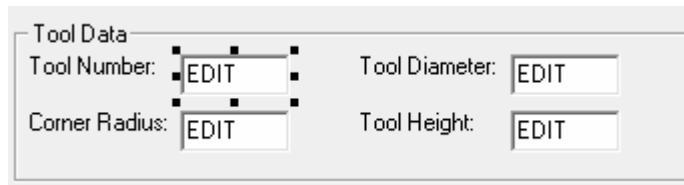
Clearing the checkbox of any of these prompts will disable the corresponding field from appearing in the form.

The remaining fields in the Field Selection section control the form's title, size, and whether the form can be docked along any edge of the **NCL** window.

Form Display

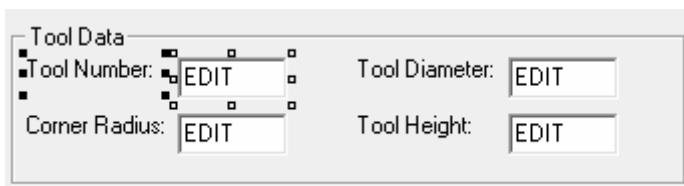
The Form Display section shows what the form currently looks like. The user can interactively place, move, and resize fields in this area.

Clicking on a field will select this field and display the resizing handles around the field in black. This is the main selection.



You will notice that the actual fields can be selected separately from their prompts. This is because the prompt can be positioned separately from its field. In the above example notice how all of the prompts and edit boxes are lined up. Only by separately positioning the prompt and edit fields can this be accomplished.

If you would like to move a prompt and its edit field together (or any two fields for that matter), then after selecting the edit field you can select the prompt field, but don't let up on the left mouse button, keep it pressed down while positioning both fields.



You can also manually select multiple fields by first selecting a field and then holding down either the Shift or Ctrl key while selecting other fields. Holding down the Shift key will keep the first field selected as the controlling field (displayed using black resize handles). Holding down the Ctrl key will make

the last field selected the controlling field. All other selected fields are displayed using clear resize handles.

Once you make a multiple selection you can either drag the fields to another location within the form or use the alignment buttons described in the Menu Bar section to align the fields.

To deselect all fields simply select a position within the form where there are no fields defined.

Field Properties

The Field Properties section defines the properties of the currently selected field. When you select a field in the form its properties will be displayed in this section. Following is a list of properties that can be changed.

Input item: contains the Macro parameter that is associated with the selected field. All fields contained in the form must be associated with a Macro parameter and all Macro parameters must have an associated field. A check is made when the form is saved that both of these conditions are true.

Label: defines the prompt label to display with this field.

Size (cx,cy): The field size is typically controlled by selecting the field and interactively manipulating its size by using its resizing handles. You can optionally type in a new size in this field and the size of the field will change to match the input values.

Type: specifies the input type of an EDIT field. It can be set to either TEXT or VALUE. TEXT specifies that the input type can be a text string, such as a label, scalar value, vocabulary word, etc. The format of a TEXT field is further controlled by the Input field. The format of a VALUE field is controlled by the Length, Precision, and Range fields. The Type field is only displayed when the form field is an EDIT field.

Input: The Input field is only active when Type is set to TEXT. It controls the picking mechanism for this form field. You can select from one of the options.

By Text	The prompt is not placed in a Push Button. The user must enter the data within this field by using the keyboard.
By Pick	The prompt text is displayed in a Push Button. The user can press this button to select an entity from the graphics area or enter data using the keyboard.
By Location	The prompt text is displayed in a Push Button. The user can press this button to select a screen location from the graphics area or enter data using the keyboard.
Label Only	The prompt text is displayed in a Push Button. The user can press this button to select an entity from the graphics area or enter data using the keyboard. If an entity is selected, then only its label without a corresponding subscript is entered into this field.
Subscript Only	The prompt text is displayed in a Push Button. The user can press this button to select an entity from the graphics area or enter data using the keyboard. If an entity is selected, then only its subscript value without a corresponding label is entered into this field.

Choices:

is only enabled when the form field is a CHOICEBOX field. Enter the vocabulary words that will be presented to the user here. The vocabulary words are separated by commas and can be quoted or not. Only valid **NCL** vocabulary words can be used.

Length:

is only enabled when the field type is set to VALUE. It defines the total length in digits of this form field's value for formatting purposes.

Precision:

is also only enabled when the field type is set to VALUE. It defines the number of digits to the right of the decimal point to display in the form field value.

Range (min,max): is also only enabled when the field type is set to VALUE. It defines the minimum and maximum values that can be entered into this form field.

Field Color: defines the text (fg) and background (bg) colors for the selected field. Pressing either of these buttons brings up the Colors form allowing you to choose the color for the selected field.

Prompt Color: defines the text (fg) and background (bg) colors for the selected field's prompt. Pressing either of these buttons brings up the Colors form allowing you to choose the color for the selected prompt.

Font Size: specifies the scale factor for the select field. A value of 1.0 displays the text at the default size, while a larger value displays larger text (2.0 will display the text at double its default size) and a smaller value displays smaller text (.5 displays the text at half its normal size). This field will only be enabled when the selected field is a CHOICEBOX or EDIT type field.

G.13.3 Adding Picture To A Macro Form

NCL has the ability to allow the user to add pictures to the macro form. This can be done by modifying the corresponding macro form file manually or interactively.

A “#PICTUREBOX#” section as shown below can be added to the corresponding macro form file to add a picture. Multiple “#PICTUREBOX#” sections can be added to add multiple pictures to the form.

```
#PICTUREBOX#
/FILE/ file.ext
/NAME/ label
/POSITION/ x,y
/SIZE/ x, y
```

- “file.ext” is the filename of the picture to display. BMP, GIF, JPG and DIB file types are supported.
- “label” defines the picture name. Multiple pictures can be displayed in a single form.
- “/POSITION/” determines the position at the upper left of the frame in device pixels.

- “/SIZE/” determines the size of the picture field in device pixels.

This section can be added anywhere in the corresponding macro form file between sections. They are usually placed immediately after the “#HEADER#” section for ease of maintenance.

Virtual boxes can be created inside the picture frame and used to activate a linked field defined in the form. The following line of code can be added to the end of the section needs to be linked.

/PICTURE/ label [,“tooltip”], xmin, ymin, xmax,ymax,[, parm]

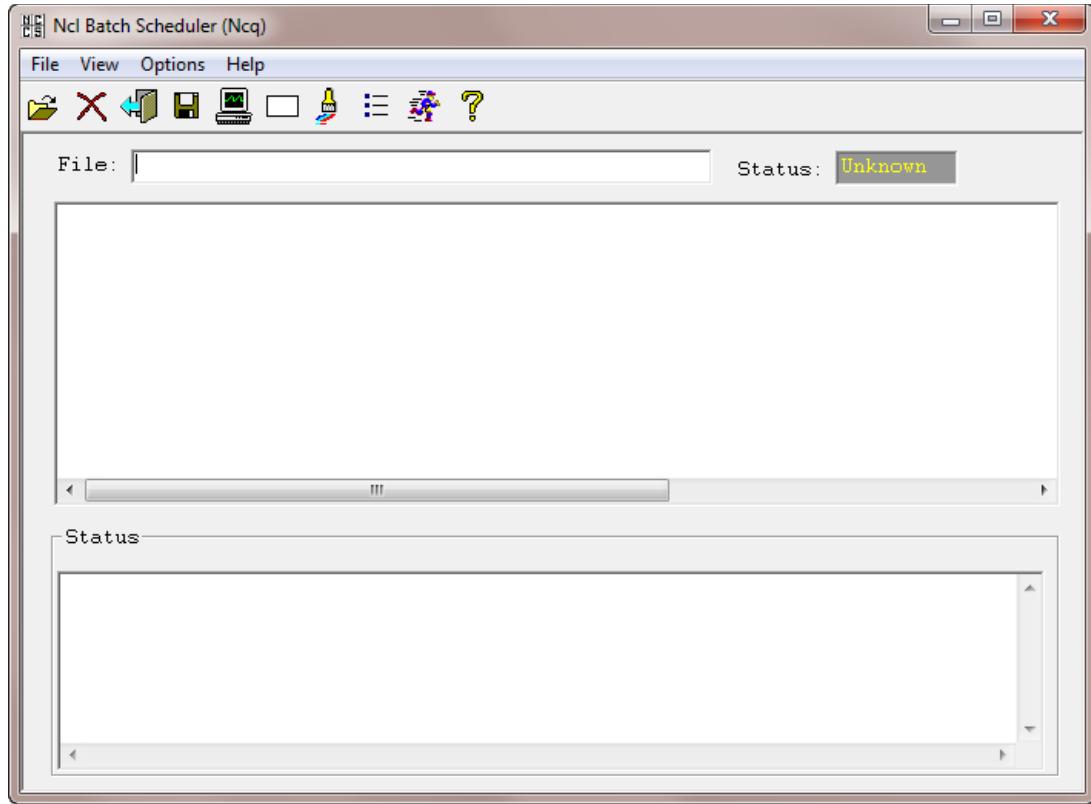
- “label” is the name of the #PICTUREBOX#” this field refers to.
- “tooltip” is an optional quoted text string that contains the text that will be displayed when the cursor is moved over the defined area in the picture.
- “xmin,ymin, xmax,ymax” defines the area of the picture that is used to activate this field or display the tool tip. “xmin, ymin” defines the upper left corner and “xmax,ymax” defines the lower right corner. They can have a value between “0” and “100”. The upper left corner of the picture is defined as “0,0” and the lower right corner of the picture is defined as “100,100”.
- “parm” is an optional parameter as defined below and depends on the type of field it links to.

<u>Field</u>	<u>Action</u>
CHECKBOX	Activates the checkbox field. An optional parameter can be specified that is used to make a selection within the checkbox. “ON” enables the checkbox, “OFF” disables it, and “TOGGLE” will toggle the state of the checkbox.
CHOICEBOX	Activates the choice box field. An optional value can be specified that is used to make a selection within the choice box. A value of “0” selects the first choice listed, “1” the second choice, etc. A value of “-1” will select the next choice (toggles through the choice box selections).

CHOICE_LIST	Activates the list. An optional text string can be specified the automatically selects the matching entry in the list.
COMBLIST_SIMPLE	Activates the list. An optional text string can be specified the automatically selects the matching entry in the list.
DATATABLE	Activates the table.
EDIT	Activates the text field. If the optional text string is specified, then the text field will be set to this string.
LISTBOX	Activates the list.
LISTTABLE	Activates the table.
PUSHBUTTON	Presses the button.

G.14 Batch Processing

Click **Tools > Batch Proc > Run Batch** while in interactive mode or double click the “Batch Process” shortcut icon on the Windows Desktop to open the following form as shown on next page



This form consists of the following sections: Pull Down Menus, Icon Menus, **Ncq** Window.

Pull Down Menus:

File: This menu has the following options:

Open:

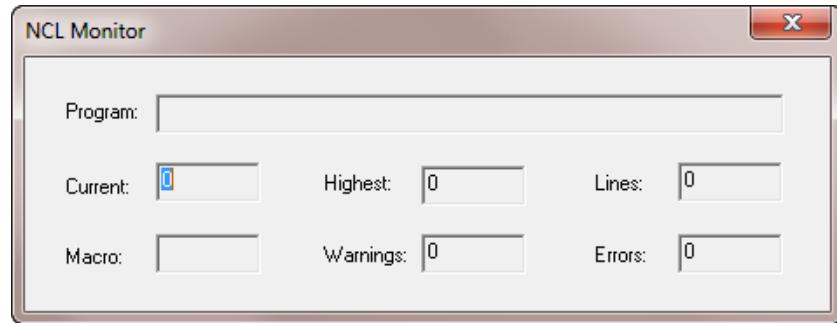
Click this to open a browser and locate a part program for batch processing. Once the file is opened, its name will be put into the *File Name* text box and the *Main* window. The batch process Options must be set first before opening a file.

With *Windows Operation* systems, a file can be dragged and dropped directly into this form for batch processing without using the *File > Open* menu. However, the name of the file will not put into the *File Name* text box and will only put into the *Main* window.

