SYSTEM DOCUMENTATION

FOR

SSV3 INTERFERENCE CHECKING MODULE

PS-90-SS-01

VOLUME II

6/90

CAM-I, Inc.
1250 E. Copeland Road, Suite 500
Arlington, Texas 76011 USA
Tel: (817) 860-1654
Fax: (817) 275-6450

System Documentation

for

SSV3 Interference Checking Module

November, 1989

D.A.C.A.M
Cranfield Institute of Technology

Contents

Notation

The following notation is used in the document.

$\underline{A}$	Vector quantity
$\underline{A} \wedge \underline{B}$	Cross product of $\underline{A}$ and $\underline{B}$
$\underline{A} . \underline{B}$	Dot product of $\underline{A}$ and $\underline{B}$
D.P.	Double Precision
C*20	Character string of 20 characters
[item]	Optional item
$\begin{pmatrix} \text{item1} \\ \text{item2} \end{pmatrix}$	Alternative items

1 Introduction

The SSV3 version of the APT4 + Sculptured Surfaces System has a new facility for checking for possible interference of ball-ended, corner radius and flat-ended cutters with a parametric surface during regional milling and taking appropriate action to avoid gouging. In addition an APT3 type GOUGCK facility similar to that provided in the Arelem has been added to regional milling. These facilities have currently only been implemented in the guide surface type regional milling module REGMILL and not with the isoparametric milling module GOLOFT.

Also the full APT seven segment cutter has been implemented in regional milling and additional tool axis control provided with SCON/AXIS.

The interference checking process is a two stage algorithm that is performed each time a new cutter position is computed. The first step is to identify all patches of the surface where there is a possibilty of penetration occuring, this is a non-iterative process; the second step is to check for actual interference with each of these candidate patches, perforce this is an iterative procedure. Every effort has been made to speed up the iteration, for example by taking the results of the iterative process at the previous cutter location as start values for the current position, but the process is inevitably computationally intensive.

The APT3 type GOUGCK facility that has been added to regional milling, causes a check to be made on each cut vector computed to ensure that gouging with the point of contact does not occur. This may be due either to a very rapid change in the surface curvature and thus the computed stepout although OK for curvature at the initial point is too large, or to a significant change in the point of contact on the tool caused by a change of sign of the cross product of the surface normal and the tool axis. In either case, a gouge is avoided by computing additional cut vectors.

Inteference checking is invoked by programming CHKSRF/PS,ON prior to the regional milling command SMIL. This causes a flag to be set which can be cancelled by CHKSRF/OFF. Similarly gouge checking is invoked by programming GOUGCK/ON and cancelled by GOUGCK/OFF (i.e. in exactly the same manner as for 'APT3' gouge checking in the Arelem). The avoidance control strategies to be used in the event of interference being detected are set using the AVCTRL command, which should also be programmed before SMIL. Full details of the usage of these new commands, the full 7 segment cutter in regional milling and the new tool axis control provided with SCON/AXIS can be found in the Part Programmers Manual, Issue 3, Volume 2, Chapter 15.

2 Interference Checking Algorithms

The fact that for large multi-patch surfaces most of the surface will be remote from the tool at any instance has been taken into account during the development of the interference checking algorithms. The result is a two stage procedure. The first stage is to determine those patches that are sufficiently remote from the tool for there to be no possibility of interference with the tool and those where there is a possibility of the tool penetrating the surface. The second stage is to determine if there is any actual interference with these suspect patches.

In order to facilitate the preliminary scanning of the surface to determine suspect patches, it was decided that some method of creating a set of simple analytic surfaces 'boxing' each patch was required. The technique eventually used is to compute a set of six spherical "shells" for each patch as follows,

a) Sphere of minimum radius that fully encloses the patch
b) Upto 4 "edge" spheres
c) A "lid" sphere

The last five spheres are calculated so as to be as close a fit as possible to the actual surface.

Although the computational effort required to calculate these enclosing "shells" is significant, it need only be made once during the machining of a particular surface. The subsequent checking against these "shells" to determine suspect patches is a simple series of tests. The routine INEX performs the preliminary computation of these spherical "shells" and is only called by the routine SMIL when interference checking is active and a new part surface is being machined. Thus if there is a second call to SMIL following the invocation of interference checking INEX will only be called again if a new part surface is to be machined and the "shell" data for the previous surface will then be overwritten.

INEX also calculates and saves the maximum and minimum curvatures of each patch and the limits for stepout in u and v for each patch that will be used to test for convergence in any subsequent iterative procedures involved in determining the presence or absence of penetration by the tool into the surface.

After interference checking has been invoked in the part program, every time a new tool end location and tool axis orientation is computed the interference checking algorithms are performed and when appropriate avoidance action taken and the modified tool location and axis again checked to ensure no new interference has been introduced.

The procedures followed for each tool location and axis are:

1) Scan all the surface patches except the current patch, (i.e. the patch that the tool is in contact with at its cutting point), to determine any suspect patches where there could possibly be penetration.

2) Check all suspect remote patches for actual interference.

3) Check the current patch for actual interference, other than at the cutting point.

The preliminary scanning of the remote surface patches is carried out by routine REMSCN, which first calculates three spherical "shells" for the tool at its current location. These are:

1) The tool contact sphere which is determined by the tool geometry and the angle between the tool axis and the surface normal at the point of contact. (See subroutine description 6.1.34 TCSPHR for details).
2) The large shallow sphere tangential to the tool tip and containing the tool.
3) The minimum radius sphere enveloping the tool.

The non-current patches are then scanned for possible interference using the following schedule.

1) Test for interference of infinite tool cylinder with minimum patch sphere.
2) If there is still possible interference with this patch, test for interference of tight tool sphere with all patch convex hull spheres.
3) If there is still possible interference with this patch, test for interference of shallow tool sphere with all patch convex hull spheres.
4) If there is still possible interference with this patch, test for interference of tool contact sphere with all patch convex hull spheres.
5) If there is still possible interference with this patch, HULCUT is called to test for interference of actual tool with all patch convex hull spheres unless cutter is ball-ended.

If the patch is not cleared by a negative result in any of the above tests then it is added to the list of suspect patches.

The next step in the interference checking procedure is to test these suspect patches for actual penetration. The routine REMINT does this by first calculating the 'tool center' for simple ball-ended and corner radius cutters as:

$$\underline{TOOLCE} = \underline{TE} + f*\underline{TA}$$

Then for each suspect patch, the iterative procedures to determine the penetration of the tool into or the minimum distance of the tool from the patch are performed. If interference was detected with the patch at the previous tool position, the U and V parameters of that penetration point are taken as the iteration start values for U and V, otherwise U=0.5 and V=0.5 are used.

The routine MIXAL performs the iterative search for interference of a

tool with a patch using the following algorithms.

Given initial estimates of U, V on patch NPATCH, the surface point R, together with its derivatives and normal are evaluated. Then the penetration of point R into the cutter, PENMAX (mimimum distance if no penetration) is computed, together with the associated point P on the tool, and the tool normal PN at P. Next the point T on tool with same normal as the surface at R is calculated. If T is on the flat base of the cutter then algorithm 2 is forced.

MIXAL then proceeds to determine values of DU,DV to give a better estimate of the penetration point, this is done by using values computed by two algorithms,

Algorithm 1 states:

> Given a surface point R and surface normal RN and a point on the tool T with the same normal, compute the point B where this tool normal intersects the tangent plane at R, then by induced parameterization on the tangent plane determine the step out in DU and DV on the surface.

Algorithm 2 states:

> Given an initial surface point, and the tool normal TN at the 'nearest' point on the tool, using a local quadratic approximation to the surface determine DU,DV of point S with same normal TN.

Whenever possible the results of the both algorithms are used, otherwise the results of either algorithm 1 or algorithm 2 only are used as appropriate. The following formula used to estimate DU and DV based on the results of the two algorithms is:

$$DU = \frac{DU1*DU2}{DU1+DU2} \qquad DV = \frac{DV1*DV2}{DV1+DV2}$$

where DU1 & DV1 result from algorithm 1 and DU2 & DV2 result from algorithm 2.

The values of DU and DV are limited to keep U and V within the current patch.

The resultant values of DU and DV are compared with the limiting values DUMIN and DVMIN in order to test for convergence. If no convergence has been achieved after NITLIM iterations, three more iterations are performed and the results interrogated to determine the worst penetration detected.

On convergence the surface point, tool point and normal in addition to the penetration and the U,V parameters of the surface point where the penetration is deepest are returned to the calling routine. If there is no convergence then the worst case results are returned. IFLAG set to -1

if penetration has been detected without convergence otherwise it is set zero. Also if penetration has been detected without convergence a warning message is output. A debug print of the results of the last four iterations can be forced by setting IBUG=11 (MAXDP/-4,11).

If interference is detected the details are stored in the interference table.

The third and final stage of the interference check schedule is to check for interference with the current patch. The following test schedule for current patch is performed by routine CURINT:

1) Test for the patch being convex or 'pseudo' convex by comparing the maximum curvature for the patch with a curvature tolerance that is dependent on the current intol value and the tool geometry. That is treat shallow concave patches as if they were convex provided that the cutter cannot interfere with the patch. If the patch is or 'pseudo' convex then no local interference should occur.

2) Calculate the curvature CT of the minimum spherical convex hull for the tool with the cutting point as a diametral point of contact. Test against the maximum curvature of the patch, CP (Minimum radius of curvature).
If CP <= CT no local interference can occur.

3) Otherwise the same algorithms are used as for the remote patches to search for interference on the patch, using a number of starting points arranged in a pattern about the cutting point. (The cutting point cannot be used as a starting point for these searches, since it is a solution). Five start points are actually used. One in the forward direction, one to each side, and two in the rearward direction with respect to the point of contact, at distances approximately, half the cutter radius (D/4) and in the case of the second rearward point 3D/4.

All non-coincident penetration points are stored in the interference table. That is, if the algorithms converge to the same point as a result of different start points only one solution will be saved.

Finally, if any interference is detected, the largest penetration is determined and its location in the interference table recorded in order that the avoidance procedures can access the relevant data.

3 Avoidance Procedures

There are two possible avoidance strategies either to tilt the tool axis or lift off either normal to the surface or in the direction of the tool axis.

If avoidance strategy type 1 has been specified then an attempt is made to avoid interference by tilting the tool in the most appropriate direction either rolling about or pitching towards or away from the forward direction that is perpendicular to TA. This is done by determining the most appropriate direction, then calculating the tilt angle required to just avoid the surface at the worst point of interference. The direction is indicated by the local axis UL perpendicular to TA that defines the plane in which the tilt is to occur, the sign of the angle being selected accordingly. The constant technological angle specified by the user, needed to clear the surface is then added to this angle, and the resultant angle checked to ensure that it is within the limits indicated by the user, if not it is constrained to be within the limits.

After saving the current toolend coordinates and toolaxis vector, the new tool axis vector, TA that would result from tilting the tool through the resultant angle,B in the direction UL is computed.

$$\underline{TA} = \cos(B)\ \underline{TA} + \sin(B)\ \underline{UL}$$

and the resultant new toolend coordinates TE are computed. Then the centre and radius of the tool contact sphere are computed determined.

A series of checks are then made in order to confirm that interference has been avoided.

First a check is made for possible local interference of the tool with the surface at the point of contact as a result of an increase in the effective radius of the tool caused by tilting. If interference is detected then the maximum angle of tilt so that the tool will fit the surface at the point of contact is computed. This may or may not be possible. If a suitable angle is computed it is again constrained to the permitted limits and yet again a new TA and TE are computed, otherwise the original TE and TA are restored and the tool will be lifted off as the only means of avoiding interference. Suitable messages are output to indicate what action has been taken if PRINT/AVMSG,ON has been programmed.

Finally, the new tool position and orientation (TE and TA) are checked for possible interference by calling the interference checking procedure again, using the original interference data as start points, if avoidance has been successful the original interference data is restored and if requested a suitable message output detailing the avoidance action taken.

If tilting has not sucessfully avoided interference or lift off only was selected, then an attempt is made to avoid the unwanted penetration by lifting off in the specified direction.

$$d = \frac{D}{\underline{SN}.\underline{VL}}$$

where D is maximum penetration at INTLOC
$\underline{SN}$ is unit surface normal at INTLOC
$\underline{VL}$ is unit vector in direction of lift off

If the surface is concave at the point of contact then an additional compensatory liftoff is computed,

$$e = \sqrt{(RCMIN^2 - D^2 + d^2)} - RCMIN$$

where RCMIN is minimum radius of curvature of the surface.

The liftoff is finally increased by the outtol value and checked against the maximum permitted. If outside the limit an appropriate message will be output. Otherwise the new $\underline{TE}$ will be computed ($\underline{TA}$ unchanged) and if liftoff normal to the part surface has been selected the interference checking procedure is again called to confirm that no new interference has been generated as a result of taking avoiding action.

4 Gouge Checking

The gouge checking facility that has been added to regional milling, causes a check to be made on each cut vector computed to ensure that gouging with the point of contact does not occur. This may be due either to a very rapid change in the surface curvature and thus the computed stepout although OK for curvature at the initial point is too large, or to a significant change in the point of contact on the tool caused by a change of sign of the cross product of the surface normal and the tool axis. A gouge is avoided in either case by computing additional cut vectors.

In order to carry out the gouge checking the coordinates and derivatives of the parametric midpoint on the surface between the current and previous surface points at which the tool contacted the surface are computed. Then the surface curvature at this midpoint in chord direction is computed plus the tool contact point at the current and last tool locations and tool normal at midpoint. Note the tool contact point and surface contact point differ by an offset distance equal to the tolerance compensation value. This is because the cut vector is intended to be tangential to the surface between the two endpoints.

The next step is to compute the two tool points, XTP and XTPL that correspond to the tool contact point at the alternative tool location. Then the following deviations from the cut vector are computed:

a) The deviation of the midpoint RM from the chord between the two tool contact points TP and TPL
b) the deviations of the midpoint RM from the chord between the tool contact point and corresponding point for the previous location for both tool locations. (TP to XTPL and TPL to XTP)

The maximum and minimum deviations (DEVMAX and DEVMIN) are then determined and checked to see if the appropriate tolerance band has been violated according to the following rules.

If the surface curvature in the chord direction at the midpoint, SKRM is positive (surface concave) and the absolute value of minimum deviation DEVMIN is greater than the OUTTOL value (TOLO) then excess material will be left on, therefore the step is reduced by a factor equal to the minimum of TOLO/abs(DEVMIN) and 0.95, unless TOLO is less than 1.D-10 (OUTTOL=0) in which case the step is reduced by a factor of 0.75 and a new tool position computed.

If the surface curvature in the chord direction at the midpoint SKRM is negative (surface convex) and the maximum deviation DEVMAX is greater than TOLI then gouging of surface will occur, therefore the step is reduced by a factor equal to the minimum of TOLI/abs(DEVMAX) and 0.95, unless TOLI is less than 1.D-10 (INTOL=0) in which case the step is reduced by a factor of 0.75 and a new tool position computed.

Otherwise no gouge will occur therefore the current tool location is accepted.

Note that whenever gouging is detected the step will be continuously reduced until gouging no longer occurs, the formula for computing the step reduction factor has been chosen so as to minimize the number iterations required to achieve a no gouge situation while maintaining as large cut vectors as possible.

5 Implementation in SSV3

Inteference checking is invoked by programming CHKSRF/PS,ON prior to the regional milling command SMIL. This causes a flag to be set which can be cancelled bu CHKSRF/OFF. Similarly gouge checking is invoked by programming GOUGCK/ON and cancelled by GOUGCK/OFF. The avoidance control strategies are set using the AVCTRL command, which should also be programmed before SMIL. Full details of the usage of these new commands, the 7 segment cutter in regional milling and the new tool axis control provided with SCON/AXIS can be found in the Part Programmers Manual, Issue 3, Volume 2, Chapter 15.

The introduction of new language required modifications to be made to the Load Complex data file and the Translator and for calls to the new execution phase routines to process the new commands to be added to XCALL. Note the AVCTRL command is processed within the routine TSSMIL which was also modified.

As pointed out in the introduction, these new features have only been implemented in the guide surface type regional milling, consequently both facilities are called from within the subroutine SSPATH. However, the preliminary interference checking computations performed by the routine INEX are called from SMIL. The changes needed to implement the APT 7 segment cutter in regional milling were made in routine TULPOS and the changes to add the new tool axis control were made in SCON, DGEOM, AXSET, SSPATH and TULPOS. The actual changes made to AXSET, DGEOM, SCON, SMIL, SSPATH and TULPOS are shown in sections 5.1.1. 5.1.2, 5.1.3, 5.1.4, 5.1.5 and 5.1.6.

The effort involved in invoking interference checking and gouge checking in GOLOFT and/or the Arelem should be straight forward, since they are essentially self contained Modules entered via calls to the routines, INEX, INTCHK, AVOID and GCHK whose arguments are described in the subroutine documentation that follows.