- Delete:** Click this to remove a file from the queue.
- Load Queue:** Click this to open a browser to load an external queue file into the queue.
- Save Queue:** Click this to open a browser and save the active queue to an external queue file.
- Run *NCL* Batch:** Click this to run the batch process for all the files listed in the *Queue* window.
- Exit:** Click this to exit the *NCL* Batch Process.

View: This menu has the following option:

- Update:** Click this to refresh the *Ncq* window.
- NCL Monitor:** Click this to bring up the *display only* monitor form as shown below. This is a toggle function, if the menu is selected again, it will close this form.



Program:

Displays the name of the part program currently running.

Current:

Displays the current line number being processed by *NCL*.

Highest:

Displays the highest line number processed.

Lines:

Displays the total number of lines in the part program file.

Macro:

Displays the name of the currently active Macro or blank if a Macro is not being called.

Warnings:

Displays the current number of warnings generated.

Errors:

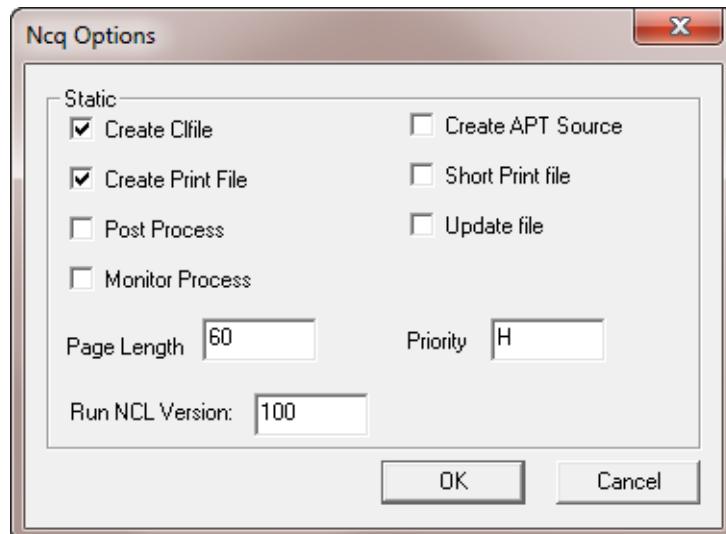
Displays the current number of errors generated.

Update Rate: Set the update rate (1, 2, 3, 5, 10 secs) for the information displayed in this form. The default update rate is 1 sec.

Clear Status: Click this to clear the *Status* window.

Options: This menu has the following options:

Options: Click this to open the form shown below.



Create Clfile:

Check this item if a CL file is to be created. The default is to output a CL file.

Create Apt Source:

Check this item if an APT source file is to be created. The default is not to output an APT file.

Create Print File:

Check this item if a listing file (.pr) is to be created. The default is to output a listing file.

Short Print File:

Check this item if a short listing file (.pr) is to be created.

Post Process:

Checking this item will cause the batch process to run the post processor automatically. The default is not to run the post processor.

Update File:

Check this item to cause the part program file to be updated with any changes kept at the end of the run. The default is not to update this file.

Monitor Process:

This specifies the *Monitor* window will display automatically during the batch process.

Page Length:

This specifies the number of lines per page in the listing file (.pr). The default is 60 lines per page.

Priority:

This specifies the priority of the run (A=Highest, Z=Lowest). The default setting is H.

Run NCL Version:

This specifies the version of **NCL** for batch processing.

Ok:

Click this button to accept the changes and close the “**Ncq** Options” form.

Cancel:

Click this button to close the “**Ncq** Options” form without accepting any changes made to the entries.

Priority: Choose between LOW, NORMAL, and HIGH. The default is NORMAL.

Time Limit: Choose between 15 minutes, 30 minutes, 1 hour, 2 hours and Disable.

A time limit can be set to make sure the batch process will not hang up in a loop. If the process time for a single part program exceeds this time limit, then **NCL** will stop the processing of this file and restart with the next program in the queue. The default is 30 minutes.

Cancel Current: Instruct **Ncq** to cancel the current files and move to the next file in the queue.

Cancel All: Instruct **Ncq** to cancel the current file and pause the batch process. The process can be resumed at the next file in the queue by pressing the RUN button.

Help: Click “*About Ncq*” to show the version number.

Icons Menus:

Same as the “File - Open” Pull down menu.



Same as the “File - Delete” Pull down menu.



Same as the “File - Load Queue” Pull down menu.



Same as the “File - Save Queue” Pull down menu.



Same as the “View - **NCL** Monitor” Pull down menu.



Same as the “View - Clear Status” Pull down menu.



Same as the “View - Update” Pull down menu.



Same as the “Options- Options” Pull down menu.



Same as the “File - Run **NCL** Batch” Pull down menu



Same as the “File - Help” Pull down menu.

Ncq Window:

This section consists of four windows. First is the *File Name* text box which displays the name of the file that is to be put into the queue. The second is the *Status* box which shows the current state of the batch process. The third is the *Main* window which lists all the files with their corresponding specified

options. The last one is the *Status* window which shows the status of the jobs that have finished.

G15 Mouse Buttons

All the mouse buttons are programmable and are defined by the mouse definition file specified by the “UZ_MOUSE_BUTDEFS” environmental variable in the “ncl.init” file or the “user.init” file.

Following shows the default settings of the mouse buttons.

```
#CHOICE#
/Left/ NCL_DYNAMIC_PAN
/Middle/ mouse_default.menu
/Right/ NCL_DYNAMIC_XYROT
/Wheel_Down/ NCL_ZOOM_UP
/Wheel_Up/ NCL_ZOOM_DOWN

#LOCATE#
/Left/ KEY_PICK_LOCATE
/Middle/ KEY_DONE
/Right/ KEY_REJECT
/Wheel_Down/ NCL_ZOOM_UP
/Wheel_Up/ NCL_ZOOM_DOWN

#PICK#
/Left/ KEY_PICK_LOCATE
/Middle/ KEY_DONE
/Right/ KEY_REJECT
/Wheel_Down/ NCL_ZOOM_UP
/Wheel_Up/ NCL_ZOOM_DOWN

#TEXT#
/Left/ KEY_NOOP
/Middle/ KEY_DONE
/Right/ KEY_REJECT
/Wheel_Down/ KEY_UPFIELD
/Wheel_Up/ KEY_DNFIELD
```

G.16 NCL Initialization Files

The following initialization files contain parameters which control the default start up condition of **NCL** and corresponding utilities. Following is a list of common parameters for each file that can be modified by the user and their meaning.

G.16.1 drwsiz.init

This file is a script file used for setting of environmental variables used by **NCL** and CADPLOT to determine the physical X and Y Limits of various plotters. Nothing needs to be changed by the user.

G.16.2 nccs.init

This file contains parameters which specify which initialization files to use. Nothing needs to be changed by the user.

G.16.3 ncl.init

This file contains parameters which specify the default conditions of **NCL**. Nothing needs to be changed by the user.

G.16.4 ncliges.init

This file contains parameters which specify the default conditions of **NCL/IGES**. Nothing needs to be changed by the user.

G.16.5 nclstep.init

This file contains parameters which specify the default conditions of **NCL/STEP**. Nothing needs to be changed by the user.

G.16.6 nclplot.ini

This file contains parameters which specify the default conditions of **NCL** plotting routine. Nothing needs to be changed by the user.

G.16.7 ncq.ini

This file contains parameters which specify the default options for the **NCL** batch process. The following is a list of parameters which can be set in this file and their meanings.

- [NO]CL Indicates to generate a CL file or not.
- [NO]OP Indicates to output a listing file (.pr) containing generated data and error and warning messages or not.
- [NO]PP Indicates to run the post processor automatically or not.
- [NO]AS Indicates to output an APT file or not.
- [NO]LI Indicates the listing file (.pr) will contain the input part program as well as the generated data or not.
- [NO]UP Indicates to update the part program with any changes kept at the end of the run or not.
- [NO]MO Indicates to display the **Ncq** Monitor window automatically or not.
- PR:x Indicates priority (A=highest, Z=lowest).
- NL:xx Indicates the number (xx) of lines per page in the listing file (.pr).

This file is also used to specify the default file extensions and descriptions for the “File - Open” browser window. Lines in this file that begin with the dash character (-) are assumed to be option definitions. All other lines are considered default file extensions and descriptions. Multi-lines are allowed to specify for multi-choices. The format of the file extension lines is as follows:

file_ext1, file_ext2; description

A comma is used to separate file extensions. A semi-colon is used to separate the file extensions from the description.

Example:

pp, mot; Part Program Files
geo; Geometry Build Files
ncl; NCL Files

G.16.8 user.init

This file contains parameters which can be used to override the default conditions specified in the *ncl.init* file. The following is a list of parameters which can be added to this file and their meanings.

UL_NCL_EDIT = notepad

This parameter specifies the interactive session default text editor. This example shows “notepad” is the default text editor.

UV_VPORT_DISPLAY = SHADED

This parameter defines the default state of shading. This can have the value “SHADED”, “WIREFRAME” or “HIDDEN”.

NCL_INCDIR = D:\MACROS

This parameter defines the default directory for include files. This example shows the default include file directory as “MACROS” on the “D” drive.

U_UNITS = INCH

This parameter defines the default units of **NCL**. This can either be “INCH” or “MM”.

UZ_CAM_KEYDEFS = %NCL_INIT_FILES\nclkeys.kdf

This parameter defines the default hotkey definition file. This example shows the file “nclkeys.sgi” under the path “%NCL_INIT_FILES” as the default hotkey definition file.

UZ_MOUSE_BUTDEFS = %NCL_INIT_FILES\mouse_default.mo

This parameter defines the default mouse buttons definition file. This example shows the file “mouse_default.mo” under the path “%NCL_INIT_FILES” as the default mouse buttons definition file.

NCL_TEMP_CLFILE = file_name

This parameter defines the default name of the temporary clfile to create. The string “%s” can be specified within this file_name. This causes the name of the active part program file to be inserted into the temporary clfile name as shown in the following example.

NCL_TEMP_CLFILE = %s_temp
Part Program file = sample.pp
Temporary clfile = sample_temp.cl

G17**NCL Modal Files**

The modals files contain parameters which control the modal setting of **NCL**. All the parameters in these files can be edited by the user manually and most of the setting can be changed through the interactive interface. Following is a list of common parameters for each file that can be modified by the user and their meaning.

G17.1**ncl.mod****#INCLUDE#**

This section defines the names of all the corresponding modal files. The name of the corresponding modal files must be as is and cannot be changed.

#MACROS#

This section defines the default conditions of Dynamic Macro Calls.

/MODAL/ *NO

Defines the modal state of macro calls. The Macro form will be redisplayed so that it can be called again with new parameters after initially calling the Macro when this parameter is set to *YES. Setting it to *NO causes **NCL** to return to its previous state after calling the Macro.

/DEFAULT/ *NO

The Macro CALL statement will contain all Macro arguments, even if they are the same as the default values defined in the 'macro=MACRO' command by setting this parameter to *YES. Setting it to *NO will cause the Macro CALL statement to leave off any Macro arguments that are set to their default values.

/VALUES/ *NO

When this parameter is set to *YES **NCL** will remember the values input for the Macro arguments and the next time this Macro is dynamically called, then these values will be displayed as the default values in the form. Normally the default values as defined in the 'macro=MACRO' command are used. The default for this field can be set using the 'PROMPT/macro,...,DEFALT/RETAIN' command.

#MATERIALS#

This section defines the material properties that are assigned to surfaces for lighting purposes. Each surface is assigned one of the sixteen available materials. The material properties defined in this file determine how the surface looks when it is shaded.

/TYPE/ 1

This parameter defines the material types to be defined by the user. Up to 16 materials type definition are allowed.

/NAME/ Default

Defines the name of the material currently being defined. Predefined names are:

Type Number	Name	Type Number	Name
1	Default	2	Plastic
3	Aluminum	4	Steel
5	Titanium	6	Chrome
7	Stone	8	Rubber
9	Brass	10	Copper
11	Gold	12	Bronze
13	Pewter	14	Carbon
15	Emerald	16	Jade

The name can also be defined by the user.

Note: Predefined Type 1 to Type 8 materials can be applied to any color and only change the way the surface is shaded. Predefined Type 9 to Type 16 materials should only be applied to surfaces that are *White* or *Grey* in color as they will actually modify the color of the surface.

/AMBIENT/ 1.0000

Defines the ambience or glow of the material. This is the light that emits from the surface even if the physical light does not shine exactly on it. This value can be from 0 to 1 and is a percentage of the surface color, so that the ambience color is always a shade of the surface color.

/DIFFUSE/ 0.65

Defines the diffuse property of the material, which is the light that is reflected off of the material by the active lights. Setting a high diffuse factor and a low ambient factor will result in heavily shaded surfaces similar to the real world, but not always appropriate for the screen. This value can be from 0 to 1 and is a percentage of the surface color, so that the diffuse color is always a shade of the surface color.

/SPECULAR/

Defines the specular property of the material, which produces highlights on the surface when directly hit by a light source. This value can be from 0 to 1 and is a percentage of the specular color.

/S_COLOR/ 0.0000, 0.0000, 0.0000

Defines the highlight color for the specular property. Each value defines the Red, Green, and Blue color components and can be between 0 and 1.

/EXPONENT/ 20.0000

The Exponent value determines the size and brightness of the highlight created by the specular property. The higher the exponent, the smaller and brighter the highlights. The exponent can be in the range of 0 to 128.

#GEO_COLOR#

This section defines the default color for every geometry type.

/POINT/ *RED

Defines the default color for points. The **NCL** default color is Red.

/LINE/ *GREEN

Defines the default color for lines. The **NCL** default color is Green.

/CIRCLE/ *BLUE

Defines the default color for circles. The **NCL** default color is Blue.

/PLANE/ *GREEN

Defines the default color for planes. The **NCL** default color is Green.

/VECTOR/ *RED

Defines the default color for vectors. The **NCL** default color is Red.

/PNTVEC/ *RED

Defines the default color for point vectors. The **NCL** default color is Red.

/MATRIX/ *MAGNTA

Defines the default color for matrices. The **NCL** default color is Magenta.

/PATERN/ *YELLOW

Defines the default color for patterns. The **NCL** default color is Yellow

/SHAPE/ *MAGNTA

Defines the default color for shapes. The **NCL** default color is Magenta.

/NCL_CV/ *CYAN

Defines the default color for **NCL** curves. The **NCL** default color is Cyan.

/SPLINE/ *CYAN

Defines the default color for splines. The **NCL** default color is Cyan.

/COMP_CV/ *YELLOW

Defines the default color for composite curves. The **NCL** default color is Yellow.

/SSPLINE/ *SEAGRN

Defines the default color for surface splines. The **NCL** default color is Seagreen.

/NCL_SURF/ *MAGNTA

Defines the default color for **NCL** Surfaces. The **NCL** default color is Magenta.

/NURBS_SUR/ *MAGNTA

Defines the default color for NURBS surfaces. The **NCL** default color is Magenta.

/TRIM_SURF/ *MAGNTA

Defines the default color for trimmed surfaces. The **NCL** default color is Magenta.

/REV_SURF/ *MAGNTA

Defines the default color for surfaces of revolution. The **NCL** default color is Magenta.

/NET_SURF/ *BROWN

Defines the default color for Net surfaces. The **NCL** default color is Brown.

/SOLID/ *SEAGRN

Defines the default color for Solids. The **NCL** default color is Seagreen.

/MAXIS/ *YELLOW

Defines the default color for the Modeling axis. The **NCL** default color is Yellow.

/WAXIS/ *CYAN

Defines the default color for the Working Plane axis. The **NCL** default color is Cyan.

#LIGHTS#

This section is used to define the active lights and their properties. The active lights and their properties in addition to the material properties determine how surfaces are shaded.

/LIGHT/ 1

This parameter determines which light is currently being defined. Up to 5 lights can be defined and active at any given time.

/ACTIVE/ *YES

Determines if this light is active or not. The value can be “*YES” or “*NO”.

/TYPE/ *DIRECT

Defines the type of light being defined. “*DIRECT” lights are infinitely far away from the scene and will light up the entire scene. An example of an infinite light is the sun. “*POINT” lights can be positioned by the user, they display light in all directions, similar to a light bulb. “*SPOT” lights can also be positioned by the user, they only display light in a user defined direction.

/SPACE/ *SCREEN

Point and Spot lights can be positioned in reference to “*MODEL” Space or “*SCREEN” Space. When they are positioned in Model Space, they are moved with the model as it is rotated/translated. In Screen Space, the lights do not move with the model, but rather stay fixed to the screen viewport. Screen Space coordinates are specified as a percentage of the viewport size, therefore they always remain in the viewport.

/COLOR/ *WHITE

Defines the color of the light. This color affects the Diffuse and Specular attributes of the surface material.

/INTENSITY/ 80

Defines the intensity or brightness of the light. It can be in the range of 0 to 100 percent.

/POSITION/ 0,0,1

Defines the position (X, Y, Z) of the light for Point and Spot lights.

/CONE/ 30

Sets the angle between the axis of the cone and the outside of the cone for Spot lights. Only the surface areas within this cone will be affected by this light.

/DIRECTION/ 0,0,1

Defines the vector direction (I, J, K) of Directional and Spot lights.

/AMBIENT/ 0.2

The Ambient value determines the amount of ambience this light emits. This value affects the ambient property of the surface material, so that a lower value causes less of the ambient light of the material to be generated. The Ambient value can be between 0 (no ambient light) and 1.

Note:

- All the color parameters in this file can have any one of the standard colors shown below or any of the 48 customized colors defined in the *ncl_color.mod* file.

*BLACK	*WHITE	*BLUE	*RED
*GREEN	*MAGNTA	*CYAN	*YELLOW
*BROWN	*LTTAN	*LTBLUE	*SEAGRN
*ORANGE	*PINK	*PURPLE	*GREY

G17.2 ncl_autosave.mod

This file controls the automatic save parameters for both Unibase and part programs during an **NCL** session:

/AUTO_U/ *ON

Defines the Unibase autosave mode. It can be “*ON” or “*OFF”.

/TYPE_U/ *TIME

Defines the Unibase autosave is based on either an elapsed “*TIME” in minutes or number of “*CHANGES” made to the4 file.

/INTERVAL_U/ n

“n” defines the number of minutes or number of changes value for autosave Unibase

/AUTO_PP/ *ON

Defines the part program autosave mode. It can be “*ON” or “*OFF”.

/TYPE_PP/ *TIME

Defines the part program autosave is based on either an elapsed “*TIME” in minutes or number of “*CHANGES” made to the4 file.

/INTERVAL_PP/ n

“n” defines the number of minutes or number of changes value for autosave part program.

/MAX_PP/ n

“n” defines the maximum number of file versions to maintain during the active session.

G17.3 ncl_background.mod

This file controls the background display in the **NCL** window.

/SHADE/ *GRADUATE

Defines which background shader to use. *SOLID uses a solid color, *GRADUATE varies smoothly from the top color to the bottom color, and 4_CORNER varies smoothly from colors defined for each corner of the window.

/COLOR/ *BLACK

Defines the *SOLID background color.

/RGB/ 0, 0, 0

Defines the Red, Green, and Blue color components to use for the *SOLID background color if *RGB is specified. These values can be between 0 and 1.

/TOP_COLOR/ *RGB

Defines the color to use at the top of the screen if *GRADUATE is specified as the background shade. The color of the background will vary smoothly between this color and a different color specified for the bottom of the screen.

/BOT_COLOR /*RGB

Defines the color to use at the bottom of the screen if *GRADUATE is specified as the background shade. The color of the background will vary smoothly between this color and a different color specified for the top of the screen.

/TOP_RGB/ 0, 0.43, 0.6

Defines the Red, Green, and Blue color components to use for the *GRADUATE shade top color if *RGB is specified. These values can be between 0 and 1.

/BOT_RGB/ 0.1, 0.1, 0.2

Defines the Red, Green, and Blue color components to use for the *GRADUATE shade bottom color if *RGB is specified. These values can be between 0 and 1.

/UL_COLOR/ *RGB

Determines the style for recreating the graphics behind a displayed cutter after it is moved or erased.

/UR_COLOR/ *LTtan

Determines the style for recreating the graphics behind a displayed cutter after it is moved or erased.

/LL_COLOR/ *LTTAN

Determines the style for recreating the graphics behind a displayed cutter after it is moved or erased.

/LR_COLOR/ *RGB

Determines the style for recreating the graphics behind a displayed cutter after it is moved or erased.

/UL_RGB/ 0.4, 0.2, 0

Defines the Red, Green, and Blue color components to use for the *4_CORNER shade upper left corner color if *RGB is specified. These values can be between 0 and 1.

/UR_RGB/ 0.618, 0.528, 0.408

Determines the style for recreating the graphics behind a displayed cutter after it is moved or erased.

/LL_RGB/ 0.618, 0.528, 0.408

Determines the style for recreating the graphics behind a displayed cutter after it is moved or erased.

/LR_RGB/ 0.4, 0.2, 0

Determines the style for recreating the graphics behind a displayed cutter after it is moved or erased.

Note:

- All the color parameters in this file can be any one of the standard colors shown below or any of the 48 customized colors defined in the *ncl_color.mod* file.

*BLACK	*WHITE	*BLUE	*RED
*GREEN	*MAGNTA	*CYAN	*YELLOW
*BROWN	*LTTAN	*LTBLUE	*SEAGRN
*ORANGE	*PINK	*PURPLE	*GREY

G.17.4 ncl_chain.mod

This file defines the process of chain selecting wireframe entities (lines, circles, curves).

/CHAIN/ *NORMAL

When an intersection is reached during chain selection and there are two or more possible curves to choose from, then the behavior can be set to either automatically select the first curve that is found (*NORMAL) or to prompt the user for the proper curve to select (*CONDITIONAL). When Conditional is set, the curves that can be selected from will be displayed in Red and the user simply selects the curve to be included in the chain selection, which will then continue along the selected path.

/TOL/ *0.01000

This parameter defines the tolerance value that determines how close the end points of two curves have to be in order to be considered connected for chaining purposes.

/PLANAR/ *NO

Set this field to “*YES” if all entities considered for chaining must lie on a plane. Entities that are not entirely lying on this plane will not be considered for chaining. Entering “*NO” will allow the chaining operation to select any curves that have a matching end point, whether they are planar or not.

/PLANE/ 0, 0, 1, 0

Contains either the name of the plane to use or the canonical form of the plane (i,j,k,d) when the Planar field is set to “*YES”.

/LINES/ *INCLUDE

Lines can either be “*INCLUDE”d or “*EXCLUDE”d from the chaining operation.

/CIRCLE/ *INCLUDE

Circles can either be “*INCLUDE”d or “*EXCLUDE”d from the chaining operation.

/BSPLINES/ *INCLUDE

Curves can either be “*INCLUDE”d or “*EXCLUDE”d from the chaining operation. The supported curve types include B-splines, **NCL** Curves, Conics, and Surface-splines.

/COMPCURVES/ *INCLUDE

Composite curves can either be “*INCLUDE”d or “*EXCLUDE”d from the chaining operation.

/SURFTOL/ 0.005

This defines the tolerance value that determines how close the boundaries of two surfaces have to be in order to be considered connected for surface chaining purposes.

/VECTORS/ 4

This defines the number of directions to be displayed on the first surface for chaining and it can be set to any positive integer value.

/DIRECTION/ *EDGE

This defines the chaining direction could be along the U or V lines of the base surface (*UV), along the direction of chaining from the previous surface (*PREVIOUS), along the direction perpendicular to the boundary edge of the new surface (*PERPTO), or, along the boundary edge of the new surface (*EDGE).

/MULTI_PICK/ *YES

Set this field to “*YES” allow a surface can be selected multiple times during the chaining process. Set this field to “*NO” does not allow a surface to be picked multiple of times during the chaining process.

G.17.5 ncl_cmdline.mod

This file defines the command mode prompt default condition.

/LENGTH/ 72

Defines the maximum command line length (40 to 1026) that will be created when a command is generated using the menu interface. This

includes the command manually entered through the command prompt mode. It also controls the length of the line when formatting interface-generated commands. “72” is the default value.

Note: While it is permissible to set the command line length to be shorter than 72 characters it is not recommended, particularly when Source Formatting is in effect. Shorter command line lengths could result in lines of code being broken up in unpredictable ways and truncation of PPRINT and INSERT statements when written to the CL and APT source files.