5.1 New Common Blocks and associated Include Files

Five new common blocks have been introduced AVCPRM, INIDAT, INTERF, RMFLGS and TACPRM and are incorporated into routines as include files.

5.1.1 AVCPRM

Include file: AVCPRM.INC

```
C
C       AVOIDANCE CONTROL PARAMETERS
C
      DOUBLE PRECISION CTANG,BMIN,BMAX,DMAX
      LOGICAL LATYP1,LANPS,AVLIST
C
      COMMON/AVCPRM/CTANG,BMIN,BMAX,DMAX,LATYP1,LANPS,AVLIST
C
C     CTANG     CONSTANT TECHNOLOGICAL ANGLE
C     BMIN      MINIMUM PITCH ANGLE
C     BMAX      MAXIMUM PITCH ANGLE
C     DMAX      MAXIMUM LINEAR LIFT OFF
C     LATYP1    FLAG INDICATING IF TYPE1 (TILT) AVOIDANCE REQUIRED
C     LANPS     FLAG INDICATING TYPE OF LINEAR LIFT OFF
C               .TRUE.  - LIFT OFF NORMAL TO PS
C               .FALSE. - LIFT OFF PARALLEL TO TOOL AXIS
C     AVLIST    FLAG INDICATING THAT AVOIDANCE DETAILS ARE TO BE PRINTED
C
```

5.1.2 INIDAT

Include file: INIDAT.INC

```
C
C.... INTERFERENCE CHECKING INITIAL DATA
C
      DOUBLE PRECISION CXHULL,DULIM,DVLIM,CURMIN,CURMAX
      INTEGER NGP,NCXHUL,MXNP,ICURSF
C
      PARAMETER (MXNP=1000)
C
C.... SPHERICAL SURFACE ENCLOSURES (HULLS)
      DIMENSION CXHULL(4,6,MXNP)
C.... NUMBER OF HULLS
      DIMENSION NCXHUL(MXNP)
C.... DU ITERATION LIMIT FOR PATCHES
      DIMENSION DULIM(MXNP)
C.... DV ITERATION LIMIT FOR PATCHES
      DIMENSION DVLIM(MXNP)
C.... MAXIMUM AND MINIMUM CURVATURE OF PATCHES
      DIMENSION CURMAX(MXNP),CURMIN(MXNP)
C
C     MXNP   -  MAXIMUM NUMBER OF PATCHES PER SURFACE
C     NGP    -  NUMBER OF PATCHES IN CURRENT SURFACE
C     ICURSF -  EXTERNAL LDA FILE RECORD NUMBER FOR CURRENT SURFACE
C
      COMMON/INIDAT/CXHULL,DULIM,DVLIM,CURMAX,CURMIN,NCXHUL,NGP,ICURSF
C
```

5.1.3 INTERF

Include file: INTERF.INC

```
C
C     INTERFERENCE TABLE
C
      INTEGER MAXNP
      PARAMETER (MAXNP=1000)
C
      INTEGER PINTR,LPINTR,NPINTR,LNPINT,ISEG
      DOUBLE PRECISION DINTR,UINTR,VINTR,LDINTR,LUINTR,LVINTR
      DOUBLE PRECISION TPINTR,SPINTR,SNINTR
C
      DIMENSION DINTR(MAXNP),PINTR(MAXNP),UINTR(MAXNP),VINTR(MAXNP)
      DIMENSION LDINTR(MAXNP),LPINTR(MAXNP),LUINTR(MAXNP),LVINTR(MAXNP)
      DIMENSION TPINTR(3,MAXNP),SPINTR(3,MAXNP),SNINTR(3,MAXNP)
      DIMENSION ISEG(MAXNP)
C
      COMMON/INTERF/DINTR,UINTR,VINTR,LDINTR,LUINTR,LVINTR,
     +              TPINTR,SPINTR,SNINTR,PINTR,LPINTR,ISEG,NPINTR,LNPINT
C
```

where

DINTR	Penetration distance
UINTR	U parameter of surface point at penetration
VINTR	V parameter of surface point at penetration
LDINTR	Previous penetration distance
LUINTR	U parameter of surface point at previous penetration
LVINTR	V parameter of surface point at previous penetration
TPINTR	Coordinates of point on tool at penetration
SPINTR	Coordinates of point on surface at penetration
SNINTR	Surface normal at penetration
PINTR	Patch number for surface point at penetration
LPINTR	Patch number for surface point at previous penetration
ISEG	Tool segment for penetration
NPINTR	Number of interference points detected
LNPINT	Previous number of interference points detected

5.1.4 RMFLGS

Include file: RMFLGS.INC

```
C
C.... NEW REGIONAL MILLING FLAGS
C
      LOGICAL LGCHK,LINTCK,LAVCTL,LVAXIS
C
      COMMON/RMFLGS/LGCHK,LINTCK,LAVCTL,LVAXIS
C
C     LGCHK  - GOUGE CHECK FLAG
C     LINTCK - INTERFERENCE CHECKING FLAG
C     LAVCTL - AVOIDANCE CONTROL FLAG
C     LVAXIS - VARIABLE TOOL AXIS FLAG
C
```

5.1.5 TACPRM

Include file: TAC.INC

```
C
      DOUBLE PRECISION ALPHA,BETA,COSA,COSB,SINA,SINB
      COMMON/TAC/ALPHA,BETA,COSA,COSB,SINA,SINB,ITAC
C
C     ALPHA  TOOL AXIS SLEW ANGLE WRT SN
C            IN GUIDE SURFACE CRSSPL DIRN. IF PATH IN TANSPL DIRN.
C            AND VICE VERSA
C     BETA   TOOL AXIS TILT ANGLE (CUTANG) WRT SN
C            IN PATH DIRECTION
C     COSA   COS(ALPHA)
C     COSB   COS(BETA)
C     SINA   SIN(ALPHA)
C     SINB   SIN(BETA)
C     ITAC   FLAG INDICATING THAT ATANGL TYPE TOOL AXIS CONTROL
C            IS REQUIRED =1, =0 FOR FIXED OR NORMDS
C
```

5.2 Changes to Regional Milling Routines

The changes made to regional milling routines are shown in the following subsections. Actual changes are highlighted in bold. In order to reduce the amount of text included in this document unchanged sections of code are omitted and indicated by

```
                :
                :
```

5.2.1 AXSET

```
      SUBROUTINE AXSET(MODE,FAX,VAX,AX)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON/IBUGG/IBUG,IPCOLC,IPCOLM
C--PURPOSE IS TO SET AXIS AX TO FAX IF MODE IS 0 OR VAX IF
C--MODE IS NOT ZERO.  IF VAX IS ZERO, AX WILL BE SET TO FAX
C.... IF MODE=2 THEN VAX WILL BE VARIED BY APPLYING ANGLE ALPHA
C                    IN DIRECTION INDICATED BY FAX
      DIMENSION FAX(3),VAX(3),AX(4),FAXN(3)
      CHARACTER*20 BADMSG
      INCLUDE 'TAC.INC'
      DATA ONE,SMAL/1.0D0,1.0D-38/
                :
                :
                :
   50 CONTINUE
C---VARIABLE AXIS CASE
      CALL DOTF(SIZ,VAX,VAX)
      IF(SIZ.LT.SMAL) GO TO 10
      FAC=ONE/DSQRT(SIZ)
C--ADJUST SIGN OF AXIS ACCORDING TO SIGN OF MODE
      IF(MODE.LT.0) FAC=-FAC
      DO 60 I=1,3
   60 AX(I)=VAX(I)*FAC
C...NOW APPLY ALPHA IF REQUIRED
      IF (ABS(MODE).EQ.2) THEN
C... NORMALIZE FAX
        CALL VNORM(FAX,FAXN)
        DO 70 I=1,3
          AX(I)=COSA*AX(I)+SINA*FAXN(I)
   70   CONTINUE
      ENDIF
C
  100 CONTINUE
                :
                :
                :
      END
```

5.2.2 DGEOM

```
      SUBROUTINE DGEOM(DSURF,IDTYPE,U,V,IDAXIS,DAXIS,
     X      ITLDIR,TLDIR,DTOL,DR,U1,V1,NPAT,CX)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON/IBUGG/IBUG,IPCOLC,IPCOLM
C--PURPOSE IS TO GENERATE ALL NECESSARY INFORMATION AT CURRENT
C--POINT U,V OF DRIVE GEOMETRY.  INPUT PARAMETERS ALL EXPLAINED
C--IN PATH
C--OUTPUT
C--DR  ARRAY CONTAINING ALL RELEVANT CURVE INFORMATION AT POINT U,V
                :
                :
                :
 8809 CONTINUE
      DO 10 I=1,16
   10 DR(I)=CX(I)
C--SET AXIS ACCORDING TO IDAXIS SETTING
      IF (ABS(IDAXIS).LE.1) THEN
      CALL AXSET(IDAXIS,DAXIS,CX(17),DR(17))
      ELSE
        CALL AXSET(IDAXIS,CX(13),CX(17),DR(17))
      ENDIF
C--SET THE TOOL DIRECTION NEXT
      CALL AXSET(ITLDIR,TLDIR,CX(17),DR(21))
      GO TO 200
                :
                :
                :
  120 CONTINUE
C--SET THE TOOL AXIS FIRST
      IF (ABS(IDAXIS).LE.1) THEN
      CALL AXSET(IDAXIS,DAXIS,CX(29),DR(17))
      ELSE
        CALL AXSET(IDAXIS,DR(29),CX(29),DR(17))
      ENDIF
C--SET THE TOOL DIRECTION NEXT
      CALL AXSET(ITLDIR,TLDIR,CX(29),DR(21))
C
  200 CONTINUE
                :
                :
                :
      END
```

5.2.3 SCON

```
      SUBROUTINE SCON
                :
                :
                :
C
      COMMON/IBUGG/IBUG,I1,I2
      COMMON/XDS/DGEOM(4),DLIM(4),TLDIR(3),DOFSET,ITLDIR,IDTOOL,NDS
      COMMON/XPS/PGEOM(4),PSIDE,PTHICK,IPTOOL,NPS
      COMMON/XFEED/CLRPLN(4),FED(4),NCLRPL,NFEED
      COMMON/XSTEPO/SSTEP(4),NSTEPO
      COMMON/XAXIS/AXIS(3),IAXIS,NAXIS
      INCLUDE 'TAC.INC'
      INCLUDE 'ZNUMBR.INC'
C
C
C--WORD MODE VALUES IN REAL FORM AND HOLLERITH FORM.
C--THESE ARE WORDS USED IN SMIL AND SCON COMMANDS
C
                :
                :
                :
      DATA ATANSP/136.D0/,WTANSP/'TANSPL'/
      DATA ATO/3.D0/,WTO/'TO    '/
      DATA AVECTO/20.D0/,WVECTO/'VECTOR'/
      DATA AZIGZA/61.D0/,WZIGZA/'ZIGZAG'/
      DATA AATANG/10.D0/,WATANG/'ATANGL'/
      DATA ACUTAN/71.D0/,WCUTAN/'CUTANG'/
      DATA ALEAD/74.D0/,WLEAD/'LEAD  '/
      DATA ALAG/75.D0/,WLAG/'LAG   '/
C
                :
                :
                :
C
  600 CONTINUE
C---AXIS PROCESSING FOLLOWS
      NAXIS=NOGOOD
      ICUR=ICUR+2
      IF(COM(ICUR).EQ.ANORMD) GO TO 650
      IF(COM(ICUR).EQ.AVECTO) GO TO 610
      IF(COM(ICUR).EQ.AATANG) GO TO 680
      GO TO 998
                :
                :
                :
  610 CONTINUE
      IF(CKDEF(COM(ICUR+1))) GO TO 997
      CALL DOTF(VSQ,COM(ICUR+1),COM(ICUR+1))
      IF(VSQ.LT.SMAL) GO TO 996
      VSQ=DSQRT(VSQ)
      DO 620 I=1,3
  620 AXIS(I)=COM(ICUR+I)/VSQ
```

```
      IAXIS=0
      ITAC=0
      ICUR=ICUR+4
      IF(COM(ICUR).NE.APLUS.AND.COM(ICUR).NE.AMINUS) GO TO 690
      ICUR=ICUR+2
      IF(COM(ICUR-2).NE.AMINUS) GO TO 690
C---REVERSE THE VECTOR DIRECTION.
      DO 670 I=1,3
  670 AXIS(I)=-AXIS(I)
      GO TO 690
C
  650 CONTINUE
      ICUR=ICUR+2
      IAXIS=1
      ITAC=0
      DO 660 I=1,3
  660 AXIS(I)=ZERO
      IF(COM(ICUR).NE.APLUS.AND.COM(ICUR).NE.AMINUS) GO TO 690
      IF(COM(ICUR).EQ.AMINUS) IAXIS=-1
      ICUR=ICUR+2
      GO TO 690
C                  PS
C.... AXIS,ATANGL,DS,ALPHA,CUTANG,BETA
  680 ICUR=ICUR+2
      CALL HOLFRM(COM(ICUR+1),CTEST,1,6,NWD)
      IF (CTEST.EQ.WDS) THEN
        IAXIS=2
        ITAC=1
        DO 682 I=1,3
         AXIS(I)=ZERO
  682   CONTINUE
      ELSE IF (CTEST.EQ.WPS) THEN
        IAXIS=0
        ITAC=2
        AXIS(1)=ZERO
        AXIS(2)=ZERO
        AXIS(3)=ONE
      ELSE
        GOTO 998
      ENDIF
      ICUR=ICUR+2
      IF (COM(ICUR).EQ.ASCALA) THEN
        IF (CKDEF(COM(ICUR+1))) THEN
          GOTO 997
        ELSE
          ALPHA=COM(ICUR+1)*DEGRAD
          COSA=COS(ALPHA)
          SINA=SIN(ALPHA)
          IF (COSA.LT.0) THEN
            IAXIS=-IAXIS
          ENDIF
        ENDIF
      ELSE
        GOTO 998
```

```
      ENDIF
      ICUR=ICUR+2
      IF (COM(ICUR).EQ.ACUTAN) THEN
        ICUR=ICUR+2
        IF (COM(ICUR).EQ.ALEAD) THEN
         IFLG=1
         ICUR=ICUR+2
        ELSE IF (COM(ICUR).EQ.ALAG) THEN
         IFLG=-1
         ICUR=ICUR+2
        ELSE
         IFLG=1
        ENDIF
        IF (COM(ICUR).EQ.ASCALA) THEN
          IF (CKDEF(COM(ICUR+1))) THEN
            GOTO 997
          ELSE
            BETA=COM(ICUR+1)*DEGRAD*FLOAT(IFLG)
            COSB=COS(BETA)
            SINB=SIN(BETA)
            ICUR=ICUR+2
          ENDIF
        ELSE
          GOTO 998
        ENDIF
      ELSE
        GOTO 998
      ENDIF
C
  690 CONTINUE
      IF(ICUR.NE.NLAST) GO TO 995
      NAXIS=IOK
      GO TO 999
                :
                :
                :
```