/COMMENT/ 0

Defines the starting column for fixed comment text on input command lines. A value of “0” disables any fixed column comment text causing the entire input line to be processed as a command. “0” is the default value. This also indirectly specifies the maximum length of each command line, i.e. one less than the value specified for this parameter.

Note: While it is permissible to set the starting comment column to be shorter than 73 it is not recommended, particularly when Source Formatting is in effect. Shorter starting comment column could result in lines of code being broken up in unpredictable ways and truncation of PPRINT and INSERT statements when written to the CL and APT source files.

/EDIT/ *NO

Defines the edit mode is always on or not.

/WINDOW/ *NO

Defines the status window is always on or not.

/INSERT/ *SAME

This controls how interface generated commands are input to the part program file. “*ON” performs the same as *INSERT mode, in that new statements are added prior to the current part program statement. “*OFF” behaves the same as *CONSOL, where interface generated commands will overwrite the part program lines. “*SAME” will rely on the setting of *INSERT/*CONSOL to determine how interface commands are output to the part program file. This setting affects commands generated from the interface only, it does not affect commands typed in while in Command Mode.

G.17.6 ncl_color.mod

NCL supports 16 standard colors and a maximum of 48 customized color. This modal file specifies the definition of the 48 customized colors and has the following format.

```
#COLOR#
/NAME/ color_1
/RGB/ red,green,blue
/NAME/ color_2
/RGB/ red,green,blue
.....
.....
/NAME/ color_48
/RGB/ red,green,blue
```

where:

color_n	The customized color name
red, green, blue	Defines the Red, Green, and Blue color components of the corresponding customized color “color_n” and can be any value between 0 and 255.

G.17.7 ncl_graphic.mod

The modal file controls how dynamic style graphics are displayed, such as entity dragging, rubber banding, and dynamic viewing.

This form controls how the screen is refreshed after the graphics have been updated. Since different implementations of OpenGL can vary on the speed graphics is displayed (normally dependant on the graphics card installed), this setting allows the user to optimize the graphics display for their system.

/BUFFER/ *PIXEL

Determines the style to use when refreshing the graphics. *SWAP swaps the back buffer with the front buffer. In order to use this mode, the graphics in the back buffer must remain unchanged when swapping the buffers. Most graphics cards on Windows systems support this feature (it may be a setting such as "Force Copy Swap").

*PIXEL copies the affected area of the screen from the back to the front buffer. This method should work on all platforms, but may be slower than the "Swap Buffer" method on some platforms.

/ERASE/ *REDRAW

Determines how a geometric entity is to be erased. Either all entities that are in the same screen area of the erased entity can be redrawn (*REDRAW) or the entity being erased can be drawn in black (*ERASE).

Using the *REDRAW option provides for a much cleaner display, but will usually take longer to complete. Using the *ERASE option will leave black areas on the screen where the entity was erased, but can be much faster than the *REDRAW option depending on how many entities are defined.

/CUTTER/ *FAST

This field controls the display of the wireframe cutter. It can be displayed using a *FAST method, which will display the cutter much faster on most systems, but could cause a flashing effect of the cutter. *SMOOTH will display the wireframe cutter in a manner that generates a less jumpy display, but will take much longer (2 to 5 times) to animate on most systems.

A solid cutter will always be displayed using the *SMOOTH method, since it does not display well using the *FAST method.

G17.8 ncl_interface.mod

This file controls the Menu display attributes, command line text font and cursor settings, geometry label fonts, and the "form help" text font.

/AUTO_CURSOR/ *OFF

When Auto Cursor is enabled, then the cursors will automatically be positioned at the menu that is activated. When Auto Cursor is disabled, then the cursor will remain where it was when the new menu is activated.

/ICON_SIZE/ *32

Determines the displayed size of the menu bitmaps on permanent menus. Pull down menus are displayed with a bitmap size of 16x16 when the Icon Size is set to 32x32 or lower and with a bitmap size of 24x24 when the Icon

Size is set larger than 32x32. This field is only valid for Windows platforms.

/BROWSER/ *LOCAL

Defines the initial directory used when a file browser is brought up. Either the local directory (*LOCAL) that **NCL** is currently running in can be displayed or the last directory accessed (*SAVED) by an **NCL** file browser can be displayed.

/TEXT_FONT/ Arial

Defines the font name to be used for the command line text. A known font name should be used, if it is not a known font, then the default operating system font will be used. Typical fonts are Arial, Courier, Helvetica, and Times. This field is only valid for Windows platforms. If the font name has a space between it, the font name must be enclosed in a pair of double quotes, i.e. "font name".

/TEXT_SIZE/ *12

Defines the font point size of the command line text. The smaller the font size, the smaller the text will appear. Typical settings are 8, 10, or 12. This field is only valid for Windows platforms.

/TEXT_CURSOR/ *END

Specifies to position the cursor at the “*BEGIN”ning or “*END” of the command line when the command line is initiated.

/FORM_HIGHLIGHT/ *ON

Enabling the Auto Highlight field will cause the Form Field text to be automatically selected when the text field is first entered. Typing a single character into the Text Field when the text is selected/highlighted will overwrite the entire text line. Clicking in the Text Field with the mouse or entering a left or right arrow key will unselect the text.

Disabling this field will not automatically highlight the Form Field text, you will have to use normal operating system methods to manually highlight the text.

/TEXT_HIGHLIGHT/ *ON

Enabling the Auto Highlight field will cause the command line text to be automatically selected when it is first displayed and the lines are scrolled through. Typing a single character into the command line when the text is selected/highlighted will overwrite the entire text line. Clicking in the command line with the mouse or entering a left or right arrow key will unselect the text.

Disabling this field will not automatically highlight the command line text, you will have to use normal operating system methods to manually highlight the text.

/COM_KEYPAD/ *FUNCTION

This field determines how the Numeric Keypad will behave while in command mode. Selecting Numeric allows you to enter numbers using the Numeric Keypad. Selecting Functions will disable the Numeric Keypad for numeric input and instead use it to call **NCL** functions that have been assigned to its keys.

/FORM_HELP_FONT/ Courier

Defines the font name to use for the form help text. A known font name should be used, if it is not a known font, then the default operating system font will be used. Typical fonts are Arial, Courier, Helvetica, and Times. This field is only valid for Windows platforms. If the font name has a space between it, the font name must be enclosed in a pair of double quotes, i.e. “font name”.

/FORM_HELP_SIZE/ *12

Defines the font point size of the form help text. The smaller the font size, the smaller the text will appear. Typical settings are 8, 10, or 12. This field is only valid for Windows platforms.

/FORM_PICTURE_POS/ *OFF

“*OFF” will cause the tooltip text to display when the cursor is within a defined region of the picture. “ON” will display the location of the cursor within the picture, making it easier for the user to define the picture areas referenced in the form fields.

/STATUS_TEXT_FONT/ Courier

Defines the font name to be used for the Status Window text. A known font name should be used, if it is not a known font, then the default operating system font will be used. Typical fonts are Arial, Courier, Helvetica, and Times. This field is only valid for Windows platforms. If the font name has a space between it, the font name must be enclosed in a pair of double quotes, i.e. "font name".

/STATUS_TEXT_SIZE/ *12

Defines the font point size of the Status Window text. The smaller the font size, the smaller the text will appear. Typical settings are 8, 10, or 12. This field is only valid for Windows platforms.

/PLABEL_FONT/ Courier

Defines the font name to be used for the Prompt Area text. A known font name should be used, if it is not a known font, then the default operating system font will be used. Typical fonts are Arial, Courier, Helvetica, and Times. This field is only valid for Windows platforms. If the font name has a space between it, the font name must be enclosed in a pair of double quotes, i.e. "font name".

/PLABEL_SIZE/ *12

Defines the font point size of the Prompt Area text. The smaller the font size, the smaller the text will appear. Typical settings are 8, 10, or 12. This field is only valid for Windows platforms.

/ELABEL_FONT/ Courier

Defines the font name to be used for the Error Area text. A known font name should be used, if it is not a known font, then the default operating system font will be used. Typical fonts are Arial, Courier, Helvetica, and Times. This field is only valid for Windows platforms. If the font name has a space between it, the font name must be enclosed in a pair of double quotes, i.e. "font name".

/ELABEL_SIZE/ *12

Defines the font point size of the Error Area text. The smaller the font size, the smaller the text will appear. Typical settings are 8, 10, or 12. This field is only valid for Windows platforms.

/LIVE_MOUSE/ *ON

“*ON” enables or “*OFF” disables the usage of the Live Mouse. When the Live Mouse is active then geometry will automatically be highlighted when the cursor is placed over it and the mouse buttons can be programmed to activate certain functions as defined in the *UZ_MOUSE_BUTDEFS* mouse definition file.

When the Live Mouse is not active then while in choice mode, the geometry will not be highlighted and pressing the mouse buttons will not perform any action. This mode is convenient when there is so much geometry on the screen that it causes **NCL** to slow down whenever the mouse is moved.

/ACTIVE LINE/ *PINK

Defines the highlight color of the active line while in multi-line command mode.

/CURRENT_LINE/ *YELLOW

Defines the highlight color of the line that the cursor is on if the line is not the active line.

Note:

- All the color parameters in this file can be any one of the standard colors shown below or any of the 48 customized colors defined in the *ncl_color.mod* file.

*BLACK	*WHITE	*BLUE	*RED
*GREEN	*MAGNTA	*CYAN	*YELLOW
*BROWN	*LTAN	*LTBLUE	*SEAGRN
*ORANGE	*PINK	*PURPLE	*GREY

G17.9 ncl_labels.mod

This file controls the display of labels.

/LABELS/ *OFF

When Label Text is enabled (*ON), the automatic labeling feature is turned on. Once set all subsequent geometric entities will be labeled as they are defined. The other choice is *OFF.

/LABEL_COLOR/ *BLACK

Defines the color of the geometry labels.

/LABEL_SIZE/ 12, 5

Defines the font height and width in pixels for the displayed geometry labels.

/OVERLAP_DISTANCE/ 6

Defines the minimum distance in pixels between the labels to avoid overlap. A typical value is 6.

/BACKGROUND/ *ON

When Label Background is enabled (*ON), a background box is displayed behind every displayed label. When Label Background is disabled (*OFF), the labels are displayed without this box.

/BACKGROUND_COLOR/ *BLACK

Defines the color of the label background.

/LEADER_LINES/ *OFF

When Leader Lines are enabled (*ON), the automatic leader lines feature is turned on. Once set all subsequent geometric entities will have leader lines (if altered) as they are defined. The other choice is *OFF.

/LEADER_COLOR/ *WHITE

Defines the color of the label leader line.

/LEADER_ARROW/ *OFF

When Leader Line Arrows are enabled (*ON), a arrow head is displayed on the leader line pointing towards the entity. When Leader Line Arrows are disabled (*OFF), the leader lines are displayed without this arrow head.

/OUTPUT_CMD/ *YES

Defines to output or not to output the DRAFT/LABEL command to the part program.

Note:

- All the color parameters in this file can be any one of the standard colors shown below or any of the 48 customized colors defined in the *ncl_color.mod* file.

*BLACK	*WHITE	*BLUE	*RED
*GREEN	*MAGNTA	*CYAN	*YELLOW
*BROWN	*LTTAN	*LTBLUE	*SEAGRN
*ORANGE	*PINK	*PURPLE	*GREY

G17.10 ncl_motion.mod

This file defines the modal settings of motion for motion display and playback. By default the contents of this file is included under the #MOTION# section of the ncl.mod file.

/CUTTER_COLOR/ *YELLOW

This parameter defines the default color of the displayed cutter.

/CUTTER_PEN/ 1

Defines the pen number (between 1 and 256) to be associated with the cutter shapes when projected onto a drawing.

/CUTTER_SHADED/ *YES

Defines the cutter to be displayed as shaded (*YES) or not shaded (*NO).

/CUTTER_TRANS/ 100

Defines the cutter translucency in shaded mode. This value can be in the range of 1 to 100.

/SHANK_COLOR/ *ORANGE

This parameter defines the default color of the displayed shank portion of the cutter.

/SHANK_PEN/ 1

Defines the pen number (between 1 and 256) to be associated with the cutter shanks when projected onto a drawing.