5.2.4 SMIL

```
      SUBROUTINE SMIL
                :
                :
                :

C
      COMMON/IBUGG/IBUG,I1,I2
      COMMON/XDS/DGEOM(4),DLIM(4),TLDIR(3),DOFSET,ITLDIR,IDTOOL,NDS
      COMMON/XPS/PGEOM(4),PSIDE,PTHICK,IPTOOL,NPS
      COMMON/XFEED/CLRPLN(4),FED(4),NCLRPL,NFEED
      COMMON/XSTEPO/SSTEP(4),NSTEPO
      COMMON/XAXIS/AXIS(3),IAXIS,NAXIS
C
      INCLUDE 'RMFLGS.INC'
C
      DIMENSION TUL(7),VEC(3),VINC(4),TE(3),TA(3),TD(3)
      DIMENSION PTEM(2)
C
                :
                :
                :
  170 CONTINUE
C
      DSTEP=ONE
      IPART=PGEOM(4)
      IDRIV=DGEOM(4)
      STEPMX=ZERO
C
C.... PERFORM PRELIMINARY INTERFERENCE CHECKING CALCULATIONS IF REQD
      IF (LINTCK) CALL INEX(COM(IPART),PSIDE,PTOLI,PTOLO)
C
      CALL SSPATH(COM(IDRIV), IDTYPE, PDIR, DTOL, DSTEP, DOFSET,
     X           DLIM, IDTOOL, IAXIS, AXIS, ITLDIR,
     X       TLDIR, COM(IPART), PSIDE, PTHICK, PTOLI, PTOLO, IPTOOL,
     X           TUL, UST, VST, SCALHT, STEPOV, STEPMX,
     X           AMAXDP,ELMAX, MAXCL, ICLSKP, IQUAL,
     X           TE,  TA,  TD,  PTEM,  IRR)
      IF(IRR.NE.0) GO TO 990
C--POSN SUCESSFUL, NOW COMPENSATE TOOL END
                :
                :
                :
  220 CONTINUE
C
      IPART=PGEOM(4)
      DSTEP=ONE
      IDRIV=DGEOM(4)
      STEPMX=ZERO
C
C.... PERFORM PRELIMINARY INTERFERENCE CHECKING CALCULATIONS IF REQD
      IF (LINTCK) CALL INEX(COM(IPART),PSIDE,PTOLI,PTOLO)
C
```

```
      CALL SSPATH(COM(IDRIV), IDTYPE, PDIR, DTOL, DSTEP, DOFSET,
     X           DLIM, IDTOOL, IAXIS, AXIS, ITLDIR,
     X       TLDIR, COM(IPART), PSIDE, PTHICK, PTOLI, PTOLO, IPTOOL,
     X           TUL, UST, VST, SCALHT, STEPOV, STEPMX,
     X           AMAXDP,ELMAX, MAXCL, ICLSKP, IQUAL,
     X           TE,  TA,  TD,  PTEM,  IRR)
      IF(IRR.NE.0) GO TO 990
      GO TO 999
                :
                :
                :
  310 CONTINUE
      IF(ICUR.NE.NLAST) GO TO 995
      IF(NDS.EQ.NOGOOD) GO TO 994
      IF(NPS.EQ.NOGOOD) GO TO 994
      IF(NAXIS.EQ.NOGOOD) GO TO 994
      IF(NFEED.EQ.NOGOOD) GO TO 994
      IF(NSTEPO.EQ.NOGOOD) GO TO 994
      IF(WORD.EQ.APICKF.AND.NCLRPL.EQ.NOGOOD) GO TO 994
C
C--GET OTHER NEEDED APT VARIABLES
      CALL GETAPT(TUL,PTOLI,PTOLO,DTOL,AMAXDP,ELMAX,MAXCL,IR)
      IF(IR.NE.0) GO TO 993
C--FETCH PART AND DRIVE SURFACES
      CALL GFETCH(NBIG,MAXLDA,COM,DGEOM,PGEOM,IR)
      IF(IR.NE.0) GO TO 992
C--SET APPROPRIATE STARTING VALUES FOR CALL TO SSPATH
      IF(UST.LT.DLIM(1)-SMAL.OR.UST.GT.DLIM(2)+SMAL) GO TO 996
      IF(VST.LT.DLIM(3)-SMAL.OR.VST.GT.DLIM(4)+SMAL) GO TO 996
C--CUTREG ROUTINE SEPARATED TO REDUCE SUBROUTINE SIZE
      IQUAL=0
C
C.... PERFORM PRELIMINARY INTERFERENCE CHECKING CALCULATIONS IF REQD
      IPART=PGEOM(4)
      IF (LINTCK) CALL INEX(COM(IPART),PSIDE,PTOLI,PTOLO)
C
      CALL CUTREG(WORD,CDIR,UST,VST,PDIR,IDTYPE,IQUAL,
     X   TUL,PTOLI,PTOLO,DTOL,AMAXDP,ELMAX,MAXCL,ISKP,IRR)
      IF(IRR.NE.0) GO TO 990
      GO TO 999
C
                :
                :
                :
```

5.2.5 SSPATH

```
      SUBROUTINE SSPATH(......
                :
                :
                :
C--OUTPUT
C--TE,TA,TD LAST GOOD TOOL END,AXIS AND DIRECTION OF PROJECTION
C--PAR  LAST PARAMETRIC VALUES ON THE DRIVE GEOMETRY
C--IRR  ERROR FLAG. NON ZERO ON RETURN FROM SSPATH IF ERROR OCCURS
C
      DIMENSION DSURF(*),PSURF(*),TUL(*),DAXIS(*),TLDIR(*)
      DIMENSION PLIM(4)
      DIMENSION DR(36),R(32),RL(32),RX(32),RXL(32)
      DIMENSION RVEC(3),PAR(2),FWD(3)
      DIMENSION RDIF(3),TE(3),TEL(3),TAL(3),TA(3),TEX(3),TAX(3)
      DIMENSION TC(3),TD(3),TBEF(3),TBEFL(3),TBEFX(3)
      CHARACTER*20 BADMSG
C
C---      AKMXMN THE LARGEST POSITIVE AND NEGATIVE CURVATURES
C---      IN THIS CALL TO SSPATH
C---      AKDRV(2,2) THE DRIVE SURFACE PARAMETERS WHERE THEY OCCURRED
C---      AKPRT(2,2) THE PART SURFACE PARAMETERS WHERE THEY OCCURRED
      DIMENSION AKMXMN(2),AKDRV(2,2),AKPRT(3,2)
C
      INCLUDE 'DARRAY.INC'
      INCLUDE 'RMFLGS.INC'
      INCLUDE 'TAC.INC'
C.... LRET - INTERFERENCE FLAG = .TRUE. IF TOOL PENETRATES PS
C.... LARET - AVOIDANCE FLAG = .TRUE. IF INTEFERENCE AVOIDED
      LOGICAL LRET,LARET
C.... INTLOC - LOCATION IN INTERFERENCE TABLE OF WORST PENETRATION
      INTEGER INTLOC
C
      DATA ZERO,ONE,SMAL/0.0D0,1.0D0,1.0D-38/
      DATA BIG/1.0D+10/
      DATA HALF/0.5D0/,ZEM5/0.00001D0/
      DATA ISEQLS,KPATH/0,0/
                :
                :
                :
C---      SET FLAG FOR MODE TO SKIP SURFACE CALCULATION
      ISKBAL=0
      IF(PSURF(1).NE.DSURF(1)) GO TO 60
      IF(DABS(DOFSET).GT.SMAL) GO TO 60
      IF(IDTOOL.NE.1) GO TO 60
      ISKBAL=1
   60 CONTINUE
C
C.... SIZE OF PART SURFACE CANONICAL FORM
      IPSIZE=NINT(PSURF(8))
C
C--SURFACE CONTACT BLOCK.  DROP SURFACE FROM POINT ON DRIVE GEOMETRY
C--TO PART SURFACE
```

```
  100 CONTINUE
                :
                :
                :
  200 CONTINUE
C--FOR FIRST POINT OR FIXED PARAMETER STEP, SKIP THIS
      IF(IPASS.EQ.1.OR.DSTEP.LT.ZERO) GO TO 400
C
      CALL TULPOS(R,PSIDE,DR,TUL,PTHICK,ZERO,IPTOOL,IDTOOL,
     X            TEX,TAX,TBEFX,PDIR)
C--NEW CANDIDATE TOOL END TOOL AXIS IS TEX,TAX
:
:
:
  230 CONTINUE
      STEPL=STEP
      STEP=STEP*FAC
C--CUT DOWN STEPOUT AND TRY AGAIN
  240 IFIN=0
      ISTEPK=ISTEPK+1
C****    ERROR
      IRR=2
      IF(ISTEPK.GT.MXSTEP) GO TO 999
      GO TO 610
:
:
:
C--FIND NEW TE,TA
      CALL TULPOS(R,PSIDE,DR,TUL,PTHICK,TOLCOM,IPTOOL,IDTOOL,
     X            TE,TA,TBEF,PDIR)
C--CARRYOUT GOUGE CHECKING IF REQUIRED
      IF ( LGCHK.AND.(IPASS.NE.1).AND.(IQUAL.NE.1)) THEN
        CALL GCHK(PSURF,PSIDE,R,TE,TA,NPAT,UP,VP,
     +            RL,TEL,TAL,NPATL,UPL,VPL,TOLCOM,PTOLI,PTOLO,GFAC,IRET)
        IF (IRET.NE.0) THEN
C--EITHER GOUGING OCCURS OR MATERIAL IS LEFT ON
C  REDUCE STEPOUT UNLESS FIXED PARAMETRIC STEP
          IF (DSTEP.LT.ZERO) THEN
            IF (IRET.GT.0) THEN
             CALL CFORM(' *** WARNING - GOUGING OCCURS ***',DARRAY,1,33)
            ELSE
             CALL CFORM(' *** WARNING - EXCESS MATERIAL LEFT ON ***',
     +                  DARRAY,1,42)
            ENDIF
            CALL CPRINT(DARRAY)
            CALL BAD(3,1,'TE  ',TE)
            CALL BAD(-3,1,'TEL ',TEL)
          ELSE
            STEPL=STEP
            STEP=STEP*GFAC
C.... RESTORE PREVIOUS VALUES OF TE,TA AND TBEF
            DO 415 I=1,3
              TE(I)=TEL(I)
              TA(I)=TAL(I)
```

```
            TBEF(I)=TBEFL(I)
  415       CONTINUE
            GOTO 240
          ENDIF
        ENDIF
      ENDIF
C
C.... CARRY OUT INTERFERENCE CHECKING IF REQUIRED
C
      IF (LINTCK) THEN
        CALL INTCHK(PSURF,IPSIZE,PSIDE,PTHICK,PTOLI,TUL,TE,TA,
     +              R,NPAT,UP,VP,LRET,INTLOC)
        IF (LRET) THEN
C.... INTEFERENCE HAS BEEN DETECTED - PRINT DETAILS OF INTEFERENCE
          CALL INTMSG(TE,TA,INTLOC)
          IF (LAVCTL) THEN
C.... AVOIDANCE CONTROL REQUESTED
C     COMPUTE GUIDE SURFACE FORWARD DIRECTION
            CALL VSCALE(PDIR,DR(13),FWD)
            CALL AVOID(PSURF,IPSIZE,PSIDE,PTHICK,PTOLI,PTOLO,TOLCOM,FWD,
     +                 TUL,TE,TA,R,NPAT,UP,VP,INTLOC,LARET)
            IF (LARET) THEN
              CALL OUTMSG('INTERFERENCE AVOIDED')
            ELSE
              CALL ERROR(-3555,'SSPATH  ')
            ENDIF
          ENDIF
        ENDIF
C.... RESET INTEFERENCE TABLE
        CALL RESET

      ENDIF
C--COMPUTE TOOL END DISTANCE
      TBEMAX=ZERO
      TEMOV=ZERO
      IF(IPASS.EQ.1) GO TO 430
                :
                :
                :
  440 CONTINUE
      RKL=RK
C--SAVE PATCH NO, U,V OF THIS SURFACE POINT
      NPATL=NPAT
      UPL=UP
      VPL=VP
      IF(IBUG.NE.11) GO TO 8806
                :
                :
                :
 8806 CONTINUE
C.... IF AXIS/ATANGL,PS,.. TOOL AXIS CONTROL - SAVE CURRENT TA
C     AS INITIAL TA FOR NEXT CUTTER LOCATION
      IF (ITAC.EQ.2) THEN
        DO 445 I=1,3
```

```
          DAXIS(I)=TA(I)
  445   CONTINUE
        ENDIF
C--RESET COUNTERS AND CHECK PATH LIMITS
        ISTEPK=0
        ICLCT=ICLCT+1
C****     ERROR
        IRR=3
        IF(ICLCT.GT.MAXCL) GO TO 999
                  :
                  :
                  :
```

5.2.6 TULPOS

```
      SUBROUTINE TULPOS(R,SIDE,DR,TUL,PTHICK,TOLCOM,IPTOOL,IDTOOL,
     X                  TE, TA,TBEF,PDIR)
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
C
C-- R     INPUT SURFACE CONTACT POINT/DERIVATIVES(SEE CNSURF)
C-- SIDE  INPUT 1 OR -1, DEFINES CUTTER SIDE OF SURFACE
C-- DR    DRIVE ELEMENT POINT/DERIVATIVES(SEE DGEOM)
C-- TUL   CUTTER PARAMETERS
C-- PTHICK PART GEOMETRY THICK SPECIFICATION
C-- TOLCOM TOLERANCE BAND COMPENSATION
C-- IPTOOL PART SURFACE CONTROL FLAG( 0 FOR ON, 1 FOR TO)
C-- IDTOOL DRIVE ELEMENT CONTROL FLAG
C-- TE   OUTPUT TOOL END
C-- TA   OUTPUT TOOL AXIS
C-- TBEF OUTPUT EFFECTIVE TOOL END BASED ON SPHERICAL CUTTER
C-- PDIR DIRECTION OF MOTION WRT INCREASING PARAMETER (+1 OR -1)
      COMMON/IBUGG/IBUG,IPCOLC,IPCOLM
      DIMENSION VEC(3)
C--PURPOSE IS TO COMPUTE TOOL END AND TOOL AXIS WHEN AN A
C--ACCEPTABLE SURFACE CONTACT POINT HAS BEEN FOUND
      DIMENSION R(32),DR(36),TUL(4),TE(3),TA(3),TBEF(3)
      CHARACTER*20 BADMSG
      CHARACTER*120 DARRAY
      DIMENSION VECL(3),V(3),VN(3)
      DOUBLE PRECISION NPS
      DIMENSION FWD(3),NPS(3),ROLL(3),PITCH(3)
C--TOOL CANONICAL DATA
      INCLUDE 'TLLDAT.INC'
C--TOOL AXIS CONTROL DATA
      INCLUDE 'TAC.INC'
C--INITIALZE VECL AND FIRST CALL FLAG
      LOGICAL LFIRST
      DATA VECL/3*0.D0/,LFIRST/.TRUE./
C--SAVE LAST DIRECTION FROM SP TOWARDS TOOL AXIS,AND FLAG LFIRST
      SAVE VECL,LFIRST
C--FOR FIRST CALL (LFIRST=.TRUE.) SET VECL TO REV. MOTION DIRECTION
      IF (LFIRST) THEN
        DO 5 I=1,3
          VECL(I)=-1.0*PDIR*DR(12+I)
   5    CONTINUE
        LFIRST=.FALSE.
      ENDIF
C.... IF TOOL AXIS CONTROL WRT PS - COMPUTE LOCAL PITCH AND ROLL AXES
      IF (ITAC.EQ.2) THEN
C.... GUIDE SURFACE FORWARD DIRECTION - PDIR*(DRIVE TANGENT VECTOR)
        CALL VSCALE(PDIR,DR(13),FWD)
C.... CUTTER SIDE OF PART SURFACE
        CALL VSCALE(SIDE,R(29),NPS)
C.... COMPUTE UNIT VECTOR PERPENDICULAR TO GUIDE SURFACE FORWARD AND
C     PROJECTION VECTORS
        CALL CROSSV(FWD,DR(21),V)
C.... COMPUTE LOCAL FORWARD MOTION DIRECTION IN PS TANGENT PLANE
```

```
C            (ROLL AXIS)
        CALL CROSSV(NPS,V,ROLL)
C.... COMPUTE LOCAL AXIS ORTHOGONAL WITH ROLL AXIS AND PS NORMAL
C            (PITCH AXIS)
        CALL CROSSV(NPS,ROLL,PITCH)
C.... DETERMINE DIRECTION IN WHICH ROLL ANGLE IS TO BE APPLIED
C     POSITIVE IN STEPOVER DIRECTION
        CALL DOTV(SGN,PITCH,DR(29))
        SGN=SIGN(1.D0,SGN)
       ENDIF
C-- TWO MAIN CASES, TOOL ON PS OR NOT ON.
       IF(IPTOOL.NE.0) GO TO 50
C--GENERAL CASE, TOOL OFFSET FROM SURFACE
       OFFSET=SIDE*(PTHICK+TOLCOM+TUL(1))
       DO 10 I=1,3
        IF (ITAC.EQ.1) THEN
C.... APPLY CUTANG BETA TO TA IN DIRECTION OF MOTION
         TA(I)=COSB*DR(16+I) + PDIR*SINB*DR(12+I)
        ELSE IF (ITAC.EQ.2) THEN
C.... APPLY BETA WRT TO PS NORMAL IN MOTION DIRECTION
C     AND ALPHA IN PLANE PERPENDICULAR TO MOTION DIRECTION
         TA(I)=COSA*COSB*NPS(I) + SINB*ROLL(I)
     +        + SGN*SINA*COSB*PITCH(I)
        ELSE
         TA(I)=DR(16+I)
        ENDIF
        TBEF(I)=R(I)+OFFSET*R(I+28)-TUL(1)*TA(I)
   10 CONTINUE
C--GENERAL CUTTER CALCULATION.
       CALL CROSS(TA,R(29),VEC)
C...IF TA IS PARALLEL TO SN USE PREVIOUS VALUE OF VEC INSTEAD OF SN
       VLEN2=VEC(1)*VEC(1)+VEC(2)*VEC(2)+VEC(3)*VEC(3)
       IF (VLEN2.LT.1.0D-12) THEN
         CALL CROSS(TA,VECL,VEC)
       ENDIF
       DO 20 I=1,3
   20 VEC(I)=SIDE*VEC(I)
       CALL CROSS(VEC,TA,VEC)
       CALL VNORM(VEC,VEC)
C...MODIFY OFFSET FOR 7 SEGMENT CUTTER CALCULATION
       OFFSET=OFFSET-SIDE*TUL(1)
         DO 32 I=1,3
          VECL(I)=VEC(I)
   32    CONTINUE
C...COMPUTE TANGENT OF ANGLE BETWEEN TOOL NORMAL AND TOOL RADIAL AXIS
       CALL DOTV(CANG,VEC,R(29))
       CALL CROSS(VEC,R(29),V)
       CALL VNORM(V,VN)
       CALL DOTV(SANG,V,VN)
       IF (ABS(CANG).GT.1.D-15) THEN
         TANG=SANG/CANG
       ELSE
         TANG=SIGN(1.D0,SANG)*1.D15
       ENDIF
```

```
C...IS ANGLE VALID FOR SEGMENTS 3,4,5 OF CUTTER
C   I.E. CORNER RADIUS AND ADJACENT POINT CIRCLE SEGMENTS
      DO 35 I=3,5
        IF ((TCONT(I).GT.0).AND.
     1      ((TANG-TANLO(I))*(TANG-TANHI(I)).LE.0.D0)) THEN
C...COMPUTE TOOL END COORDS
          DO 34 K=1,3
            TE(K)=R(K)+(OFFSET+SIDE*CORRAD(I))*R(28+K)
     1           +RI(I)*VEC(K)-TLHITE(I)*TA(K)
   34     CONTINUE
          GOTO 999
        ENDIF
   35 CONTINUE
C...NO - IS ANGLE VALID FOR TOOL TIP
      IF ( (TCONT(1).GT.0).AND.
     1   ( (TANG-TANLO(1))*(TANG-TANHI(1)).LE.0.D0) ) THEN
C...YES - OUTPUT WARNING
        CALL CFORM(' *** WARNING - CUTTING ON TOOL TIP ***',DARRAY,1,38)
        CALL CPRINT(DARRAY)
        DO 40 K=1,3
          TE(K)=R(K)+OFFSET*R(28+K)
   40   CONTINUE
C...NO - IS ANGLE VALID FOR TOP OF UPPER LINE SEGMENT
      ELSE IF ( (TCONT(7).GT.0).AND.
     1         ( (TANG-TANLO(7))*(TANG-TANHI(7)).LE.0.D0) ) THEN
C...YES - OUTPUT WARNING
        CALL CFORM(' *** WARNING - CUTTING ON POINT CIRCLE AT TOP OF UPP
     1ER LINE SEGMENT ***',DARRAY,1,71)
        CALL CPRINT(DARRAY)
        DO 42 K=1,3
          TE(K)=R(K)+OFFSET*R(28+K)+RI(7)*VEC(K)-TLHITE(7)*TA(K)
   42   CONTINUE
C...NO - INVALID ANGLE FOR THIS CUTTER
      ELSE
        CALL CFORM(' *** WARNING - UNABLE TO POSITION 7 SEGMENT CUTTER A
     1T CURRENT SURFACE POINT - SPHERICAL CUTTER ASSUMED',DARRAY,1,101)
        CALL CPRINT(DARRAY)
C
        DO 45 K=1,3
          TE(K)=TBEF(K)
   45   CONTINUE
      ENDIF
C...PRINT USEFUL DIAGNOSTIC INFORMATION WHEN WARNINGS OCCUR
      CALL BAD(3,1,'SP  ',R(1))
      CALL BAD(-3,1,'SN  ',R(29))
      CALL BAD(3,1,'TE  ',TE)
      CALL BAD(-3,1,'TA  ',TA)
      CALL BAD(3,1,'VEC ',VEC)
      CALL BAD(-3,1,'VECL',VECL)
      CALL BAD(-1,1,'TANG',TANG)
      GO TO 999
   50 CONTINUE
C--TOOL TIP ON PART SURFACE
      DO 60 I=1,3
```

```
       IF (ITAC.EQ.1) THEN
C.... APPLY CUTANG BETA TO TA IN DIRECTION OF MOTION
        TA(I)=COSB*DR(16+I) + PDIR*SINB*DR(12+I)
       ELSE
        TA(I)=DR(16+I)
       ENDIF
      TE(I)=R(I)+SIDE*TOLCOM*R(I+28)
      TBEF(I)=TE(I)
   60 CONTINUE
C
  999 CONTINUE
C
      IF(IBUG.NE.11) GO TO 1999
      CALL BAD(-1,0,' ',0)
      BADMSG=' AFTER TULPOS'
      CALL CPRINT(BADMSG)
      CALL BAD(-1,1,'TCOM',TOLCOM)
      CALL BAD(3,1,'TE  ',TE)
      CALL BAD(-3,1,'TA  ',TA)
      CALL BAD(-3,1,'TBEF',TBEF)
 1999 CONTINUE
C
      RETURN
      END
```

6 Subroutine Descriptions

Detailed descriptions of the subroutines in the two new modules INTCHECK and GOUGECHK follow, they are in alphabetic order.

6.1 **Module INTCHECK**

6.1.1 ALGO1

Language: FORTRAN 77

Type: Subroutine

Purpose: To calculate DU,DV , estimates of the incremental step out in U and V aimed at determining a point on the surface and a corresponding point on the tool with coincident normals, using Algorithm 1.

Which states:

Given a surface point R and surface normal RN and a point on the tool T with the same normal, compute the point B where this tool normal intersects the tangent plane at R, then by induced parameterization on the tangent plane determine the step out in DU and DV on the surface.

Calling sequence:

```
CALL ALGO1(T,R,RN,RU,RV,DU,DV)
```

Arguments:

Variable	Type	Description
T	D.P.	Point on tool with same normal as R
R	D.P.	Surface point
RN	D.P.	Surface normal at R
RU	D.P.	Surface tangent in U direction at R
RV	D.P.	Surface tangent in V direction at R
DU	D.P.	Estimated U increment
DV	D.P.	Estimated V increment

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
V4	D.P.	Work vector
UVSTEP	D.P.	UV stepout on tangent plane

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Calculate the UV stepout on the tangent plane at $\underline{R}$, as follows:

$$\underline{UVSTEP} = (\underline{T}-\underline{R}) - ((\underline{T}-\underline{R}) . \underline{RN}) * \underline{RN}$$

Then routine VECMOD is called to compute the step length $S = | \underline{UVSTEP} |$, prior to calling the routine DEGSOL to compute DU and DV from the induced parameterization of the tangent plane at $\underline{R}$.

Notes: NONE

Referenced by: MIXAL

Subsidiaries: VECMOD
DEGSOL

6.1.2 ALGO2

Language: FORTRAN 77

Type: Subroutine

Purpose: Calculate DU,DV , estimates of the incremental step out in U and V aimed at determining a point on the surface and a corresponding point on the tool with coincident normals, using Algorithm 2.

Which states:

Given an initial surface point R, and the tool normal TN at the 'nearest' point on the tool, using a local quadratic approximation to the surface determine DU,DV of point S with same normal TN.

Calling sequence:

```
CALL ALGO2(TN,RU,RV,RUU,RUV,RVV,DU,DV,IOK)
```

Arguments:

Variable	Type	Description
TN	D.P.	Tool normal at 'nearest' point on tool
RU	D.P.	First derivative wrt U at R
RV	D.P.	First derivative wrt V at R
RUU	D.P.	)
RUV	D.P.	) Second derivatives at R
RVV	D.P.	)
DU	D.P.	Estimated DU using algorithm 2
DV	D.P.	Estimated DV using algorithm 2
IOK	LOGICAL	Flag indicating that DU,DV have been calculated

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
A	D.P.	)
B	D.P.	)
C	D.P.	) Work variables
D	D.P.	)
E	D.P.	)
F	D.P.	)

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

First, the determinant of the second fundamental matrix of the surface at $\underline{R}$ is computed.

$$F = (\underline{TN}.\underline{RUV})*(\underline{TN}.\underline{RUV}) + (\underline{TN}.\underline{RUU})*(\underline{TN}.\underline{RVV})$$

Then, provided that abs(F) is greater than or equal to 1.D-6, DU and DV are computed as follows:

$$DU = ((\underline{TN}.\underline{RU})*(\underline{TN}.\underline{RVV}) - (\underline{TN}.\underline{RUV})*(\underline{TN}.\underline{RV})) / F$$

$$DV = ((\underline{TN}.\underline{RV})*(\underline{TN}.\underline{RUU}) - (\underline{TN}.\underline{RUV})*(\underline{TN}.\underline{RU})) / F$$

and the return flag IOK is set .TRUE..

If abs(F) is less than 1.D-6 then at least one curvature is zero, that is the surface is locally a ruled surface, therefore it is not possible to compute DU and DV using Algorithm 2, so IOK is returned as .FALSE..

Notes: NONE

Referenced by: MIXAL

Subsidiaries: DOTF

6.1.3 AVCTRL

Language: FORTRAN 77

Type: Subroutine

Purpose: To process the AVCTRL statement, set flags and store parameters.

Calling sequence:

```
CALL AVCTRL(DONOFF,DAVTP1,G,BMN,BMX,DAVTP2,DMX)
```

Arguments:

Variable	Type	Description
DONOFF	D.P.	ON or OFF Invokes or cancels avoidance control
DAVTP1	D.P.	Type1 avoidance control (modification of tool axis) CUTANG - Modify tilt angle NULL - No modification of tool axis
G	D.P.	Constant technological angle (degrees)
BMN	D.P.	Minimum tilt angle (degrees)
BMX	D.P.	Maximum tilt angle (degrees)
DAVTP2	D.P.	Type2 avoidance control (lift off) NORMPS - Lift off normal to PS AXIS - Lift off parllel to tool axis NULL - None specified (axis assumed)
DMX	D.P.	Maximum lift off distance

Global Variables:

Variable	Block	Type	Description
CTANG	AVCPRM	D.P.	Constant technological angle
BMIN	AVCPRM	D.P.	Minimum tilt angle
BMAX	AVCPRM	D.P.	Maximum tilt angle
DMAX	AVCPRM	D.P.	Maximum linear lift off
LATYP1	AVCPRM	LOGICAL	Flag indicating if type1 (TILT) avoidance required
LANPS	AVCPRM	LOGICAL	Flag indicating type of linear lift off .TRUE. - Lift off normal to PS .FALSE. - Lift off parallel to tool axis
DPMAX	FXCOR	D.P.	Max. allowable step length
IBUG	IBUGG	INTEGER	Flag that forces debug printout
Z0	ZNUMBR	D.P.	Value 0.D0

Local Variables:

Variable	Type	Description
ONOFF	C*6	Character version of DONOFF
AVTYP1	C*6	" " " DAVTP1

AVTYP2	C*6	" " " DAVTP2
NWD	I	No of integer words occupied by encoded character string
MSG	C*30	Debug print message

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

The arguments represent the parameters in the AVCTRL statement. On entry the double precision arguments DONOFF, DAVTY1 and DAVTP2 are processed by HOLFRM to give the decoded character string variables ONOFF, AVTYP1 and AVTYP2.

If ONOFF is the string 'ON' then the flag LAVCTL is set .TRUE., otherwise it is set .FALSE. .

The remainder of the arguments are then examined.

If both AVTYP1 and AVTYP2 are 'NULL' then the avoidance control parameters in common block AVCPRM are not changed since either AVCTRL?ON or AVCTRL/OFF have been programmed.

If AVTYP1 (type 1 avoidance control) is 'CUTANG' then LATYP1 is set .TRUE. and the arguments G, BMN and BMX (angular values in degrees) are converted to radians and stored in CTANG, BMIN and BMAX respectively in common block AVCPRM. Otherwise, LATYP1 is set .FALSE. and CTANG, BMIN and BMAX are set to zero.

If AVTYP2 (type 2 avoidance control) is 'NORMPS' the LANPS is set .TRUE. otherwise it is set .FALSE. .

Finally, if the absolute value of DMX is less than 1.D-6, that is no limit on lift off has been specified then DMAX is set equal to DPMAX, otherwise the lift off limit DMX is stored in DMAX.

Notes: NONE

Referenced by: XCALL

Subsidiaries: BAD
CPRINT
HOLFRM

6.1.4 AVMSG

Language: FORTRAN 77

Type: Subroutine

Purpose: Print message when interference successfully avoided

Calling sequence:

```
CALL AVMSG(TE,TA,LANPS,B,TYPE,D)
```

Arguments:

Variable	Type	Description
TE	D.P.	Tool end coordinates
TA	D.P.	Tool axis vector
LANPS	LOGICAL	Lift off direction vector
B	D.P.	Tilt angle (if any)
TYPE	C*5	Type of tilt (ROLL or PITCH)
D	D.P.	Lift off distance

Global Variables:

Variable	Block	Type	Description
Z0	ZNUMBR	D.P.	Value 0.D0
Z1EM6	ZNUMBR	D.P.	Value 1.D-6

Local Variables:

Variable	Type	Description
MSG	C*120	Print buffer
IPOS	INTEGER	Location in print buffer

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Augment the basic message 'INTERFERENCE SUCESSFULLY AVOIDED BY' with details of the avoidance action taken, selected from the following.

[(PITCHING / ROLLING) THROUGH b DEGREES] [AND] [LIFTING OFF BY d (NORMAL TO PS / PARALLEL TO TA)]

Notes: NONE

Referenced by: AVOID

Subsidiaries: CFORM
CPRINT
FCONV

6.1.5 AVOID

Language: FORTRAN 77

Type: Subroutine

Purpose: To carry out avoidance strategies when interference has been detected.

Calling sequence:

```
CALL AVOID(PSURF,IPSIZE,PSIDE,PTHICK,PTOLI,PTOLO,TOLCOM,
+                FWD,TUL,TE,TA,R,NPAT,UP,VP,INTLOC,LRET)
```

Arguments:

Variable	Type	Description
PSURF	D.P.	Canonical form of part surface
IPSIZE	INTEGER	Size of PS canonical form
PSIDE	D.P.	1 or -1 to adjust surface normal (RU^RV)
PTHICK	D.P.	Thickness offset from surface
PTOLI	D.P.	) PS tolerances
PTOLO	D.P.	)
TOLCOM	D.P.	Tolerance band compensation
FWD	D.P.	DS forward direction
TUL	D.P.	Tool geometry
TE	D.P.	Tool end coordinates
TA	D.P.	Tool axis vector
R	D.P.	Surface contact point and derivatives
NPAT	INTEGER	Patch number for contact point
UP	D.P.	U parameter of contact point
VP	D.P.	V parameter of contact point
INTLOC	INTEGER	Location interference table of worst penetration
LRET	LOGICAL	Return flag .TRUE. if interference avoided .FALSE. if there is still interference

Global Variables:

Variable	Block	Type	Description
CTANG	AVCPRM	D.P.	Constant technological angle
BMIN	AVCPRM	D.P.	Minimum tilt angle
BMAX	AVCPRM	D.P.	Maximum tilt angle
DMAX	AVCPRM	D.P.	Maximum linear lift off
LATYP1	AVCPRM	LOGICAL	Flag indicating if type1 (TILT) avoidance required
LANPS	AVCPRM	LOGICAL	Flag indicating type of linear lift off .TRUE. - Lift off normal to PS .FALSE. - Lift off parallel to tool axis
AVLIST	AVCPRM	LOGICAL	Flag indicating that avoidance details are to be printed
DINTR	INTERF	D.P.	Penetration distance

SNINTR	INTERF D.P.	Surface normal at penetration
Z0	ZNUMBR D.P.	Value 0.D0
Z1	ZNUMBR D.P.	Value 1.D0
Z1E6	ZNUMBR D.P.	Value 1.D6
Z1EM6	ZNUMBR D.P.	Value 1.D-6

Local Variables:

Variable	Type	Description
B	D.P.	Computed tilt angle to clear interference
UL	D.P.	Local axis in FWD direction perpto TA
S	D.P.	Sign of tilt angle (+ve in FWD direction)
TCCENT	D.P.	Centre of tool contact sphere
TCRAD	D.P.	Radius of tool contact sphere
CX	D.P.	Principle curvatures, directions and coefficients
RCMIN	D.P.	Minimum radius of curvature of surface at contact point
NPS	D.P.	Part surface normal at point of contact
VL	D.P.	Lift off direction
D1	D.P.	)
TEMP	D.P.	) Local work variables
D	D.P.	Lift off distance
E	D.P.	Curvature compensation
MSG	C*120	Print buffer
MSG1	C*52	)
MSG2	C*52	) Messages
IPOS	INTEGER	Location in print buffer
IRET	LOGICAL	INTCHK return flag - .TRUE. if interference detected
JRET	LOGICAL	CALMXB return flag - .TRUE. if beta calc. OK
BDEG	D.P.	B in degrees
TYPE	C*5	Type of tilt (ROLL or PITCH)
OLDTE	D.P.	Original TE
OLDTA	D.P.	Original TA
MAXK	D.P.	Maximum curvature

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

On entry the surface normal at the point of contact of the tool with the surface, pointing towards the tool, is determined. Then KURSRF is called to compute the principle curvatures of the surface at the contact point. From these the minimum radius of curvature of the surface at the contact point is determined.