/SHANK_SHADED/ *YES

Defines the shank to be displayed as shaded (*YES) or not shaded (*NO).

/SHANK_TRANS/ 100

Defines the shank translucency in shaded mode. This value can be in the range of 1 to 100.

/HOLDER_COLOR/ *PURPLE

This parameter defines the default color of the displayed holder portion of the cutter.

/HOLDER_PEN/ 1

Defines the pen number (between 1 and 256) to be associated with the cutter holders when projected onto a drawing.

/HOLDER_SHADED/ *YES

Defines the holder to be displayed as shaded (*YES) or not shaded (*NO).

/HOLDER_TRANS/ 100

Defines the holder translucency in shaded mode. This value can be in the range of 1 to 100.

/MOTION_COLOR/ *WHITE

This parameter defines the default color of the displayed motion that is programmed with a feedrate rather than in rapid mode.

/MOTION_STYLE/ *SOLID

This parameter defines the default line type of the displayed motion that is programmed with a feedrate rather than in rapid mode.

/MOTION_PEN/ 1

Defines the pen number (between 1 and 256) to be associated with the motion when projected onto a drawing.

/RAPID_COLOR/ *RED

This parameter defines the default color of rapid motion.

/RAPID_STYLE/ *DASHED

This parameter defines the default line type of the displayed motion that is programmed in rapid mode.

/RAPID_PEN/ 1

Defines the pen number (between 1 and 256) to be associated with the rapid motion when projected onto a drawing.

/CUTTER_STEP/ 999

This parameter defines the number of motion steps to display a cutter when static cutter display is in effect (non-moving cutter). The cutter will always be displayed at the end of a motion regardless of this value.

/TRACUT/ *IGNORE

This parameter determines if the displayed motion should honor the active TRACUT matrix. This can be either “*IGNORE” or “*APPLY”.

/CUTTER_ITERATE/ *NO

This parameter determines if the cutter visualized at each step of the programmed motion: “*YES”, or only be displayed at the end of a programmed motion: “*NO”. “*NO” will speed up the motion display. This field will only affect moving cutters, not static cutters. If the Motion Display Stack is active and Iterative Cutter Display is turned off, then the cutter can be visualized at each step using the Playback Current Motion form.

/STACK_SIZE/ 10

The Motion Display Stack keeps track of the previous 'n' motions. It is used with the Playback Current Motion form to allow for a quick playback of

the last programmed motions. A value of zero will disable this stack. Specifying a large value can increase memory storage dramatically.

WARNING - Changing the size of the Motion Display Stack will reinitialize it, deleting any motions that are currently on the stack.

/COMMANDS/ *NO

This parameter determines if the associated DRAFT and CUTTER/DISPLY commands will be output to the part program when there is a setting change in the Motion Display Modal Form.

Note:

- All the color parameters in this file can has any one of the standard colors shown below or any of the 48 customized color defined in the *ncl_color.mod* file.

*BLACK	*WHITE	*BLUE	*RED
*GREEN	*MAGNTA	*CYAN	*YELLOW
*BROWN	*LTTAN	*LTBLUE	*SEAGRN
*ORANGE	*PINK	*PURPLE	*GREY

- All the line style parameters in this file can has any one of the values shown below:

*SOLID	*SMALL_DASH	*DOTTED	*CENTER
*PHANTOM	*DASHED	*DASH_DOT	*DASH_SPACE

G17.11 ncl_pick.mod

This file define the default conditions of picking mode.

/APERTURE/ 0.0100

Defines the pick aperture zone in percentage of the screen area. This value can be in the range of 0.001 to 1.

/VERIFY_MODE/ *ON

Defines to turn on or turn off verify mode.

/MARKING/ *DYNAMIC

Defines the marking is dynamic or static.

/HIERARCHY/ *ON

Enable (ON) or disable (OFF) the entities picking hierarchy. If there are multiple entities at the same z-level within a small tolerance when enabled, then the picking order will be as follows:

1. Points
2. Point-Vectors
3. Vectors
4. Lines and Circles
5. Curves
6. Composite Curves
7. Matrices
8. Annotations
9. Symbols
10. Drafting
11. Planes
12. Surfaces and Solids
13. Net Surfaces

/HIGHLIGHT_COLOR/ *PINK

Define the default highlight color.

/VERIFY_COLOR/ *RED

Defines the default verify color.

Note:

- All the color parameters in this file can has any one of the standard colors shown below or any of the 48 customized color defined in the *ncl_color.mod* file.

*BLACK	*WHITE	*BLUE	*RED
*GREEN	*MAGNTA	*CYAN	*YELLOW
*BROWN	*LTTAN	*LTBLUE	*SEAGRN
*ORANGE	*PINK	*PURPLE	*GREY

G17.12 ncl_playfeed.mod

The file sets the colors of display motion in based on the feed rate value when playing back motion in **NCL** with Analyzation set to Feed Rate.

NCL/IPV will also use the color scheme defined in this form for displaying cuts when Analyzation is set to Feed Rate during **NCL/IPV** motion playback.

/FEED1/ 10.0

Defines feed rate ranges that control the color of the displayed motion. Whenever the feed rate is less than or equal to the feed rate, then the corresponding color will be used to display the motion. Any programmed feed rate that is greater than the highest feed rate specified in this form will be displayed using the color assigned to the highest feed rate. Up to 10 feed rate ranges definition are allowed.

/FEED1_COLOR/ *WHITE

Defines the color used to display the motion within the corresponding feed rate range.

Note:

All the color parameters in this file can be any one of the standard colors shown below or any of the 48 customized colors defined in the [*ncl_color.mod*](#) file.

*BLACK	*WHITE	*BLUE	*RED
*GREEN	*MAGNTA	*CYAN	*YELLOW
*BROWN	*LTAN	*LTBLUE	*SEAGRN
*ORANGE	*PINK	*PURPLE	*GREY

- Black color will display the motion as invisible.

G17.13 ncl_playinterp.mod

This file sets the colors and line types to display various motion types in when playing back motion in **NCL** with Analyzation set to Interpolation.

NCL/IPV will also use the color scheme defined in this form for displaying cuts when Analyzation is set to Interpolation during **NCL/IPV** motion playback. The geometry attributes and line styles are ignored in **NCL/IPV**.

The 'Default' settings will use the default geometry/motion colors/line-styles for the geometry or specified type of motion.

/GEO_COLOR/

Defines the color attributes to display the model with during motion playback. All geometry in the model will be displayed using these attributes during motion playback. The original attributes for each geometry will be restored when motion playback is exited.

/GEO_LINE/

Defines the line style attributes to display the model with during motion playback. All geometry in the model will be displayed using these attributes during motion playback. The original attributes for each geometry will be restored when motion playback is exited.

/LINEAR_COLOR/

Defines the color attribute to display fixed tool axis moves during motion playback.

/LINEAR_LINE/

Defines the line style attribute to display fixed tool axis moves during motion playback.

/CIRCLE_COLOR/

Defines the color attribute to display circular moves during motion playback.

/CIRCLE_COLOR/

Defines the line style attribute to display circular moves during motion playback.

/RAPID_COLOR/

Defines the color attributes to display rapid moves during motion playback.

/RAPID_COLOR/

Defines the line style attributes to display rapid moves during motion playback.

/CYCLE_COLOR/

Defines the color attribute to display canned cycle moves during motion playback.

/CYCLE_LINE/

Defines the line style attribute to display canned cycle moves during motion playback.

/TLAXIS_COLOR/

Defines the color attribute to display changing tool axis moves during motion playback.

/TLAXIS_LINE/

Defines the line style attribute to display changing tool axis moves during motion playback.

Note:

- All the color parameters in this file can be any one of the standard colors shown below or any of the 48 customized colors defined in the *ncl_color.mod* file.

*BLACK	*WHITE	*BLUE	*RED
*GREEN	*MAGNTA	*CYAN	*YELLOW
*BROWN	*LTTAN	*LTBLUE	*SEAGRN
*ORANGE	*PINK	*PURPLE	*GREY

- Black color will display the motion as invisible.
- All the line style parameters in this file can has any one of the values shown below:

*SOLID	*DASHED	*DOTTED	*CENTER
*PHANTM	*DASHLN	*DASHDT	*DASHSP

G.17.14 ncl_source.mod

This file controls the format of the statements written to the part program in an interactive session.

/FORMAT/ *YES

Defines the statements output to the part program should be formatted or not.

/MAJOR_CASE/ *SAME

Defines the case of all major words output to the part program. *SAME means the case will be exactly the same as what you entered and is the default. However, all major words generated by the interface icon menu will be in upper case. *UP means all the major words will be output to the part program in upper case regardless of the case entered at the command line. *LOW means all the major words will be output to the part program in lower case.

/LABEL_CASE/ *SAME

Defines the case of all labels output to the part program. *SAME means the case will be exactly the same as what you entered and is the default. However, all labels generated by the interface icon menu will all output as upper case. *UP means all the labels will be output to the part program in upper case. *LOW means all the labels will be output to the part program in lower case.

/VOCAB_CASE/ *SAME

Defines the case of all vocabulary words (except the major words) output to the part program. *SAME means the case will be exactly the same as what you entered and is the default. However, all vocabulary words generated by the interface icon menu will be output as upper case. *UP means all the vocabulary words will be output to the part program in upper case. *LOW means all the vocabulary words will be output to the part program in lower case.

/ACCURACY/ 0

Defines the number of digits to the right of the decimal output to the part program for real numbers. A value of “0” means trailing zeros will be removed. “0” is the default setting.

/ALIGNMENT/ 0

Defines the minimum amount of space each parameter in the command line will use. This works similar to the *<Tab>* key. For example, a value of

10 will right justify all the parameters to the 10th column. A value of “0” means there is no minimum amount of space. “0” is the default setting.

/INDENT_ALL/ 0

Defines the column number that the statement should be started. A value of “0” means no indentation specified. “0” is the default setting.

/INDENT_SEP/ 0

Defines the column number that the statement first separator should be located. A value of “0” means no separator indentation specified. “0” is the default setting.

G.17.15 ncl_srfatt.mod

This file controls the surfaces attributes.

/U_PATHS/ 5

Defines the number of iso-parametric curves to display in the U direction of the surface. This can be any integer number between 2 and 200.

/U_PTS/ 0

Defines the number of points used to display the U iso-parametric curves. This can be any integer number between 0 and 200. A value of 0 will display the U-curves using the current display tolerance and is the recommended value.

/V_PATHS/ 5

Defines the number of iso-parametric curves to display in the V direction of the surface. This can be any integer number between 2 and 200.

/V_PTS/ 0

Defines the number of points used to display the V iso-parametric curves. This can be any integer number between 0 and 200. A value of 0 will display the V-curves using the current display tolerance and is the recommended value.

/MATERIAL/ Default

Defines the material of the surfaces. It can be any name specified in the #MATERIALS# section of the file ncl.mod

/EDGE_DISPLAY/ *YES

Defines the surface edges to be displayed (*YES) or not displayed (*NO).

/EDGE_COLOR/ *BLACK

Defines the color of the surface edges. This parameter can be any one of the standard colors shown below, or any of the 48 customized colors defined in the *ncl_color.mod* file.

*BLACK	*WHITE	*BLUE	*RED
*GREEN	*MAGNTA	*CYAN	*YELLOW
*BROWN	*LTTAN	*LTBLUE	*SEAGRN
*ORANGE	*PINK	*PURPLE	*GREY
*DEFAULT			

/SHADED/ *YES

Defines the surfaces to be displayed as shaded (*YES) or not shaded (*NO).

/TRANSLUCENCY/ 100

Defines the surface translucency in shaded mode. This value can be in the range of 1 to 100.

G17.16 ncl_unibase.mod

This file defines: a) the option for saving the surface variable lists for displaying wireframe or shaded surfaces; b) the option to restore the original light and material settings after loading a unibase.

/SAVE_DISPLAY/ *YES

Defines to save or not to save the list of points and segments used to display the wireframe model of a surface. “*YES” specifies to save these variables in the unibase, the unibase becomes larger in size and the loading

is faster. “*NO” specifies not to save these variables in the unibase, the unibase file becomes smaller in size and the loading is slower.