If avoidance strategy type 1 has been specified (**LATYP1=.TRUE.**) then an attempt is made to avoid interference by tilting the tool in the most appropriate direction (rolling or pitching) . This is done by calling routine CALCB which first determines the direction (dependent on the relative location of the worst penetration point with respect to the contact point), then calculates the tilt angle required to just avoid the surface at the worst point of interference (INTLOC). The direction of tilt is indicated by the local axis UL perpendicular to TA that defines the plane of tilt and the type of tilt by TYPE (ROLL or PITCH). The constant technological angle, CTANG, needed to clear the surface is then added to B, and the resultant angle checked to ensure that it is within the limits BMIN and BMAX, if not it is constrained to be within the limits.

After saving the current toolend coordinates and toolaxis vector, the new tool axis vector, TA that would result from tilting the tool through B in the direction UL is computed.

$$\underline{TA} = \cos(B)\ \underline{TA} + \sin(B)\ \underline{UL}$$

and the routine TECALC called to compute the resultant new toolend coordinates TE. Then the centre and radius of the tool contact sphere are computed by calling TCSPHR.

A series of checks are then made in order to confirm that interference has been avoided.

First a check is made for possible local interference of the tool with the surface at the point of contact as a result of an increase in the effective radius of the tool caused by tilting. If interference is detected then routine CALMXB is called to compute the maximum angle of tilt so that the tool will fit the surface at the point of contact. This may or may not be possible. If a suitable angle is computed it is again constrained to the permitted limits and yet again a new TA and TE are computed, otherwise the original TE and TA are restored and the tool will be lifted off as the only means of avoiding interference. Suitable messages are output to indicate what action has been taken if PRINT/AVMSG,ON has been programmed (AVLIST=.TRUE.) .

Finally, the new tool position and orientation (TE and TA) are checked for possible interference by calling INTCHK, using the original interference data as start points, if avoidance has been successful the original interference data is restored and if requested AVMSG called to output a suitable message detailing the avoidance action taken.

If tilting has not sucessfully avoided interference or lift off only was selected, the an attempt is made to avoid the unwanted penetration by lifting off in the specified direction.

$$d = \frac{D}{\underline{SN}.\underline{VL}}$$

where D is maximum penetration at INTLOC
SN is unit surface normal at INTLOC
VL is unit vector in direction of lift off

If the surface is concave at the point of contact then an additional compensatory liftoff is computed,

$$e = \sqrt{(RCMIN^2 - D^2 + d^2)} - RCMIN$$

where RCMIN is minimum radius of curvature

The liftoff is finally increased by the outtol value PTOLO and checked to against maximum permitted. If outside limit an appropriate message will be output. Otherwise the new TE will be computed (TA unchanged) and if liftoff normal to the part surface has been selected INTCHK is again called to confirm that no new interference has been generated as a result of taking avoiding action.

Notes: NONE

Referenced by: SSPATH

Subsidiaries: AVMSG
BAD
CALCB
CALMXB
CFORM
CPRINT
DOTV
FCONV
INTCHK
INTMSG
KURSRF
OUTMSG
RESET
RESTOR
TCSPHR
TECALC
VNORM
VSCALE

6.1.6 BDAVCP

Language: FORTRAN 77

Type: Block Data Subprogram

Purpose: Initialization of avoidance control parameters.

Calling sequence: NONE

Arguments: NONE

Variable	Type	Description

Global Variables:

Variable	Block	Type	Description
CTANG	AVCPRM	D.P.	Constant technological angle
BMIN	AVCPRM	D.P.	Minimum tilt angle
BMAX	AVCPRM	D.P.	Maximum tilt angle
DMAX	AVCPRM	D.P.	Maximum linear lift off
LATYP1	AVCPRM	LOGICAL	Flag indicating if type1 (TILT) avoidance required
LANPS	AVCPRM	LOGICAL	Flag indicating type of linear lift off .TRUE. - lift off normal to PS .FALSE. - lift off parallel to tool axis
AVLIST	AVCPRM	LOGICAL	Flag indicating that avoidance details are to be printed

Local Variables: NONE

Variable	Type	Description

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Initialized via DATA statements.

Notes: NONE

Referenced by: NONE

Subsidiaries: NONE

6.1.7 BDINID

Language: FORTRAN 77

Type: Block Data Subprogram

Purpose: Initialization of Interference Checking initial data.

Calling sequence: None

Arguments: None

Variable	Type	Description

Global Variables:

Variable	Block	Type	Description
CXHULL	INIDAT	D.P.	Spherical surface enclosures (hulls)
DULIM	INIDAT	D.P.	DU iteration limit for patches
DVLIM	INIDAT	D.P.	DV iteration limit for patches
CURMAX	INIDAT	D.P.	Maximum curvature of patches
CURMIN	INIDAT	D.P.	Minimum curvature of patches
NCXHUL	INIDAT	INTEGER	Number of hulls
NGP	INIDAT	INTEGER	Number of patches in current surface
ICURSF	INIDAT	INTEGER	External LDA file record number for current surface

Local Variables: NONE

Variable	Type	Description

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Initialized by DATA statements. Integer Parameter MXNP (maximum number of patches) controls the dimension of arrays.

Notes: NONE

Referenced by: NONE

Subsidiaries: NONE

6.1.8 BDINT

Language: FORTRAN 77

Type: Block Data Subprogram

Purpose: Initialize Interference Table.

Calling sequence: NONE

Arguments: NONE

Variable	Type	Description

Global Variables:

Variable	Block	Type	Description
DINTR	INTERF	D.P.	Penetration distance
UINTR	INTERF	D.P.	U parameter of surface point at penetration
VINTR	INTERF	D.P.	V parameter of surface point at penetration
LDINTR	INTERF	D.P.	Previous penetration distance
LUINTR	INTERF	D.P.	U parameter of surface point at previous penetration
LVINTR	INTERF	D.P.	V parameter of surface point at previous penetration
TPINTR	INTERF	D.P.	Coordinates of point on tool at penetration
SPINTR	INTERF	D.P.	Coordinates of point on surface at penetration
SNINTR	INTERF	D.P.	Surface normal at penetration
PINTR	INTERF	INTEGER	Patch number for surface point at penetration
LPINTR	INTERF	INTEGER	Patch number for surface point at previous penetration
ISEG	INTERF	INTEGER	Tool segment for penetration
NPINTR	INTERF	INTEGER	Number of interference points detected
LNPINT	INTERF	INTEGER	Previous number of interference points detected

Local Variables: NONE

Variable	Type	Description

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Initialized via DATA statements. The Integer Parameter MAXNP (maximum number of patches) controls the array dimensions.

Notes: NONE

Referenced by: NONE

Subsidiaries: NONE

6.1.9 CALCB

Language: FORTRAN 77

Type: Subroutine

Purpose: To compute tilt angle B to avoid interference at worst location (INTLOC).

Calling sequence:

```
CALL CALCB(W,FWD,R,INTLOC,B,TYPE,TL)
```

Arguments:

Variable	Type	Description
W	D.P.	Axis through contact point parallel to TA
FWD	D.P.	DS forward direction
R	D.P.	Surface contact point and derivatives
INTLOC	INTEGER	Location in interference table of worst penetration
B	D.P.	Tilt angle to clear interference
TYPE	C*5	Type of tilt (ROLL or PITCH)
TL	D.P.	Approp. local axis that defines plane of tilt with TA

Global Variables:

Variable	Block	Type	Description
DINTR	INTERF	D.P.	Penetration distance
UINTR	INTERF	D.P.	U parameter of surface point at penetration
VINTR	INTERF	D.P.	V parameter of surface point at penetration
LDINTR	INTERF	D.P.	Previous penetration distance
LUINTR	INTERF	D.P.	U parameter of surface point at previous penetration
LVINTR	INTERF	D.P.	V parameter of surface point at previous penetration
TPINTR	INTERF	D.P.	Coordinates of point on tool at penetration
SPINTR	INTERF	D.P.	Coordinates of point on surface at penetration
SNINTR	INTERF	D.P.	Surface normal at penetration
PINTR	INTERF	INTEGER	Patch number for surface point at penetration
LPINTR	INTERF	INTEGER	Patch number for surface point at previous penetration
ISEG	INTERF	INTEGER	Tool segment for penetration
NPINTR	INTERF	INTEGER	Number of interference points detected
LNPINT	INTERF	INTEGER	Previous number of interference points detected

Local Variables:

Variable	Type	Description
U	D.P.	Local axis in forward direction perpto W
V	D.P.	Local axis orthogonal to W and U
S	D.P.	SPINTR at INTLOC wrt local origin at contact point
T	D.P.	TPINTR at INTLOC wrt local origin at contact point
SPA	D.P.	Projection of S onto local plane of tilt
TPA	D.P.	Projection of T onto local plane of tilt
CPHI	D.P.	cos(PHI)
PHI	D.P.	Tilt angle
A	D.P.	TPA X SPA
SI	D.P.	Direction of positive PHI
SU	D.P.	Projection of S onto U axis
SV	D.P.	Projection of S onto V axis

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

On entry to the routine the local axes (U,V,W) at the contact point are computed.

$$\underline{W} = \underline{TA}$$
$$\underline{V} = \underline{W} \wedge \underline{FWD}$$
$$\underline{U} = \underline{V} \wedge \underline{W}$$

Then vectors S and T from the contact point to the pair of corresponding points on the surface and tool at the inteference point (SPINTR and TPINTR) are computed. S is then projected onto the local UV plane in order to determine the direction of tilt. i.e. whether to 'roll' about the U axis or 'pitch' about the V axis.

S and T are then projected onto the local plane of tilt and the angle PHI between them computed and given the appropriate sign. This angle B, the type of tilt, TYPE and the appropriate local axis that together with the toolaxis TA defines the plane of tilt, TL are returned to the calling routine.

Notes: NONE

Referenced by: AVOID

Subsidiaries: CROSS
CROSSV
DOTF
VNORM

6.1.10 CALCT

Language: FORTRAN 77

Type: Subroutine

Purpose: To calculate T, point on tool with same normal as surface at point R, RN.

Calling sequence:

```
CALL CALCT(TUL,TOOLCE,TOOLAX,R,RN,PSIDE,T,LBASE)
```

Arguments:

Variable	Type	Description
TUL	D.P.	Tool geometry
TOOLCE	D.P.	Tool center point
TOOLAX	D.P.	Toolaxis vector
R	D.P.	Surface point
RN	D.P.	Surface normal
PSIDE	D.P.	Tool side of surface
T	D.P.	Point on tool with same normal
LBASE	LOGICAL	Flag indicating that T is on base flat of cutter

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
ETA6	D.P.	10.E-6
V1	D.P.	)
V2	D.P.	) work arrays
A	D.P.	Work variable
A1	D.P.	Cosine of angle between SN and TA
B	D.P.	)
C	D.P.	) Work variables
D	D.P.	)
SN	D.P.	Surface normal pointing towards tool

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

On entry the length of the tool cylinder, TCYLEN is calculated and the evaluated surface normal RN, is adjusted to the tool side of the

surface, SN, and the cosine of the angle between SN and TA computed.

The point on the tool with the same normal as the surface normal SN is then determined. This depends on the shape of the tool and the relative direction of SN with respect to TA.

If SN.TA is less than zero then T is taken as the toolend and LBASE set to .TRUE. to force algorithm 2 in MIXAL.

Otherwise, if the cutter is ballended, T will be located on the hemi-spherical end or on the cylindrical shaft dependent on the angle between the surface normal and the toolaxis.

If |SN.TA| > 1.D-6 then T will be on the ball end,

$$T = TOOLCE - rSN \quad \text{where } TOOLCE = TE + rTA$$

otherwise T will be on the cylinder,

$$T = TOOLCE - rSN + A\ TA$$

where $A = (R - TOOLCE).TA$ limited to the range $0 <= A <= TCYLEN$

Note: SN is perpendicular to TA

If the cutter is corner radius, T will be located on the corner radius, the flat base or the cylindrical shaft dependent on the angle between the surface normal and the toolaxis.

If |SN.TA| < 1.D-6 then T will be on the cylinder,

$$T = TOOLCE - rSN + A\ TA$$

where $A = (R - TOOLCE).TA$ limited to the range $0 <= A <= TCYLEN$

Note: SN is perpendicular to TA

If (1 - SN.TA) < 1.D-6 then T will be on the flat base and therefore LBASE is set .TRUE. to force algorithm 2 in MIXAL. If the radial distance D of T from the tool axis is taken as the minimum of the radial distance of R from the tool axis and the offset of the corner radius center from the tool axis (TUL(3)),

$$T = TE + D\ V$$

where V is the unit vector that is the projection of the vector from TOOLCE to R on to the plane of the base of the cutter:

Otherwise,

$$T = TOOLCE + e\ V - rSN$$

where V is unit vector that is the projection of the vector from TOOLCE to R on to a plane perpendicular to the tool axis.

Notes: NONE

Referenced by: MIXAL

Subsidiaries: DOTF
NORMA
VECMOD

6.1.11 CALMXB

Language: FORTRAN 77

Type: Subroutine

Purpose: To compute maximum tilt angle where tool will fit into concave surface.

Calling sequence:

```
CALL CALMXB(RS,TUL,W,U,SN,BETA,LRET)
```

Arguments:

Variable	Type	Description
RS	D.P.	Minimum radius of curvature of surface at contact point
TUL	D.P.	Tool geometry
W	D.P.	Original tool axis
U	D.P.	Local axis direction orthogonal to W in direction of positive tilt
SN	D.P.	Surface normal at point of contact (towards tool)
BETA	D.P.	Tilt angle Value on input will be ideal tilt to clear surface and will be used to determine which of the two alternative is to be returned as the maximum
LRET	LOGICAL	Return flag = .TRUE. - Computed OK, else .FALSE.

Global Variables:

Variable	Block	Type	Description
Z0	ZNUMBR	D.P.	0.D0
Z1	ZNUMBR	D.P.	1.D0
Z1EM6	ZNUMBR	D.P.	1.D-6

Local Variables:

Variable	Type	Description
SPHI	D.P.	sin(PHI) where PHI is angle between TA
CPHI	D.P.	cos(PHI) and SN when tool and surface curvatures match
A	D.P.	)
B	D.P.	) work variables
C	D.P.	)
D1	D.P.	)
DISC	D.P.	Discriminant of quadratic equation
SB1	D.P.	)
SB2	D.P.	) two possible values for sin(BETA)
B1	D.P.	)

B2	D.P.	) two possible values of BETA
BNEW	D.P.	New value of BETA

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Compute sine of angle between tool axis (W) and SN such that tool curvature will fit surface curvature at point of contact.

$$\sin(PHI) = \frac{e}{(Rs - r)}$$

where Rs minimum radius of curvature of the surface
e offset of corner radius center from tool axis
r corner radius

and thence cos(PHI).

BETA has to satisfy two equations

$$\underline{TA} = \cos(BETA)*\underline{W} + \sin(BETA)*\underline{U} \quad (1)$$
$$\underline{TA}.\underline{SN} = \cos(PHI) \quad (2)$$

Taking SN . equation (1) and substituting for TA.SN, results in

$$\cos(PHI) = \cos(BETA)*(\underline{W}.\underline{SN}) + \sin(BETA)*(\underline{U}.\underline{SN})$$

Squaring and substituting for $\cos^2(BETA)$ will give a quadratic in sin(BETA) - hence BETA. The appropriate solution is selected dependent on the ideal BETA to clear surface as specified in call to the routine. If there is no solution to the quadratic then BETA is taken as zero and LRET returned as .FALSE..

Notes: NONE

Referenced by: AVOID

Subsidiaries: DOTF

6.1.12 CHKSRF

Language: FORTRAN 77

Type: Subroutine

Purpose: Process CHKSRF statement

Calling sequence:

```
CALL CHKSRF(SURF,DONOFF)
```

Arguments:

Variable	Type	Description
SURF	D.P.	= PS Check for interference with part surface = surface_name check for interference with named surface NOTE: Only PS implemented
DONOFF	D.P.	= ON Set interference checking flag = OFF Cancel interference checking flag

Global Variables:

Variable	Block	Type	Description
LINTCK	RMFLGS	LOGICAL	Interference checking flag

Local Variables:

Variable	Type	Description
NWD	INTEGER	Number of words occupied by encoded character string
TEST	C*2	Decoded character string

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

If SURF is PS then the second argument DONOFF is checked, otherwise the CHKSRF statement is ignored in the current implementation.

If DONOFF is ON the flag LINCHK is set .TRUE., otherwise LINCHK is set .FALSE..

Notes: NONE

Referenced by: XCALL

Subsidiaries: HOLFRM

6.1.13 CURCAL

Language: FORTRAN 77

Type: Subroutine

Purpose: To calculate principal curvatures CURVE1,CURVE2

Calling sequence:

```
CALL CURCAL(RU,RV,RUU,RUV,RVV,RNORM,SIDE,CURVE1,CURVE2)
```

Arguments:

Variable	Type	Description
RU	D.P.	1st partial derivative of R wrt U
RV	D.P.	1st partial derivative of R wrt V
RUU	D.P.	2nd partial derivative of R wrt U
RUV	D.P.	2nd mixed partial derivative wrt U & V
RVV	D.P.	2nd partial derivative of R wrt V
RNORM	D.P.	Surface normal at point R
SIDE	D.P.	Tool side of surface
CURVE1	D.P.	1st principal curvature at R
CURVE2	D.P.	2nd principal curvature at R

Global Variables:

Variable	Block	Type	Description
Z4	ZNUMBR	D.P.	4.D0
ZERO	ZNUMBR	D.P.	0.D0

Local Variables:

Variable	Type	Description
W	D.P.	Work array
G11	D.P.	)
G22	D.P.	)
G12	D.P.	)
D11	D.P.	)
D12	D.P.	) Work variables
D22	D.P.	)
A1	D.P.	)
A2	D.P.	)
A3	D.P.	)

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

The principal curvatures of a surface at a given point are computed by analysing the first and second order derivatives at the point.
The first fundamental matrix G, first order coefficients is

$$G = \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}$$

where $$g_{ij} = \frac{d\underline{R}}{dt_i} \cdot \frac{d\underline{R}}{dt_j} \qquad i,j = 1,2 \qquad (t_1 = u \text{ and } t_2 = v)$$

and the second fundamental matrix D, second order coefficents is

$$D = \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix}$$

where $$d_{ij} = \frac{dd\underline{R}}{dt_i * dt_j} \cdot \underline{RN}$$

The principal curvatures satisfy the following quadratic equation

$$A\ K*K + B\ K + C = 0$$

where $$A = g_{11}*g_{22} - 2*g_{12}$$

$$B = -(d_{11}*g_{22} - 2*d_{12}*g_{12} + d_{22}*g_{11})$$

$$C = d_{11}*d_{22} - 2*d_{12}$$

This is quadratic solved for the two values.

Notes: If the discriminant $B^2 - 4AC$ is less than zero, it is taken as zero.

Referenced by: INEX

Subsidiaries: DOTF

6.1.14 CURINT

Language: FORTRAN 77

Type: Subroutine

Purpose: To test current patch for ball-ended and corner radius cutters. Uses mixed algorithm method (MIXAL).

Calling sequence:

```
 CALL CURINT(PSURF,IPSIZE,PSIDE,PTHICK,TOLIN,TUL,TE,TOOLAX,
+            RC,NPATCH,UC,VC)
```

Arguments:

Variable	Type	Description
PSURF	D.P.	Canonical form of surface
IPSIZE	INTEGER	Size of surface canonical form
PSIDE	D.P.	1 or -1 to adjust surface normal
PTHICK	D.P.	Thickness offset from surface
TOLIN	D.P.	Surface intol
TUL	D.P.	Tool geometry data
TE	D.P.	Abs. tool tip coords
TOOLAX	D.P.	Tool axes dirn. cosines
RC	D.P.	Current point of contact and its derivatives
NPATCH	INTEGER	Current patch number
UC	D.P.	U coord. of contact point
VC	D.P.	V coord. of contact point

Global Variables:

Variable	Block	Type	Description
IBUG	IBUGG	INTEGER	Flag that forces debug printout
DULIM	INIDAT	D.P.	DU iteration limit for patches
DVLIM	INIDAT	D.P.	DV iteration limit for patches
CURMAX	INIDAT	D.P.	Maximum curvature of patches
DINTR	INTERF	D.P.	Penetration distance
UINTR	INTERF	D.P.	U parameter of surface point at penetration
VINTR	INTERF	D.P.	V parameter of surface point at penetration
TPINTR	INTERF	D.P.	Coordinates of point on tool at penetration
SPINTR	INTERF	D.P.	Coordinates of point on surface at penetration
SNINTR	INTERF	D.P.	Surface normal at penetration
PINTR	INTERF	INTEGER	Patch number for surface point at penetration
ISEG	INTERF	INTEGER	Tool segment for penetration
NPINTR	INTERF	INTEGER	Number of interference points detected

Local Variables:

Variable	Type	Description
A	D.P.	Work variable
CORRAD	D.P.	Corner radius of tool
CT	D.P.	Tool curvature
CURTOL	D.P.	Curvature tolerance
CUTR3	D.P.	CUTRAD-CORRAD
CUTRAD	D.P.	Radius of tool cylinder
CYLENG	D.P.	Length of tool cylinder
D1	D.P.	Work variable
DU	D.P.	U stepout from contact point for iteration start point
DU1	D.P	U step based on curvature of surface at contact point
DUMIN	D.P.	Minimum U step
DUMIN2	D.P.	DUMIN*DUMIN
DV	D.P.	V stepout from contact point for iteration start point
DV1	D.P.	V step based on curavture of surface at contact point
DVMIN	D.P.	Minimum V step
DVMIN2	D.P.	DVMIN*DVMIN
HRAD	D.P.	Radius of tool enveloping sphere
HCENT	D.P.	Centre of tool enveloping sphere
I,J,K	INTEGER	Loop counts
ICUR	INTEGER	Pointer to end of non-current patch interference data in interference table
IEDGE	INTEGER	Flag indicating if last DU or DV was limited
IFLAG	INTEGER	Flag indicating if interference algorithm converged
IPEN	INTEGER	Indicates which segment of tool inteference occurred on
ITOOL	INTEGER	Cutter type: 1 for ball-ended, 2 for corner radius
NHIT	INTEGER	Number of hits
NITERA	INTEGER	No. of iterations
PENMAX	D.P.	Penetration (hit +ve)
PNORM	D.P.	Surface normal at penetration
PPOINT	D.P.	Surface point at penetration
PREVDU	D.P.	Previous DU
PREVDV	D.P.	Previous DV
TLEN	D.P.	Tool length
TOOLCE	D.P.	Tool centre
TPOINT	D.P.	Point on tool at pentration
U	D.P.	U parameter
UPEN	D.P.	U value at point of penetration
V	D.P.	V parameter
VPEN	D.P.	V value at point of penetration
V1	D.P.	)
V2	D.P.	) work arrays
V3	D.P.	)

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

On entry the tool data, TUL is transferred into the local variables, CORRAD, CUTRAD, CUTR3 and TLEN, and CYLENG is calculated. Then the curvature tolerance, CURTOL is computed based on the tolerance, TOLIN and the cutter geometry.

Then the following test schedule for current patch is performed:-

1. Test for the patch being convex or 'pseudo' convex by comparing the maximum curvature for the patch, CURMAX against CURTOL. If it is then no local interference should occur.

2. Call TXHULL to calculate the curvature CT of the minimum spherical convex hull for the tool with the cutting point as a diametral point of contact. Test against the maximum curvature of the patch PATCUR (Minimum radius of curvature).
   If CP <= CT no local interference can occur.

3. Otherwise MIXAL is used to search for interference on the patch, using a number of starting points arranged in a pattern about the cutting point. (The cutting point cannot be used as a starting point for these searches, since it is a solution). Five start points are actually used. One in the forward direction, one to each side, and two in the rearward direction with respect to the point of contact, at distances approximately, half the cutter radius (D/4) and in the case of the second rearward point 3D/4.

   All non-coincident penetration points are stored in the interference table. That is, if MIXAL converges to the same point as a result of different start points only one solution will be saved.

Notes: NONE

Referenced by: INTCHK

Subsidiaries: BAD
CROSS
DEGSOL
JMPLT2
MIXAL
TXHULL
VNORM

6.1.15 DEGSOL

Language: FORTRAN 77

Type: Subroutine

Purpose: Solves degenerate eqn. system $A.\underline{V1} + B.\underline{V2} = \underline{V}$
where V, V1, V2, 3-D vectors and A,B scalars.
For no solution, IFLAG set to 1, else zero.

Calling sequence:

```
CALL DEGSOL(V,V1,V2,A,B,IFLAG)
```

Arguments:

Variable	Type	Description
V	D.P.	)
V1	D.P.	) input vectors (see equation above)
V2	D.P.	)
A	D.P.	Scalar result
B	D.P.	Scalar result
IFLAG	INTEGER	Return flag

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

The largest absolute value of the three determinants,

$$\begin{vmatrix} v1_i & v1_k \\ v2_j & v2_k \end{vmatrix}$$ is determined.

If all three determinants are zero (< ETA=1.d-10) there is no solution, therefore IFLAG is returned=1. Otherwise, A and B are computed.

Notes: NONE

Referenced by: ALGO1
CURINT

Subsidiaries: NONE

6.1.16 DIST2

Language: FORTRAN 77

Type: Double Precision Function

Purpose: Compute the square of the distance between two points.

Calling sequence:

```
VALUE = DIST2(A,B)
```

Arguments:

Variable	Type	Description
A	D.P.	Coordinates of first point
B	D.P.	Coordinates of second point

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Square and add the differences between the X,Y and Z coordinates of the two points A nd B.

Notes: NONE

Referenced by: REMSCN

Subsidiaries: NONE

6.1.17 EVALSS

Language: FORTRAN 77

Type: Subroutine

Purpose: Evaluate the point, first and second derivatives of a surface given the patch number and its u,v parametric coordinates.

Calling sequence:

```
 CALL EVALSS(NPATCH,U,V,ISIZE,GEO,MODE,
1                  R,RU,RV,RUU,RUV,RVV,RNORM,NORMOK)
```

Arguments:

Variable	Type	Description
NPATCH	INTEGER	Patch number
U	D.P.	U Parameter
V	D.P.	V Parameter
ISIZE	INTEGER	Size of canonical form of surface
GEO	D.P.	Surface canonical form
MODE	INTEGER	Flag indicating if point only (=0) or point and derivatives (=1) are to be calculated
R	D.P.	Coordinates of point
RU	D.P.	First derivative wrt U at R
RV	D.P.	First derivative wrt V at R
RUU	D.P.	)
RUV	D.P.	) Second derivatives at R
RVV	D.P.	)
RNORM	D.P.	Unit surface normal at R
NORMOK	INTEGER	Flag indicating normal computed satifactorily.

Global Variables:

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
B	D.P.	) Current Patch data
IFLAG	D.P.	)
ITOP	INTEGER	Current Patch topology
SPV	D.P.	Point and derivatives
ICPAT	INTEGER	Current patch number

Files: NONe

Restrictions: NONE

Diagnostics: NONE

Method:

On entry the routine checks if the data for the patch on which the point is to be evaluated is already loaded in B, IFLAG and ITOP, if not it is loaded by calling LODPCH. Then CNSURF is called to evaluate the point and its derivatives. These are then transferred from the array SPV into the individual arrays R, RU, RV, RUU, RUV, RVV and RNORM.

Notes: NONE

Referenced by: INEX
MIXAL

Subsidiaries: LODPCH
CNSURF

6.1.18 HULCUT

Language: FORTRAN 77

Type: Subroutine

Purpose: Calculate the penetration of corner radius cutter into spherical convex 'hull' ,SPHR. If no penetration occurs, then ISEG = -1, else the value returned in ISEG indicates the cutter segment that penetrates the sphere. (ISEG = 0 bottom flat of cutter, = 1 corner radius, = 2 cylinder).

Calling sequence:

```
CALL HULCUT(TE,TA,TUL,SPHR,ISEG)
```

Arguments:

Variable	Type	Description
TE	D.P.	Tool end
TA	D.P.	Tool axis
TUL	D.P.	Tool geometry
SPHR	D.P.	Centre and radius of test sphere
ISEG	D.P.	Cutter segment = 0 bottom flat 1 corner radius 2 cylinder

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
V	D.P.	Vector from sphere center to tool center
A	D.P.	Projection of V onto TA
B	D.P.	Length of V
C	D.P.	Perpendicular distance of sphere center from tool axis
P	D.P.	Penetration (-ve value for a miss)

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

The vector V from the center of the sphere to the tool center is calculated. The tool center is the point on the tool axis at the same height as the corner radius center. This is then projected onto the tool axis TA (A) and its length (B) and perpendicular distance (C) from the

tool axis computed.

If the center of the sphere is 'above' the tool center, (A>0), in other words V intersects the tool surface on the cylindrical section, then the penetration P is calculated as follows,

```
P = Rs + d/2 - C        where Rs = sphere radius
                               d  = tool diameter
```

If the center of the sphere is 'below' the tool center and its perpendicular distance from the tool axis (C) is less than the corner radius center offset (e = d/2 - r) then the intersection (if any) will be on the flat base and penetration P is calculated as follows,

```
P = Rs + f - A          where Rs = sphere radius
                               f  = height of tool center above
                                    tool end = ( TUL(4)+TUL(1) )
```

Otherwise, the intersection (if any) will be on the corner radius and P is computed as follows,

```
P = Rs + r - sqrt( D*D - A*A )

                        where Rs = sphere radius
                              r  = tool corner radius
                              D  = C - corner radius offset dist.
```

A negative value of P indicates a hit in which case ISEG is set accordingly,

```
ISEG =  0  bottom flat case
        1  corner radius case
        2  cylinder case
```

If no hit, P positive then ISEG = -1 is returned.

Notes: NONE

Referenced by: REMSCN

Subsidiaries: DOTF
VECMOD

6.1.19 INEX

Language: FORTRAN 77

Type: Subroutine

Purpose: Interference checking initialisation procedure.
Calculates: Maximum and minimum curvatures CURMAX,CURMIN
Iteration limits DULIM, DVLIM
Convex spherical hulls CXHULL

Calling sequence:

```
CALL INEX(GEO,SIDE,TOLI,TOLO)
```

Arguments:

Variable	Type	Description
GEO	D.P.	Surface canonical form
SIDE	D.P.	Tool side of surface w.r.t. surface normal
TOLI	D.P.	Surface intol
TOLO	D.P.	Surface outtol

Global Variables:

Variable	Block	Type	Description
CXHULL	INIDAT	D.P.	Spherical surface enclosures (hulls)
DULIM	INIDAT	D.P.	DU iteration limit for patches
DVLIM	INIDAT	D.P.	DV iteration limit for patches
CURMAX	INIDAT	D.P.	Maximum curvature of patches
CURMIN	INIDAT	D.P.	Minimum curvature of patches
NCXHUL	INIDAT	INTEGER	Number of hulls
NGP	INIDAT	INTEGER	Number of patches in current surface
ICURSF	INIDAT	INTEGER	External LDA file record number for current surface

Local Variables:

Variable	Type	Description
AMESH	D.P.	Mesh size
A	D.P.	Sum of U distances between corner points
B	D.P.	Sum of V distances between corner points
C	D.P.	Distance between corner points
I,J,K,L,M	INTEGER	Loop counts, indices etc.
IREC	INTEGER	External LDA file record number for surface under consideration
ISIZE	INTEGER	Size of surface canonical form
MODE	INTEGER	Evaluation mode
N	INTEGER	No. of intervals between mesh points
N3	INTEGER	Value 3
N4	INTEGER	Value 4
NGRID	INTEGER	Mesh dimension (parameter)

NNGRID	INTEGER	Number of mesh points
NPATCH	INTEGER	Patch number
RNET	D.P.	Mesh points
NCOR	INTEGER	Location of 4 patch corner points in RNET
RCORN1	D.P.	) Patch corner points
RCORN2	D.P.	)
CV1	D.P.	) Patch corner vectors
CV2	D.P.	)
M1	INTEGER	) Pointers for above
M2	INTEGER	)
R	D.P.	Coordinates of point
RU	D.P.	First derivative wrt U at R
RV	D.P.	First derivative wrt V at R
RUU	D.P.	)
RUV	D.P.	) Second derivatives at R
RVV	D.P.	)
RNORM	D.P.	Unit surface normal at R
NORMOK	INTEGER	Flag indicating normal computed OK
TOLIN	D.P.	Surface tolerance
U	D.P.	U parameter
V	D.P.	V parameter
CRVMAX	D.P.	Maximum curvature for patch
CRVMIN	D.P.	Minimum curvature for patch
CURVE1	D.P.	1st principal curvature of surface
CURVE2	D.P.	2nd principal curvature of surface
SCENT	D.P.	Centre of spherical shell
SRAD	D.P.	Radius of spherical shell
NHULL	INTEGER	No. of spherical shells computed per patch
NFLAGS	INTEGER	Array of flags indicating if a sphere has been calculated
RAD	D.P.	Radius of sphere computed by SHELLA
W	D.P.	Centre of sphere computed by SHELLA

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

On entry, a check is made to see if this surface has already been processed by INEX, by comparing IREC, the record number for the start of the surface canonical form in the external C.F. file, AXFILE, with ICURSF the pointer to the surface last processed by INEX. If they are the same control is returned to the calling routine. Otherwise, for each patch of the surface, a mesh of points and their derivatives, currently set at 5*5, are evaluated by calling EVALSS and the principal curvatures computed at each point by CURCAL. The maximum and minimum values of the principal curvatures for the patch are determined and stored in CURMAX and CURMIN. Based on these values, the distances between the corners of the patch and the tolerance, the limits for the DU and DV step out are

computed and stored in DULIM and DVLIM.