/SAVE_TESSEL/ *YES

Defines to save or not to save the list of vertices and normals for the shaded surface to be displayed. “*YES” specifies to save these variable in the unibase, the unibase becomes larger in size and the loading is faster. “*NO” specifies not to save these variable in the unibase, the unibase becomes smaller in size and the loading is slower.

/RESTORE_LIGHT/ *YES

“*YES” specifies the light settings defined in the “#LIGHTS#” section of the ncl.mod file will be used when the unibase is loaded. “*NO” specifies the color settings stored in the unibase will be used.

/RESTORE_MATERIAL/ *YES

“*YES” specifies the material settings defined in the “#MATERIAL#” section of the ncl.mod file will be used when the unibase is loaded. “*NO” specifies the material settings stored in the unibase will be used.

/RESTORE_COLOR/ *MERGE

“*YES” specifies the original color settings are not overwritten by the Unibase color settings when it is loaded, rather the color settings in effect prior to loading the Unibase will be retained. “*NO” specifies the color settings stored in the unibase will be used. “*MERGE” specifies the original color settings are stay and the new custom colors will be appended to the end of the original colors.

/RESTORE_UNITS/ *YES

“*YES’ specifies to maintain the Units setting (Inches/MM) that was in effect prior to loading a Unibase using the interface. “*NO” specifies to take on the Units of the Unibase being loaded.

G17.17 ncl_view.mod

This file defines the default settings used during dynamic viewing.

/DISPLAY/ *PART

*PART defines to view the part during dynamic viewing. *AXIS defines to view the axes during dynamic viewing. *BOTH defines to view the part and the axes during dynamic viewing.

/CENTER/ *VIEWPORT

*VIEWPORT defines the view oriented about the viewport center during dynamic viewing. *USER defines the view oriented about a user defined location dynamic viewing. *AUTO will use the mouse cursor position at the time that a mouse button is pressed as the dynamic viewing center location, allowing for the view center to be easily manipulated during dynamic viewing.

The user defined location is selected by pressing the KEY_FUNCTION key during dynamic viewing. You will then be prompted to select the dynamic center point. The default user defined location is the viewport center before the KEY_FUNCTION is applied after **NCL** starts.

/Z_CENTER/ *PICK

“*PICK” specifies to attempt to pick an entity and use it for the center location. If an entity cannot be picked it will calculate the Z-level based on the geometry displayed in the window. “*CALC” specifies to bypass the picking of the geometry and go straight to the calculation of the Z-level based on the geometry displayed in the window. “*OFF” specifies to work as it did in previous versions prior to **NCL** 99, it will not calculate the Z-level center but use the reference center Z-level.

/STATUS_LINE/ *ON

“*ON” specifies the status Line can be updated with the current rotation, viewport center, or zoom factor whenever dynamic viewing other than Mouse Viewing is enabled, such as XY-Rotate, Zoom, Pan, etc. “*OFF” specifies the status Line will not be updated.

/SEGMENT/ 10000

Some models can be quite large and take a longer time to manipulate than what is desired. This field allows you to limit the maximum number of geometry entities that can be displayed during dynamic viewing so that the display is quicker.

/MS_PAN_GAIN/ 0.03

This defines how much the view will change with each pan movement of the mouse during dynamic viewing.

/MS_ROTATE_GAIN/ 3.00

This defines how much the view will change with each rotate movement of the mouse during dynamic viewing.

/MS_WHEEL/ *OFF

This defines should the mouse wheel disabled dynamic viewing. “OFF” means disables the mouse wheel and “ON” means do not disable the mouse wheel.

/MS_ZOOM_GAIN/ 0.03

This defines how much the view will change with each zoom movement of the mouse during dynamic viewing.

/KB_PAN_GAIN/ 0.01

This defines how much the view will change with each pan movement of the key board during dynamic viewing.

/KB_ROTATE_GAIN/ 3.00

This defines how much the view will change with each rotate movement of the key board during dynamic viewing.

/KB_ZOOM_GAIN/ 0.03

This defines how much the view will change with each zoom movement of the keyboard during dynamic viewing.

/SM_PAN_GAIN/ 0.03

This defines how much the view will change with each pan movement of the space mouse during dynamic viewing.

/SM_ROTATE_GAIN/ 0.0005

This defines how much the view will change with each rotate movement of the space mouse during dynamic viewing.

/SM_ZOOM_GAIN/ 0.0005

This defines how much the view will change with each zoom movement of the space mouse during dynamic viewing.

G17.18 ncliges.mod

The file determines the **NCL/IGES** default conversion options such as automatic naming conventions for all entities, what type of entities should be translated or not translated, the level of matching and output the non-matched secondary unibase entities or not.

The following section defines the names of the color modal file. The name of the color modal file must be as is and cannot be changed.

```
#INCLUDE#
/FILE/ ncliges_color.mod
```

The following section defines the automatic naming conventions for naming entities.

```
#GEOMETRY_NAME_MODALS#
/POINTS/PT,      *NO      or      subscript_label, *YES
/PNTVECS/PV,    *NO
/LINES/LN,       *NO
/VECTORS/VE,    *NO
/PLANES/PL,     *NO
/CIRCLES/CI,    *NO
/CURVES/CV,     *NO
/SURFACES/SF,   *NO
/SHAPES/SH,     *NO
/MATRICES/MX,   *NO
/PATTERNS/PN,   *NO
```

The following section defines the filter of translation, i.e. what type of entities should be translated or not translated.

```
#ENTITY_FILTER_MASK#
```

APPENDIX G**MISCELLANEOUS**

/100/CIRCULAR ARCS,	*YES	or *NO
/102/COMPOSITE CURVES,	*YES	
/104/CONICS,	*YES	
/106/POLYLINE 2D,	*YES	
/606/POLYLINE 3D,	*YES	
/906/POLYLINE 6D,	*YES	
/108/BOUNDED PLANE,	*YES	
/110/LINES,	*YES	
/112/PARAMETRIC SPLINES CURVES,	*YES	
/114/PARAMETRIC SURFACES,	*YES	
/116/POINTS,	*YES	
/118/RULED SURFACES,	*YES	
/120/SURFACES OF REVOLUTION,	*YES	
/122/TABULATED CYLINDERS,	*YES	
/126/NURB SPLINE CURVES,	*YES	
/128/NURB SURFACES,	*YES	
/140/OFFSET SURFACES,	*YES	
/141/BOUNDED CURVES,	*YES	
/142/SURFACE CURVES,	*YES	
/143/BOUNDED SURFACES,	*YES	
/144/TRIMMED SURFACES,	*YES	
/202/ANGULAR DIMENSIONS,	*YES	
/206/DIAMETER DIMENSIONS,	*YES	
/210/LABELS,	*YES	
/212/NOTES,	*YES	
/214/ARROWS,	*YES	
/216/LINEAR DIMENSIONS,	*YES	
/222/RADIUS DIMENSIONS,	*YES	
/228/SYMBOLS,	*YES	
/402/GROUP,	*YES	
/404/DRAWING,	*YES	
/406/NAME,	*YES	
/408/SUBFIGURE INSTANCE,	*YES	
/410/VIEW,	*YES	
/502/VERTEX,	*YES	
/602/ASSOCIATION,	*YES	
/907/POLYLINE CURVE	*NO	

The following section defines the default attributes of all the entities for the output unibase file.

#ATTRIBUTES#
/SHADED/ *YES

/INIT_LABELS/	*YES
/DUPLICATES/	*NO
/COLORS/	*IGES or *NCL
/CONVERSION/	*ALL or *LAYERS
/SURF_CURVE/	*CURVE or *COMP, *BOTH
/LABELS/	*IGES or *LABEL, *PROPERTY, *UNIBASE
/MAX_LABEL/	0
/SUBSCRIPTS/	*NO
/CONCATENATE/	*NO
/EDGE_DISPLAY/	*YES
/EDGE_COLOR/	*BLACK

The following section defines the level of label matching, matching tolerance and related items.

#LABEL_MATCHING#		
/LEVEL/	*EXACT	or *LEVEL_1, *LEVEL_2, *LEVEL_3, *LEVEL_4
/TOLER/	.001in	or mm for MM units
/LAYER_EXACT/	1	
/LAYER_1/	2	
/LAYER_2/	3	
/LAYER_3/	4	
/LAYER_4/	5	
/LAYER_MATCH/	6	
/LAYER_IMPORT/	7	
/COLOR_EXACT/	*AUTO	or any valid NCL color
/COLOR_1/	*AUTO	
/COLOR_2/	*AUTO	
/COLOR_3/	*AUTO	
/COLOR_4/	*AUTO	
/COLOR_MATCH/	*AUTO	
/COLOR_IMPORT/	*AUTO	
/IMPORT_GEO/	*NO	
/START-UNMATCH/	*NEXT	or *SECONDRY
/REGRESSIVE/	*YES	

G17.19 ncliges_color.mod

NCL/IGES supports 16 standard colors and a maximum of 48 customized color. This modal file specifies the definition of the 48 customized colors and has the following format.

```
#COLOR#
/NAME/ color_1
/RGB/ red,green,blue
/NAME/ color_2
/RGB/ red,green,blue
.....
.....
/NAME/ color_48
/RGB/ red,green,blue
```

where:

color_n	The customized color name
red,green,blue	Defines the Red, Green, and Blue color components of the corresponding customized color “color_n” and can be any value between 0 and 255.

G.17.20 nclstep.mod

The file determines the **NCL/STEP** default conversion options such as automatic naming conventions for all entities, what type of entities should be translated or not translated, the level of matching and output the non-matched secondary unibase entities or not.

The following section defines the names of the color modal file. The name of the color modal file must be as is and cannot be changed.

```
#INCLUDE#
/FILE/ nclstep_color.mod
```

The following section defines the automatic naming conventions for naming entities.

```
#GEOMETRY_NAME_MODAL#
/POINTS/PT,      *NO      or      subscript_label, *YES
/PNTVECS/PV,    *NO
/LINES/LN,      *NO
/VECTORS/VE,    *NO
/PLANES/PL,     *NO
/CIRCLES/CI,    *NO
/CURVES/CV,     *NO
/SURFACES/SF,   *NO
/SHAPES/SH,     *NO
```

```
/MATRICES/MX, *NO  
/PATTERNS/PN, *NO  
/SOLIDS/SO, *NO*
```

The following section defines the default attributes of all the entities for the output unibase file.

```
#ATTRIBUTES#  
/SHADED/ *YES  
/INIT_LABELS/ *YES  
/DUPLICATES/ *NO  
/COLORS/ *STEP or *NCL  
/CONVERSION/ *ALL or *COMPONENTS  
/LABELS/ *NCL or *COMPONENT, *FACE,  
*RECORD  
/SUBSCRIPTS/ *NO  
/CONCATENATE/ *NO  
/EDGE_DISPLAY/ *YES  
/EDGE_COLOR/ *BLACK  
/ON_LAYERS/ *NO  
/LAYER_START/ 1
```

The following section defines the level of label matching, matching tolerance and related items.

```
#LABEL_MATCHING#  
/LEVEL/ *EXACT or *LEVEL_1, *LEVEL_2,  
*LEVEL_3, *LEVEL_4  
/TOLER/ .001in or mm for MM units  
/LAYER_EXACT/ 1  
/LAYER_1/ 2  
/LAYER_2/ 3  
/LAYER_3/ 4  
/LAYER_4/ 5  
/LAYER_MATCH/ 6  
/LAYER_IMPORT/ 7  
/COLOR_EXACT/ *AUTO or any valid NCL color  
/COLOR_1/ *AUTO  
/COLOR_2/ *AUTO  
/COLOR_3/ *AUTO  
/COLOR_4/ *AUTO  
/COLOR_MATCH/ *AUTO  
/COLOR_IMPORT/ *AUTO
```

```
/IMPORT_GEO/      *NO
/START-UNMATCH/   *NEXT    or  *SECONDARY
/REGRESSIVE/      *YES
```

The following section defines the filter for surface curves translation.

```
#FILTER#
/CURVES          *NO
/COMPONENTS/     *NO
/SOLIDS/         *NO
```

G17.21 nclstep_color.mod

NCL/STEP supports 16 standard colors and a maximum of 48 customized color. This modal file specifies the definition of the 48 customized colors and has the following format.

```
#COLOR#
/NAME/ color_1
/RGB/ red,green,blue
/NAME/ color_2
/RGB/ red,green,blue
.....
.....
/NAME/ color_48
/RGB/ red,green,blue
```

where:

color_n	The customized color name
red,green,blue	Defines the Red, Green, and Blue color components of the corresponding customized color “color_n” and can be any value between 0 and 255.