Then the routines SHELLA and SHELLB are called to compute the centres and radii of the spherical convex 'hulls' that bound the surface patch, (upto six spheres per patch may be computed). They are stored in CXHULL.

The spheres that may be computed are:

a) Minimum enclosing sphere
b) Up to 4 "edge" spheres
c) The "lid" sphere

Notes: NONE

Referenced by: SMIL

Subsidiaries: CURCAL
EVALSS
SHELLA
SHELLB

6.1.20 INTCHK

Language: FORTRAN 77

Type: Subroutine

Purpose: To perform interference checking.

Calling sequence:

```
CALL INTCHK(PSURF,IPSIZE,PSIDE,PTHICK,TOLIN,TUL,
+                  TE,TA,R,NPAT,UP,VP,LRET,INTLOC)
```

Arguments:

Variable	Type	Description
PSURF	D.P.	Canonical form of part surface (ssurf)
IPSIZE	D.P.	Size of canonical form
PSIDE	D.P.	1 or -1 to adjust surface normal to tool side of surface
PTHICK	D.P.	Thickness offset from surface
TOLIN	D.P.	Tolerance (PS INTOL)
TUL	D.P.	Tool geometry
TE	D.P.	Tool end coordinates
TA	D.P.	Tool axis vector
R	D.P.	Surface contact point and derivatives
NPAT	INTEGER	Patch number for contact point
UP	D.P.	U parameter for contact point
VP	D.P.	V parameter for contact point
LRET	LOGICAL	Return flag = .TRUE. if interference occurs = .FALSE. if no interference
INTLOC	INTEGER	Location in interference table of largest interference

Global Variables:

Variable	Block	Type	Description
IBUG	IBUGG	INTEGER	Flag that forces debug printout
DINTR	INTERF	D.P.	Penetration distance
UINTR	INTERF	D.P.	U parameter of surface point at penetration
VINTR	INTERF	D.P.	V parameter of surface point at penetration
TPINTR	INTERF	D.P.	Coordinates of point on tool at penetration
SPINTR	INTERF	D.P.	Coordinates of point on surface at penetration
SNINTR	INTERF	D.P.	Surface normal at penetration
PINTR	INTERF	INTEGER	Patch number for surface point at penetration
ISEG	INTERF	INTEGER	Tool segment for penetration
NPINTR	INTERF	INTEGER	Number of interference points detected

Local Variables:

Variable	Type	Description
LSUSP	INTEGER	List of suspect patches
NSUSP	INTEGER	Number of suspect patches
MSG	C*120	Print buffer
N	INTEGER	= -NSUSP (for use as argument to BAD)

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

The interference checking is performed in the following manner. First test remote (non-current) patches for possible interference by calling REMSCN, then call REMINT to check out all suspect patches for actual interference and store any penetration data in the interference table. Next CURINT is called to check the current patch (i.e. the patch that is being machined) for interference and store any penetration data in the interference table.

If any interference is detected, the largest penetration is found and its location in the interference table saved in INTLOC.

Notes: NONE

Referenced by: AVOID
SSPATH

Subsidiaries: CPRINT
CURINT
REMINT
REMSCN

6.1.21 INTMSG

Language: FORTRAN 77

Type: Subroutine

Purpose: To print message and relevant data if interference has been detected.

Calling sequence:

```
CALL INTMSG(TE,TA,INTLOC)
```

Arguments:

Variable	Type	Description
TE	D.P.	Tool end coordinates
TA	D.P.	Tool axis vector
INTLOC	INTEGER	Location in interference table of worst penetration

Global Variables:

Variable	Block	Type	Description
DINTR	INTERF	D.P.	Penetration distance
UINTR	INTERF	D.P.	U parameter of surface point at penetration
VINTR	INTERF	D.P.	V parameter of surface point at penetration
PINTR	INTERF	INTEGER	Patch number for surface point at penetration

Local Variables:

Variable	Type	Description
MSG	C*120	PRINT BUFFER

Files:

Restrictions: NONE

Diagnostics: NONE

Method:

The message

```
"INTERFERENCE DETECTED"
"TE = (tool end coords) TA = (tool axis coords)"
"WORST PENETRATION = (penetration) AT PATCH = (patch no) U = u V = v "
```

is printed, giving the details of the worst penetration and the patch number, u and v parameters of the surface point where it occurs for the

current tool location,TE and tool axis TA.

Notes: NONE

Referenced by: AVOID
SSPATH

Subsidiaries: CFORM
CPRINT
FCONV
ICONV

6.1.22 JMPLT2

Language: FORTRAN 77

Type: Subroutine

Purpose: To limit the jump DU,DV to keep U,V in range 0-1, the direction of limited jump is kept the same as original, unless either U or V are already at one of the limits in which case U, V remain unchanged. Flag IEDGE indicates if U or V has been limited. (= 0 not limited, = 1 limited).

Calling sequence:

```
CALL JMPLT2(U,V,DU,DV,IEDGE)
```

Arguments:

Variable	Type	Description
U	D.P.	Current U parameter
V	D.P.	Current V parameter
DU	D.P.	Step out in U direction
DV	D.P.	Step out in V direction
IEDGE	INTEGER	Flag indicating if U or V limited

Global Variables:

Variable	Block	Type	Description
Z0	ZNUMBR	D.P.	Value 0.D0
Z1	ZNUMBR	D.P.	Value 1.D0
Z1EM9	ZNUMBR	D.P.	Value 1.D-9

Local Variables:

Variable	Type	Description
C	D.P.	Work variable

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

The incremental steps in U and V (DU and DV) both checked and if necessary limited to keep the value of the parameters with the current patch, i.e. in range 0 <= u,v <= 1. The following tests are applied.

If the absolute value of step is less that 1.D-9, or the value of the parameter is already within 1.D-9 of 0 or 1 and the step is greater than zero, then the step is set to zero.

If the new value of a parameter after applying the increment would fall outside the patch boundaries, then the increment that would just bring the parameter onto the appropriate boundary is computed and the other parametric increment is scaled correspondingly.

In either case, IEDGE set equal to 1 to indicate that the stepout has been limited. If the increments are not changed the IEDGE is returned as zero.

Notes: NONE

Referenced by: CURINT
MIXAL

Subsidiaries: NONE

6.1.23 MIXAL

Language: FORTRAN 77

Type: Subroutine

Purpose: To search for interference using mixed algorithms.

Calling sequence:

```
 CALL MIXAL(NPATCH,ISIZE,GEO,PSIDE,PTHICK,TUL,
1       TOOLCE,TA,U,V,DUMIN,DVMIN,UPEN,VPEN,PPOINT,TPOINT,
1       PNORM,PENMAX,IPEN,IFLAG,IEDGE,NITERA)
```

Arguments:

Variable	Type	Description
NPATCH	INTEGER	Patch Number
ISIZE	INTEGER	Size of canonical form
GEO	D.P.	Canonical form of surface
PSIDE	D.P.	Tool side of surface
PTHICK	D.P.	Thick allowance required
TUL	D.P.	Tool geometry
TOOLCE	D.P.	Tool centre
TA	D.P.	Tool axis
U	D.P.	U parameter estimated penetration point
V	D.P.	V parameter estimated penetration point
DUMIN	D.P.	Minimum DU indicating convergence
DVMIN	D.P.	Minimum DV indicating convergence
UPEN	D.P.	U parameter of maximum penetration
VPEN	D.P.	V parameter of maximum penetration
PPOINT	D.P.	Penetration point May not be final point in iteration, but is for true (evaluated, not extrapolated) surface point
TPOINT	D.P.	Point on tool corresponding to PPOINT
PNORM	D.P.	Common normal at PPOINT and TPOINT
PENMAX	D.P.	Maximum penetration (+ve for hit)
IPEN	INTEGER	Indicates cutter segment where penetration occurs. = 0 Convergence on bottom flat of corner radius cutter = 1 Convergence on sphere for ballended cutter, on corner radius for corner radius cutter = 2 Convergence on shank
IFLAG	INTEGER	=-1 No convergence = 0 Convergence
IEDGE	INTEGER	= 0 Last DU, DV not limited = 1 Last DU,DV limited
NITERA	INTEGER	Number of iterations

Global Variables:

Variable	Block	Type	Description
IBUG	IBUGG	INTEGER	Flag that forces debug printout

Local Variables:

Variable	Type	Description
CA	D.P.	Cosine of angle between tool and surface normals
DIFCA	D.P.	1.D0 - CA
DU	D.P.	Increment in U
DU1	D.P.	Increment in U estimated by ALGO1
DU2	D.P.	Increment in U estimated by ALGO2
DV	D.P.	Increment in V
DV1	D.P.	Increment in V estimated by ALGO1
DV2	D.P.	Increment in V estimated by ALGO2
I	INTEGER	Loop count
IALGO	INTEGER	Flag forcing algorithm 2
IEDGP	INTEGER	Previous value of IEDGE
IEXT	INTEGER	Index to arrays for storing the results of the last 4 iterations when no convergence has been achieved after NITLIM iterations
IT	INTEGER	Index of worst case of these last four iterations
MODE	INTEGER	Evaluation mode
NORMOK	INTEGER	Flag to indicate surface normal computed satifactorily
R	D.P.	Coordinates of point
RU	D.P.	First derivative wrt U at R
RV	D.P.	First derivative wrt V at R
RUU	D.P.	)
RUV	D.P.	) Second derivatives at R
RVV	D.P.	)
RN	D.P.	Unit surface normal at R
P	D.P.	Point on tool at penetration
T	D.P.	Point on tool with rool normal corresponding to RN
PN	D.P.	Tool normal at P
V1	D.P.	)
V2	D.P.	) Work variables
V3	D.P.	)
V4	D.P.	)
LBASE	LOGICAL	Flag indicating that T is on base of tool
IOK	LOGICAL	Flag indicating if ALGO2 as able to compute DU and DV satisfactorily
SP	D.P.	(PPOINT))
TP	D.P.	(TPOINT))
TN	D.P.	(PNORM)) Storage for data from
UP	D.P.	(UPEN)) last four iterations
VP	D.P.	(VPEN)) if convergence not achieved
PEN	D.P.	(PENMAX)) by NITLIM

JPEN	INTEGER	(IPEN))
IEDG	INTEGER	(IEDGE))
MSG	C*120	Print buffer

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Given initial estimates of U, V on patch NPATCH, MIXAL calls EVALSS to evaluate the surface point R, together with its derivatives and normal. Then PENCUT is called to calculate the penetration of point R into the cutter, PENMAX (mimimum distance if no penetration), in addition it calculates the associated point P on the tool, and the tool normal PN at P. Next CALCT is called to calculate the point T on tool with same normal as the surface at R. If T is on the flat base of the cutter then algorithm 2 is forced by setting IALGO=2.

MIXAL then proceeds to determine values of DU,DV to give a better estimate of the penetration point, this is done by using values computed by both ALGO1 (DU1,DV1) and ALGO2 (DU2,DV2) whenever possible, otherwise results of either ALGO1 or ALGO2 are used as appropriate. The following formula are used to estimate DU and DV based on the results of the two algorithms.

$$DU = \frac{DU1*DU2}{DU1+DU2} \qquad DV = \frac{DV1*DV2}{DV1+DV2}$$

JMPLT2 is then called to limit the values of DU and DV to keep U and V within the current patch.

The resultant values of DU and DV are compared with the limiting values DUMIN and DVMIN in order to test for convergence. If no convergence has been achieved after NITLIM iterations, three more iterations are performed and the results interrogated to determine the worst penetration detected.

On convergence the surface point, tool point and normal in addition to the penetration and the U,V parameters of the surface point are returned to the calling routine. If there is no convergence then the worst case results are returned. IFLAG set to -1 if penetration has been detected without convergence otherwise it is set zero. Also if penetration has been detected without convergence a warning message is output. If required a debug print of the results of the last four iterations can be forced by setting IBUG=11 (MAXDP/-4,11).

Notes: NONE

Referenced by: CURINT
REMINT

Subsidiaries: ALGO1
ALGO2
BAD
CALCT
CFORM
CPRINT
DOTF
EVALSS
FCONV
JMPLT2
PENCUT

6.1.24 NORMA

Language: FORTRAN 77

Type: Subroutine

Purpose: Normalize vector and compute its modulus.

Calling sequence:

```
CALL NORMA(X,XNORM,XMOD)
```

Arguments:

Variable	Type	Description
X	D.P.	Vector to be normalized
XNORM	D.P.	Normalized vector
XMOD	D.P.	Modulus of vector

Global Variables: NONE

Variable	Block	Type	Description

Local Variables: NONE

Variable	Type	Description

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Given 3-D vector X, calculate the normalised (unit) vector XNORM, and modulus of X as XMOD.

Notes: NONE

Referenced by: CALCT
PENCUT

Subsidiaries: NONE

6.1.25 PENCUT

Language: FORTRAN 77

Type: Subroutine

Purpose: Calculate the penetration of point R into cutter.

Calling sequence:

```
 CALL PENCUT(R,RN,TOOLCE,TOOLAX,TUL,PSIDE,PTHICK,
1      T,TN,PEN,ISEG)
```

Arguments:

Variable	Type	Description
R	D.P.	Surface point
RN	D.P.	Surface normal
TOOLCE	D.P.	Tool centre
TOOLAX	D.P.	Tool axis
TUL	D.P.	Cutter geometry
PSIDE	D.P.	Tool side of surface
PTHICK	D.P.	Part surface thickness
T	D.P.	Nearest point on tool to R
TN	D.P.	Tool normal at T
PEN	D.P.	Penetration: Distance R-T (+ve for hit)
ISEG	D.P.	Indicates tool segment where penetration occurs. = 0 T on:- Bottom flat = 1 Ball or corner radius = 2 Cylindrical shank

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
A	D.P.	) Work variables
B	D.P.	)
K	INTEGER	Loop count
S	D.P.	Sign for surface side of tool
V	D.P.	)
V1	D.P.	) Work variables
W	D.P.	)
X	D.P.	Length of vector
XR	D.P.	Corner radius center associated with T

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

PENCUT first calculates the height,A of point $\underline{R}$ above tool center

$$A = (\underline{R} - \underline{TOOLCE}) \cdot \underline{TA}$$

Then calculates the radial direction,W and distance,B of $\underline{R}$ from tool axis.

$$\underline{W} = \frac{(\underline{R} - \underline{TOOLCE}) - A * \underline{TA}}{B}$$

where

$$B = | (\underline{R} - \underline{TOOLCE}) - A * \underline{TA} |$$

The surface side S of the tool is determined, then PENCUT computes associated point, tool normal and penetration which will depend on relative location of surface point to tool center.

If A is greater than or equal to zero, $\underline{R}$ above the tool center, then the associated point is on the cylindrical section of the tool. There are two cases to be considered

a) if $\underline{R}$ is further from the tool axis than the corner radius center (B >= e)

$$p = d/2 + S*B + PTHICK$$

$$\underline{TN} = S*\underline{W}$$

$$\underline{T} = \underline{TOOLCE} - d/2*\underline{TN} + A*\underline{TA}$$

b) if $\underline{R}$ is within the 'inner' cylinder (B < e).

$$\underline{XR} = \underline{TOOLCE} - S*e*\underline{W}$$ Center of the appropriate corner radius

$$\underline{TN} = \frac{(\underline{R} - \underline{XR})}{|(\underline{R} - \underline{XR})|}$$

$$p = |(\underline{R} - \underline{XR})| + r + PTHICK$$

$$\underline{T} = \underline{XR} - r*\underline{TN}$$

Otherwise, the associated point $\underline{T}$ is on the corner radius or base of the cutter.

If $\underline{T}$ is on the flat base of a corner radius cutter (B <= e) then,

$$p = f + A + PTHICK$$

$$\underline{TN} = \underline{TA}$$

$$\underline{T} = \underline{TOOLCE} + B*\underline{W} - f*\underline{TA}$$

If T is on the corner radius (B > e) then,

$$\underline{XR} = \underline{TOOLCE} + e*\underline{W}$$

$$\underline{TN} = \frac{(\underline{XR} - \underline{R})}{|(\underline{XR} - \underline{R})|}$$

$$p = r - |(\underline{XR} - \underline{R})| + PTHICK$$

$$\underline{T} = \underline{XR} - r*\underline{TN}$$

The flag ISEG is set according to which segment of the tool T lies on.

ISEG	=	0	T on:- Bottom flat
	=	1	Ball or corner radius
	=	2	Cylindrical shank

Notes: NONE

Referenced by: MIXAL

Subsidiaries: DOTF
NORMA

6.1.26 REMINT

Language: FORTRAN 77

Type: Subroutine

Purpose: To check out suspect remote patches and if any penetration is detected store the worst penetration for each patch in the interference table.