G17.22 nclipv_*.mod

Refer to the [NCL/IPV reference manual](#) for detail description of all the nclipv_*.mod files.

INDEX**Symbols**

-	2-16, 2-18, 2-21, 2-45
"	1-2
\$	2-2, 7-31
\$\$	2-2, 7-31
%	2-3, 2-45, 2-46
&	2-16, 2-20, 2-44
()	2-16, 2-19, 2-21
*	2-16, 2-17, 2-21
**	2-16, 2-17, 2-21
** command	9-31
+	2-16, 2-18, 2-21, 2-45
,	2-16
.	2-16, 2-18, 2-45
/	2-16, 2-17
:	2-16, 2-17
=	2-16, 2-18
@	2-16, 2-20, 2-44, 7-11, 7-18
[]	1-2, 2-16, 2-20
[...]	1-2
{ }	2-51
'AND'	7-6, 7-8, 7-12
'EQ'	7-6, 7-8, 7-12
'GE'	7-6, 7-8, 7-12
'GT'	7-6, 7-8, 7-12
'LE'	7-6, 7-8, 7-12
'LT'	7-6, 7-8, 7-12
'NE'	7-6, 7-8, 7-12
'OR'	7-6, 7-8, 7-12
'...'	1-2
'NOT'	7-6, 7-9, 7-13

A

Abnormal Termination	7-34
ABS	2-29

INDEX

Absolute Value Function	2-29
ACOS	2-23
ADISPL	9-5, 9-12, 9-25
Advanced POCKET	6-113
ALL	2-9, 4-25, 5-5, 5-12, 8-11, 8-22, 9-9
Ampersand "&"	2-44
ANALYZ	4-1
Angle Forward	6-29
Angle Functions	2-24
Angle Right	6-29
ANGLF	2-24
ANNOTATIONS	3-420
ANOTE	2-12, 2-40, 3-420, 4-25
APTSRC	9-5, 9-13
ARC	3-7, 3-9, 3-14, 3-140, 3-149, 3-153, 6-59, 6-114, 6-124
ARCSLP	6-283
Arguments	2-19
ARROW	8-15
AS	5-3, 5-9
Ascending	5-4, 5-10
ASCII Unibase Files	5-2
ASIN	2-23
Assignment Statements	2-55
Asterisk "*"	2-17
AT	6-232, 6-293
ATAN	2-23
ATANGL	3-48, 3-91, 3-126, 3-164, 3-227, 6-31, 6-114
Attribute	8-4
Automatic Display of Geometry	9-5
Automatic Name Generation	2-12
Automatic Tool Start	9-7
AUTOPS	6-60
AUTOST	9-7, 9-17
AUTOUV	9-7, 9-18
AVOID	6-58, 6-105, 6-106, 6-117

B

Back and Forth	C-12
Barrel Cutter	6-4
BASE	9-23
Batch	1-4, 1-5

INDEX

Boundary Points	C-12
Bounded	3-2
BOX	C-18
Brackets "[]"	2-10, 2-20, 2-45

C

CALL	7-1, 7-14, 9-8
CAN	2-34, 3-383
CANON	2-55, 3-43, 4-1, 4-13, 9-7, 9-18
Canonical Form 2-34, 3-1, 3-8, 3-43, 3-51, 3-109, 3-174, 3-198, 3-251, 3-293, 3-333, 3-356, 3-383	
CASE	2-1, 2-48, 2-49, 9-8, 9-18
CBRTF	2-30
CCLW	3-149, 3-153
CENTER	2-9, 3-4, 3-31, 3-93, 3-164, 3-203, 3-204, 3-259, 3-260, 8-9
Check Surface	6-63, 6-205, 6-209
CHKPTS	6-285
CIRAPT	C-13
CIRCLE	2-9, 2-10, 2-12, 2-40, 3-1, 3-4, 3-7, 3-63, 3-120, 3-149, 3-203, 4-25, 4-49
CIRCUL	6-148, 6-303, 9-5, 9-13
Circular Interpolation	6-303
CLDIST	6-31
CLfile	1-5, 6-290, 6-292, 6-298, 6-302, 6-303, 6-304, 6-308, B-25
CLIPF	4-2
CLONE	4-4
CLOSE	4-26, 9-27, 9-30
CLW	3-149
COLF	4-3
Colon ":"	2-15, 2-17
Color	8-1
COMBIN	6-34, 6-283
Comma ","	2-16
Comments	2-2
Common Logarithm Function	2-30
COMPOS	3-63, 3-104
CONIC	3-58, 3-59, 3-61
CONSOL	9-1
CONST	6-58
CONTCT	2-34, 6-31, 6-39, 6-288
CONTIN	7-1, 7-3, 7-17
Continuous Path Motion	6-1, 6-60
Control Commands	9-1

INDEX

COPY	6-290, 6-298, 6-301
COS	2-22
COUPLE	6-115
CROSS	3-276, 3-342
Cube Root Function	2-30
CURVE	2-12, 2-40, 3-1, 3-25, 3-47, 3-129, 4-25, 4-49, 6-73, 6-153
Customized Screen Layout	G-34
Customized View	G-34
CUT	6-292
CUTTER	2-35, 2-44, 8-12, 8-22, 9-14, 9-25
CUTTER (Definition)	6-2
CUTTER/ BLADE	6-9
CUTTER/LATHE	6-10
CUTTER/TOOL	13-10, 13-14
CYCLE	6-115
CYLNDR	C-20

D

DATA	2-39, 2-40, 4-5
Data Fields	2-4, 2-9
DBFN (Obsolete)	C-9
DBSHOW	9-32, C-10
DE	9-1
Decimal Point	2-18
DECOMP	4-10
Decompose a Symbol	10-17
DECR	2-9
DEFALT	8-2, 8-4, 9-23
Default	7-2, 7-4, 7-15, 7-17, 7-20, 7-22, 7-32
DEFNAM	2-13, 2-44, 4-11
DELETE	9-1
Deleting Characters	9-2
Descending	5-4, 5-10
Direction Modifiers	2-6
DISPDB (Obsolete)	C-11
Display Point	3-175
DISPLAY	8-1, 8-20, 9-8, 9-19, B-25
DIST	2-26, 2-32
Distance Function	2-26
DISTF	2-26
Division	2-17

INDEX

DNTCUT	6-291
DNTCUT/NOMORE	6-291
DO	2-15, 7-2, 7-33
DOT	2-33
Dot Product Function	2-33
Double Asterisk "*"	2-17
Double Dollar Sign " \$\$"	2-2, 7-31
DOWN	6-84
Draft Commands	8-2
Drawing Entity	10-4
Drive Surface	6-61, 6-64, 6-73, 6-293
Dynamic Macro	7-19, 7-21, 7-24

E

ECHO (Obsolete)	C-11, C-13
ED	9-2
Edge	3-328
EDIT	9-1
EDT	9-2
ELIMIT	9-19
END	3-93, 3-164
ENDPT	3-223
ENGAGE	12-6
Entry Feedrate	6-102
Equal Sign "="	2-18
ERASE	8-1, 8-23
ERROR	7-17
Error Limit	9-19
Error Messages	A-1
ESC	5-6, 5-12
Evaluation Priority	2-19
EXPCL	9-8, 9-20
EXPF	2-31
Explicit Data Fields	2-9
Exponent Function	2-31
Exponentiation	2-17
Expressions	2-52
Extension	3-2

INDEX

F

FACE	9-23
FAN	6-32, 6-34, 6-36, 6-44, 6-210
FEDRAT	2-34, 6-293, 9-25
File	1-5, 2-60
FILES	9-25
FILLET	3-5, 3-45, 3-307, 6-283
FILTER	4-3
FIND	9-3
FINDEX Function	2-48
FINDTK	9-3
FINI	7-4
FINISH	6-205, 12-6
Finish	6-112, 6-127
FIT	3-52, 3-314
Fixed Field Words	2-1, 2-2, 2-3
Fixture	14-2
FMILL	6-84
FORMAT	7-4, 8-8
Format Function	2-45
Forward Sense	6-76, 6-77, 6-80, 9-27
FROM	6-45
FWD	3-118, 3-257, 3-338, 6-36, 6-84

G

GB	6-76
GD	6-55
GENPTS	4-12
Geometric Assignment Statements	2-56
Geometric Element	2-52
Geometric Expressions	2-53
Geometry (DISPLAY)	9-5
Geometry Type	4-21
GET	2-8, 5-2, 10-16
GF	6-77
GL	6-78
GO	6-45
GOBACK	6-76
GODLTA	6-55

INDEX

GODOWN	6-76
GOFWD	6-77
GOFWDA	6-105
GOLFT	6-78
GORGT	6-78
GOTO	6-57
GOUGCK	6-42, 6-296
Gouge Checking	6-296
GOUP	6-79
GR	6-78
GUIDE	6-39

H

Helical Radius	6-118
HELIX	6-115
HIDDEN	8-8
HIGH	9-15
HOLDER	2-36, 4-23, 8-12
Holder	13-26
Hot Keys	F-1

I

Id Number	6-298
Identifier	2-4, 2-11, 9-20, 9-25
IF	2-44
IF (arithmetic)	7-5
IF (logical)	7-5
IF-Then-Else	7-8
IFTOL	7-10
IGES	1-4, 1-6, 10-1
IGES Drawing	10-17
IGES Entity Mapping	10-20, 10-22
IGES View Entity	10-15
Implicit Data Fields	2-10
Implied Check Surface	6-63, 6-64
IN	2-7, 3-19, 3-21, 3-27, 3-321
INCLUD	2-2, 2-44, 7-10, 7-25, 7-31
INCR	2-9, 3-52, 3-145, 3-153, 3-160

INDEX

INDENT	9-9, 9-20
Indentation	9-9, 9-21
INDEX	4-11, 6-290, 6-298
INDIRP	6-46, 6-48, 6-80
INDIRV	6-46, 6-47, 6-48, 6-80
In-Process-Verification	14-1
INPUT	9-3
INS	9-3
INSERT	2-1, 7-4, 7-5, 7-11
Inserting Additional Characters	9-2
INT	2-33
Integer	2-21
Integer Function	2-33
Interactive	1-1, 1-4
INTERP	6-43, 6-44, 6-283
INTOF	3-102, 3-209, 3-225, 3-348
INTOF3	3-221
INVERS	3-368, 6-58, 12-6
INVIS	8-1, 8-24
IPM	6-293
IPR	6-293
IV	6-80

J

JUMPTO	2-44, 7-2, 7-7, 7-9, 7-12
Justified	2-45

L

LABEL	C-12, C-14
Label	2-4, 2-15, 2-40, 8-15
LARGE	2-7, 3-36, 6-303
Lathe Module	12-1
LAYER	5-5, 5-11, 8-23, 8-24
LAYF	4-4
LEADER	8-15
LEFT	2-7, 3-120, 3-122, 6-283
LENGTH	6-293
Length of a Geometric Entity Function	2-31

INDEX

Length of a Three Dimensional Curve Function	2-32
LETTER	2-1
Limit Plane	3-41
LINE	2-10, 2-12, 2-40, 3-1, 3-106, 4-25, 4-49
LINEAR	3-140, 3-141, 3-143, 3-145, 6-305
LNTH Function	2-47
LNTHF	2-31
LOAD	C-26
Load Session	G-1
LOADPP	2-44, 9-4
LOADU	2-2, 2-3, 2-44, 5-7
Local Identifiers	2-14
LOCK	6-44
LOG	2-30
LOG10F	2-30
Logical Expression	7-5, 7-6, 7-8, 7-13
Logical IF Statement	7-5, 7-7, 7-13
LONG	7-5
LOOP	9-8
LOOPND	7-2, 7-14
LOOPS	2-15
LOOPST	7-2, 7-14
LOW	9-15

M

MACRO	2-40, 3-243, 9-8, 9-28, 13-18
Macro Assignment Statements	2-59
Macro Class	7-23
Macro Parameter	2-40
MACROS	2-15
Major Word	2-5, 13-18
MARKER	3-198, 8-4
MARKF	4-4
MATRIX	2-12, 2-40, 3-1, 3-354, 4-20, 4-25, 8-9
Matrix Assignment Statements	2-59
MAX1F	2-25
MAXANG	2-35, 6-299, 9-26
MAXDP	2-35, 6-300, 9-26
Maximum and Minimum Functions	2-25
Maximum Stepover	6-125
MAXIS	8-23, 8-24

INDEX

McDonnell Douglas (Mesh Surface)	C-2
MCV	C-2
MESH SURFACES	C-2
MESH.DAT	C-2
MIDDLE	3-93, 3-164
MIN1F	2-25
Minimum Stepover	6-125
Minor Word	2-5, 13-18
MINUS	2-8, 3-274, 3-345
Minus Sign "-"	2-18
MIRROR	3-369
MODALS	9-26
MODE	6-304, 8-8
Modifiers	1-3
MODIFY	6-28, 6-290
Modify Clause	6-28, 6-30
MODSYS	3-356, 4-19, 9-26
MOTION	9-9, 9-21
MOVE	4-20, 6-23, C-27
MULTAX	6-301
Multiple Check Surfaces	6-66
Multiple Intersection	6-69
Multiplication	2-17

N

Natural Logarithm Function	2-30
Negative Number	2-18
NEGX	2-6, 3-272, 3-347
NEY	2-6, 3-272, 3-347
NEGZ	2-6, 3-272, 3-347
Nested Expressions	2-54
NET Surface	3-320
NINT	2-33
NOMORE	4-16, 4-18, 4-19, 4-45, 4-53
NON-CONTROL COMMAND	9-32
NOPS	6-47, 6-49, 6-59, 6-62
NORMAL	8-9
NORMAL, PS	6-30
Northrop (Mesh Surface)	C-2, C-5
NOWARN	7-17, 9-9, 9-21
NOWRAP	3-93, 3-164

INDEX

NSHAPE	2-12, 2-40
NSURF	3-1, 3-296
Num Function	2-39
NUMPTS	2-34, 6-301, 9-26

O

Obsolete Commands	C-9
OBTAIN	3-383, 3-454, 4-20
OFF	9-7, 9-20, 12-4, 12-7
OFFSET	3-68, 3-71, 3-105, 3-175, 3-279, 3-316, 6-17, 6-21, 6-39, 6-232
OMIT	6-44, 6-58, 6-115, 6-232
ON	2-7, 6-63, 6-119, 6-303
ONCE	6-283, 6-293, 6-299, 6-300, 6-302
OPEN	4-26, 9-30
Optional Data Fields	2-10
Orientation Modifiers	2-6
OUT	2-7, 3-4, 3-19, 3-21, 3-27, 3-318, 3-321, 6-114, 6-232, 6-293

P

PA	9-4
Parameter List	7-1
PARAMS	8-9
PARELM	6-33, 6-36
Parentheses	2-19
PARREL	2-6, 3-113, 3-117, 3-140, 3-158, 3-160, 3-179, 3-181, 3-182, 3-348
PART	6-23
Part Program	9-31
Part Surface	6-60
PARTNO	2-1, 2-44
PAST	2-7, 6-63
PATERN	2-12, 2-40, 3-1, 3-137, 4-25
PAUSE	9-4, 9-9, 9-22
Period ":"	2-18
PERPTO	2-6, 3-115, 3-129, 3-185, 3-187, 3-188, 3-189, 3-265, 3-347, 6-30
PLACE	4-24
PLANE	2-12, 2-40, 3-1, 3-132, 3-172, 4-25, 6-3, 6-5
Plotting	8-2
PLUS	2-8, 3-274, 3-345

INDEX

Plus Sign "+"	2-18
PNTVEC	2-12, 3-1, 3-248, 4-25
POCKET	2-8, 2-40, 6-111, 6-113
PODDEF	3-1, 3-243
PODPTS	3-243, 3-245
POINT	2-12, 2-40, 3-1, 3-195, 4-12, 4-25
POINT-VECTOR	3-248
Point-Vector	2-40, 4-14, 4-49
POKMOD	6-114
Positional Modifiers	2-7
Postprocessor	1-4
POSX	2-6, 3-272, 3-347
POSY	2-6, 3-272, 3-347
POSZ	2-6, 3-272, 3-347
PPRINT	2-1, 7-18
PRINT	4-25, 6-302
Priority	2-19, 2-21
PROFIL	6-168
Programmable Functions	E-1
Programmable Keys	F-1
PROJCT	3-164, 3-240, 3-282
PROMPT	2-44, 7-18, 7-23
PSEUDO	2-35, 4-23, 6-7
Pseudo Cutter	13-13
PSIS	6-60
PSTAN	6-63
Punctuation	2-16
PUT	2-8, 5-8

Q

QUIT	9-4
------------	-----

R

RADIAL	3-93, 3-164
RADIUS	2-9, 6-44
RAMP	6-115
Ramp Distance	6-118
RANDOM	3-140, 3-170

INDEX

Range Modifiers	2-8
Range Of Items	5-3
RE	9-4
READ	2-44, 7-30
Real Number	2-21
REDEF	3-321, 4-26, C-10
REFSYS	3-356, 4-19, 4-25, 4-44, 8-9, 9-26
REMARK	2-2, 2-3, 2-4, 7-31, 9-6, 9-16
REMOVE	2-50, 4-45
RENAME	4-45, 5-5, 5-6
RESERV	2-11, 2-13, 4-47
RESET	9-4, C-11
Restoring Program	7-34
RETAIN	6-44
REVERS	4-49, 6-302, B-25
REVOLV	3-93, 3-164, 3-325
RIGHT	2-7, 3-120, 3-122, 6-36, 6-283
RINDEX Function	2-48
RMILL	6-82, 6-203
ROTATE	4-24
Rotational Modifiers	2-7
ROUGH	6-205, 12-3
Ruled Surface	3-292
RUN	9-10
RUNCMD	9-9, 9-22

S

SAME	6-30, 6-289, C-12
Save Session	G-1
SAVEPP	2-44, 9-11
SAVEU	2-2, 2-3, 2-44, 5-13, 10-16
Scalar	2-40, 2-53
Scalar Assignment Statements	2-55
Scalar Expressions	2-53
SCALE	3-362, 6-292, 8-9
SCHECK	9-10, 9-22
SCRUB (Obsolete)	C-12
Secondary	6-293
SEP	9-20
SEQUNC	6-303
SET	9-12

INDEX

SHADE	6-24, 8-8
SHANK	2-35, 4-23, 6-13, 8-12, 13-22
SHAPE (Obsolete)	C-14
SHARP	6-124
SHORT	7-4
SHOW	9-24
Sign Function	2-26
SIGNF	2-26
SIN	2-22
Size Modifiers	2-7
SK	9-28
SKIP	9-28
Slash "/"	2-17
SMALL	2-7, 3-36, 6-232, 6-303
SMILL	6-220
SMOOTH	6-43
SOLID	2-12, 2-40, 3-380
SOURCE	9-26
SpaceMouse	F-1, F-7
SPLINE	2-9, 3-1, 3-52, 3-76, 3-79, 3-91, 3-93
SQRT	2-29
Square Root Function	2-29
SRFTYP	3-294
SRFVCT	6-51
SSPLIN	3-86, 3-88, 3-91, 3-93, 3-100, 3-102, 3-104, 3-105
START	3-93, 3-164
Statement Continuation	2-2
STATLN	9-10, 9-22, 9-27
STEP	6-84, 6-220, 6-300
Step Distance	6-134
Step Down Distance	6-120
stk File	C-26
STL File	3-410, C-26
Stock	14-2
STOP	7-17, 9-10, 9-22, 9-29
Store	5-9
STPCMD	9-10, 9-22
STRCMP Function	2-49
Subfigure	10-17
Subscript	3-57
Subscripted	2-9, 2-12, 4-25, 4-47, 4-51
SUBSTR	7-21
Sub-String Clause	2-45

INDEX

SURF 2-12, 2-40, 3-1, 4-25, 6-3, 6-5
Surface 3-73, 3-289, 4-27, 4-31, 10-7, 11-7
Surface: Bow-Tie 9-10, 9-22
SYMBOL 2-12, 2-40, 4-24, 4-51
Symbol 1-4, 1-6, 10-17, 13-19
Symbol Library 6-8, 6-15, 6-19
SYN 7-32
Synonyms B-27

T

TAN 2-22
TANTO 2-7, 3-15, 3-31, 3-36, 3-120, 3-122, 3-129, 3-270, 3-350, 6-63
TDIST 2-28
TDISTF 2-28
TE 3-202, 3-256
TERMAC 7-32
Text Element 2-44
Text Evaluation 2-51
Text Functions 2-47
Text String 2-43, 2-53, 2-55
Text String Assignment Statements 2-59
Text Variable 2-43
TEXTF 2-50
TH 6-305
THEN 7-17
THICK 2-35, 6-304, 9-27
THRU 1-2, 2-8, 3-170, 3-319, 4-1, 5-3, 5-9
THRU, Point 6-37
TI 9-30
TIME 9-30
TIMES 3-278, 3-341
TITLES 2-2, 2-3, 7-33
TLAXIS 3-256, 3-339, 6-28, 9-28
TLLFT 6-62, 6-81
TLOFPS 6-48, 6-49, 6-82
TLON 6-62, 6-81
TLONPS 6-48, 6-49, 6-82
TLRGRT 6-62, 6-83
TN 6-81
TO 2-7, 6-63
TOLER 2-35, 3-308, 6-305, 9-13, 9-26, 9-27

INDEX

TOOL	8-9, 9-27, 13-32
Tool Axis	9-27
Tool Axis Control Surface	6-30, 6-32
Tool Location	9-27
Tool Position	6-124
TOOLIB	13-1
TOOLPN	C-28
TRACUT	3-356, 4-19, 6-308, 8-9
TRALST	6-309
Transfer of Control	7-5, 7-12
TRANSL	3-359, 6-290
Trigonometric Functions	2-22
TRIMMED	9-23
Trimmed Surface	3-88, 3-318
True or False	7-5
TYPE	2-40
Type Function	2-40
TYPEF	2-40

U

U and V Values	6-73
UBFN	2-2, 2-3, 2-44, 5-1, 10-15, 11-15
UNDO	7-33
Unibase	1-5, 5-1, 10-15, 10-16, 11-15, C-1
UNIT	3-280, 3-340
UNITS	2-35, 6-306, 6-310, 9-28
Untrim	3-318

V

VECTOR	2-12, 3-1, 3-330, 4-25, 4-49
Vector	2-40
VER	9-23, 9-30
VIEW	8-9
VISIBL	8-1, 8-25
VMPMOD	6-260
VOCABF	2-42
Vocabulary Function	2-42
Vocabulary Word	1-2, 2-4, 2-40, B-1

INDEX

VoluMill 6-260

W

WARN 7-17
Warning 9-7, 9-21, 9-24
Waterline Roughing 6-230
WAXIS 8-23, 8-24
WINDOW 9-30
Windows OS 1-7
WIRE 8-8
WLIMIT 9-24
WRAP 3-93, 3-164

X

XAXIS 2-9, 3-124, 3-126, 3-133, 12-4, 12-7
XLARGE 1-3, 2-6, 3-15, 3-17, 3-19, 3-21, 3-23, 3-182, 3-223, 3-305
XSMALL 1-3, 2-6, 3-15, 3-17, 3-19, 3-21, 3-23, 3-182, 3-223, 3-305
XTYPE 2-41
XYROT 2-7, 3-360, 6-291

Y

YAXIS 2-9, 3-124, 3-126, 3-133, 12-4, 12-7
YLARGE 1-3, 2-6, 3-15, 3-17, 3-19, 3-21, 3-23, 3-182, 3-223, 3-305
YSMALL 1-3, 2-6, 3-15, 3-17, 3-19, 3-21, 3-23, 3-182, 3-223, 3-305
YZROT 2-7, 3-360, 6-291

Z

ZLARGE 1-3, 2-6, 3-182, 3-223, 3-305
Z-level 6-119
ZSMALL 1-3, 2-6, 3-182, 3-223, 3-305
ZSURF 3-253, 4-52
ZXROT 2-7, 3-360, 6-291