Calling sequence:

```
 CALL REMINT(TE,TA,TUL,PSURF,IPSIZE,PSIDE,PTHICK,TOLIN,
+                    LSUSP,NSUSP)
```

Arguments:

Variable	Type	Description
TE	D.P.	Tool end coordinates
TA	D.P.	Tool axis vector
TUL	D.P.	Tool geometry
PSURF	D.P.	Canonical form of part surface (SSURF)
IPSIZE	INTEGER	Size of canonical form
PSIDE	D.P.	1 or -1 to adjust surface normal to tool side of surface
PTHICK	D.P.	Thickness offset from surface
TOLIN	D.P.	Tolerance (PS INTOL)
LSUSP	INTEGER	List of suspect patches
NSUSP	INTEGER	Number of suspect patches

Global Variables:

Variable	Block	Type	Description
IBUG	IBUGG	INTEGER	Flag that forces debug printout
DULIM	INIDAT	D.P.	DU iteration limit for patches
DVLIM	INIDAT	D.P.	DV iteration limit for patches
DINTR	INTERF	D.P.	Penetration distance
UINTR	INTERF	D.P.	U parameter of surface point at penetration
VINTR	INTERF	D.P.	V parameter of surface point at penetration
LUINTR	INTERF	D.P.	U parameter of surface point at previous penetration
LVINTR	INTERF	D.P.	V parameter of surface point at previous penetration
TPINTR	INTERF	D.P.	Coordinates of point on tool at penetration
SPINTR	INTERF	D.P.	Coordinates of point on surface at penetration
SNINTR	INTERF	D.P.	Surface normal at penetration
PINTR	INTERF	INTEGER	Patch number for surface point at penetration
LPINTR	INTERF	INTEGER	Patch number for surface point at

			previous penetration
ISEG	INTERF	INTEGER	Tool segment for penetration
NPINTR	INTERF	INTEGER	Number of interference points detected
LNPINT	INTERF	INTEGER	Previous number of interference points detected

Local Variables:

Variable	Type	Description
NPAT	INTEGER	Patch number
IPEN	INTEGER	Cutter segment where penetration occurs
IFLAG	INTEGER	Convergence flag
IEDGE	INTEGER	Flag indicating if last DU or DV limited
NITERA	INTEGER	No. of iterations
TOOLCE	D.P.	Tool center
UPEN	D.P.	U parameter at penetration
VPEN	D.P.	V parameter at penetration
PPOINT	D.P.	Penetration point
TPOINT	D.P.	Point on tool corresponding to penetration point
PNORM	D.P.	Common normal at PPOINT and TPOINT
PENMAX	D.P.	Maximium penetration
USTART	D.P.	U parameter for start of search
VSTART	D.P.	V parameter for start of search

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

REMINT first calculates the 'tool center' for simple ball-ended and corner radius cutters.

$$\underline{TOOLCE} = \underline{TE} + f*\underline{TA}$$

Then for each suspect patch if interference was detected with the patch at the previous tool position, the U and V parameters of that penetration point are taken as the start values for U and V, otherwise U=0.5 and V=0.5 are used.

MIXAL is then called to search this patch for interference.
If interference is detected the details are stored in the interference table.

Notes: NONE

Referenced by: INTCHK

Subsidiaries: BAD
MIXAL

6.1.27 REMSCN

Language: FORTRAN 77

Type: Subroutine

Purpose: To test all remote (non-current) patches for possible interference.

Calling sequence:

```
CALL REMSCN(TE,TA,TUL,R,PSIDE,PTHICK,TOLIN,NPAT,LSUSP,NSUSP)
```

Arguments:

Variable	Type	Description
TE	D.P.	Tool end coordinates
TA	D.P.	Tool axis vector
TUL	D.P.	Tool geometry
R	D.P.	Surface contact point and derivatives
PSIDE	D.P.	- 1 or -1 to adjust surface normal to tool side of surface
PTHICK	D.P.	Thickness offset from surface
TOLIN	D.P.	Tolerance (PS INTOL)
NPAT	INTEGER	Patch number for contact point
LSUSP	INTEGER	Suspect patches
NSUSP	INTEGER	Number of suspect patches

Global Variables:

Variable	Block	Type	Description
CXHULL	INIDAT	D.P.	Spherical surface enclosures (hulls)
DULIM	INIDAT	D.P.	DU iteration limit for patches
DVLIM	INIDAT	D.P.	DV iteration limit for patches
CURMAX	INIDAT	D.P.	Maximum curvature of patches
CURMIN	INIDAT	D.P.	Minimum curvature of patches
NCXHUL	INIDAT	INTEGER	Number of hulls
NGP	INIDAT	INTEGER	Number of patches in current surface
ICURSF	INIDAT	INTEGER	External LDA file record number for current surface

Local Variables:

Variable	Type	Description
TCCENT	D.P.	Centre of tool contact sphere
TCRAD	D.P.	Radius of tool contact sphere
TLCENT	D.P.	Centre of large shallow sphere through TE, radius BIG5
TCENT	D.P.	Centre of tight sphere enveloping tool, radius=TUL(2)
BALL	LOGICAL	Cutter type flag = .TRUE. if cutter is ballended = .FALSE. otherwise

ISEG	INTEGER	Indicates segment of tool that penetrates = -1 if no penetration = 0 bottom flat = 1 corner radius = 2 cylinder
V1	D.P.	Vector from tool tip to centre of patch sphere
V2	D.P.	V1 X TA
D	D.P.	Minimum distance between centre of patch sphere and tool axis
D2	D.P.	Square of distance between two sphere centres
A	D.P.	Minimum distance between sphere centres to avoid interference

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

REMSCN first calls TCSPHR to calculate the tool contact sphere for the current tool position. The REMSCN calculates the centre of the large shallow sphere through TE, TLCENT and centre of tight sphere enveloping tool TCENT. Next the flag indicating cutter type is set BALL=.TRUE. for ballended cutter otherwise BALL=.FALSE. .

REMSCN then tests the non-current patches for possible interference using the following schedule.

1) Test for interference of infinite tool cylinder with minimum patch sphere.
2) If there is still possible interference with this patch, test for interference of tight tool sphere with all patch convex hull spheres.
3) If there is still possible interference with this patch, test for interference of shallow tool sphere with all patch convex hull spheres.
4) If there is still possible interference with this patch, test for interference of tool contact sphere with all patch convex hull spheres.
5) If there is still possible interference with this patch, HULCUT is called to test for interference of actual tool with all patch convex hull spheres unless cutter is ball-ended.

If the patch not cleared then it is added to the list of suspect patches.

Notes: NONE

Referenced by: INTCHK

Subsidiaries: CROSS
DIST2
HULCUT
TCSPHR
VECMOD

6.1.28 RESET

Language: FORTRAN 77

Type: Subroutine

Purpose: To reset interference table for next point.

Calling sequence:

```
CALL RESET
```

Arguments: NONE

Variable	Type	Description

Global Variables:

Variable	Block	Type	Description
DINTR	INTERF	D.P.	Penetration distance
UINTR	INTERF	D.P.	U parameter of surface point at penetration
VINTR	INTERF	D.P.	V parameter of surface point at penetration
LDINTR	INTERF	D.P.	Previous penetration distance
LUINTR	INTERF	D.P.	U parameter of surface point at previous penetration
LVINTR	INTERF	D.P.	V parameter of surface point at previous penetration
TPINTR	INTERF	D.P.	Coordinates of point on tool at penetration
SPINTR	INTERF	D.P.	Coordinates of point on surface at penetration
SNINTR	INTERF	D.P.	Surface normal at penetration
PINTR	INTERF	INTEGER	Patch number for surface point at penetration
LPINTR	INTERF	INTEGER	Patch number for surface point at previous penetration
ISEG	INTERF	INTEGER	Tool segment for penetration
NPINTR	INTERF	INTEGER	Number of interference points detected
LNPINT	INTERF	INTEGER	Previous number of interference points detected

Local Variables:

Variable	Type	Description
I	INTEGER	Loop count

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

The current values of DINTR, PINTR, UINTR and VINTR are saved in LDINTR, LPINTR, LUINTR and LVINTR respectively, and DINTR, PINTR, UINTR and VINTR reset to zero. Also ISEG is set to zero, and TPINTR, SPINTR and SNINTR to zero vectors. Finally NPINTR is saved in LNPINT before reseting to zero.

Notes: NONE

Referenced by: AVOID
SSPATH

Subsidiaries: NONE

6.1.29 RESTOR

Language: FORTRAN 77

Type: Subroutine

Purpose: To restore interference table for use as start values for next inteference check after avoiding interference.

Calling sequence:

```
CALL RESTOR
```

Arguments: NONE

Variable	Type	Description

Global Variables:

Variable	Block	Type	Description
DINTR	INTERF	D.P.	Penetration distance
UINTR	INTERF	D.P.	U parameter of surface point at penetration
VINTR	INTERF	D.P.	V parameter of surface point at penetration
LDINTR	INTERF	D.P.	Previous penetration distance
LUINTR	INTERF	D.P.	U parameter of surface point at previous penetration
LVINTR	INTERF	D.P.	V parameter of surface point at previous penetration
PINTR	INTERF	INTEGER	Patch number for surface point at penetration
LPINTR	INTERF	INTEGER	Patch number for surface point at previous penetration
NPINTR	INTERF	INTEGER	Number of interference points detected
LNPINT	INTERF	INTEGER	Previous number of interference points detected

Local Variables:

Variable	Type	Description
I	INTEGER	Loop count

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

The previous values of DINTR, PINTR, UINTR and VINTR are restored from LDINTR, LPINTR, LUINTR and LVINTR, likewise the number of intersections NPINTR is restored from LNPINT.

Notes: NONE

Referenced by: AVOID

Subsidiaries: NONE

6.1.30 SHELLA

Language: FORTRAN 77

Type: Subroutine

Purpose: To calculate center C and radius R of the smallest sphere which encloses the four points P (the corner points of patch).

Calling sequence:

```
CALL SHELLA(P,PHULL,NP,N3,C,R)
```

Arguments:

Variable	Type	Description
P	D.P.	Four corner points of patch
PHULL	D.P.	Convex polygon hull of patch
NP	INTEGER	First dimension of PHULL
N3	INTEGER	Second dimension of PHULL
C	D.P.	Center of sphere
R	D.P.	Radius of sphere

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
H	D.P.	COEFF. MATRIX FOR CASE 3
WA	D.P.	WORKSPACE FOR CASE 3
CC	D.P.	CENTRE OF TRIAL SPHERE
V1	D.P.	)
V2	D.P.	)
V3	D.P.	) WORK ARRAYS
V4	D.P.	)
V5	D.P.	)
VN	D.P.	)
X1	D.P.	) TEMP FOR TEST ONLY
X2	D.P.	)
TOL	D.P.	TOLERANCE FOR (SET = 1D-3)
HALF	D.P.	VALUE = 1/2

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Calculates center C and radius R of the smallest sphere which encloses the four points P (the corner points of patch).

Three cases are tested in the following order:-

1. One pair of points define diameter of sphere enclosing all four corner points. Choose pair of points with greatest separation, check that other two points are inside this sphere.
2. Three points define equator of sphere enclosing all four points. Try all four combinations of three points, choose smallest sphere enclosing four points.
3. Minimum sphere defined by all four points

Checks that sphere found encloses polygonal convex hull PHULL, adjusts radius of sphere by up to 10% to include PHULL, if no sphere found, then return with R=0.

Notes: NONE

Referenced by: INEX

Subsidiaries: CROSS
DEGSL
SOLVE3

6.1.31 SHELLB

Language: FORTRAN 77

Type: Subroutine

Purpose: To calculates the four "edge" shells for a patch specified by a net of points RNET.

Calling sequence:

```
      CALL SHELLB(NN,N3,N4,RCORN1,RCORN2,CV1,CV2,RNET,
     1     SCENT,SRAD,NFLAGS)
```

Arguments:

Variable	Type	Description
NN	INTEGER	First dimension of RNET
N3	INTEGER	Second dimension of RNET,RCORN1 etc. = 3 (XYZ coords)
N4	INTEGER	Number of corner points
RCORN1	D.P.	) Corner points
RCORN2	D.P.	)
CV1	D.P.	) Corner vectors
CV2	D.P.	)
RNET	D.P.	Net of points
SCENT	D.P.	Sphere centres
SRAD	D.P.	Sphere radii
NFLAGS	INTEGER	Array of flags to indicate if a sphere is calculated. (=1 calculated, =0 otherwise)

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
V1	D.P.	)
V2	D.P.	)
D	D.P.	)
E	D.P.	)
F	D.P.	) Work arrays
AB	D.P.	)
CD	D.P.	)
DE	D.P.	)
EA	D.P.	)
EB	D.P.	)
EF	D.P.	)
JEDGE	INTEGER	Edge of patch

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

For each edge in turn a sphere is calculated that just encloses all points of RNET that define the edge, and with its center lying on the line from the midpoint of the chord between the two end points of the edge and a corresponding point on the opposing edge.

The centers SCENT and radii SRAD are returned. If SHELLB is unable to compute a suitable sphere (e.g. if the edge is degenerate) then a zero radius is returned and the corresponding flag NFLAG set to zero. Note NFLAG is set to 1 whenever a 'edge' sphere is sucessfully computed.

Notes: NONE

Referenced by: INEX

Subsidiaries: DOTF
VECMOD

6.1.32 SOLVE3

Language: FORTRAN 77

Type: Subroutine

Purpose: To solve 3 simultaneous equations $A\underline{X}=\underline{B}$

Calling sequence:

```
CALL SOLVE3(A,B,X,IFLAG)
```

Arguments:

Variable	Type	Description
A	D.P.	Matrix of coefficients
B	D.P.	RHS vector
X	D.P.	Result vector
IFLAG	INTEGER	

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

First the determinant,D of the matrix A is computed. If abs(D)<1.D-8 then A is singular, therefore there is no solution and IFLAG=1 is returned.

Otherwise, $X = A^{-1}B$ is computed.

Notes: NONE

Referenced by: SHELLA

Subsidiaries: NONE

6.1.33 SSDAT3

Language: FORTRAN 77

Type: Block Data Subprogram

Purpose: Initialization of regional milling and tool axis control flags.

Calling sequence: None

Arguments: None

Variable	Type	Description

Global Variables:

Variable	Block	Type	Description
LGCHK	RMFLGS	LOGICAL	Gouge check flag
LINTCK	RMFLGS	LOGICAL	Interference checking flag
LAVCTL	RMFLGS	LOGICAL	Avoidance control flag
LVAXIS	RMFLGS	LOGICAL	Variable tool axis flag
ALPHA	TACPRM	D.P.	Tool axis slew angle wrt SN in guide surface CRSSPL direction if the path is in the TANSPL direction and vice versa.
BETA	TACPRM	D.P.	Tool axis tilt angle (CUTANG) wrt SN in path direction
COSA	TACPRM	D.P.	COS(ALPHA)
COSB	TACPRM	D.P.	COS(BETA)
SINA	TACPRM	D.P.	SIN(ALPHA)
SINB	TACPRM	D.P.	SIN(BETA)
ITAC	TACPRM	INTEGER	Flag indicating type tool axis control required, =1 for ATANGL type of tool axis control =0 for fixed or NORMDS

Local Variables: NONE

Variable	Type	Description

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Initialized by DATA statements.

Notes: NONE

Referenced by: NONE

Subsidiaries: NONE

6.1.34 TCSPHR

Language: FORTRAN 77

Type: Subroutine

Purpose: Compute the centre and radius of the tool contact sphere.

Calling sequence:

```
CALL TCSPHR(TE,TA,TUL,R,PSIDE,PTHICK,TCCENT,TCRAD)
```

Arguments:

Variable	Type	Description
TE	D.P.	Tool end coordinates
TA	D.P.	Tool axis vector
TUL	D.P.	Tool geometry
R	D.P.	Surface contact point and derivatives
PSIDE	D.P.	1 or -1 to adjust surface normal to tool side of surface (RU X RV)
PTHICK	D.P.	Thickness offset from surface
TCCENT	D.P.	Centre of tool contact sphere
TCRAD	D.P.	Radius of tool contact sphere

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
COSA	D.P.	Cosine of angle between surface normal and tool axis
SINA	D.P.	Sine of angle between surface normal and tool axis
OFFSET	D.P.	Position of centre along TA from TE
I	INTEGER	Loop count

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

TCSPHR first calculates the cosine of the angle between the surface normal and the tool axis, COSA = SN.TA

If SN is parallel to TA then the radius of the tool contact sphere, TCRAD is taken as 1.D5 and the tool center TCCENT = TE + TCRAD*TA .

Otherwise, the sine of the angle between SN and TA is computed (SINA)

and the radius of the tool contact sphere computed,

TCRAD = e/SINA + r where e is corner radius center offset from TA
r is corner radius

The position of the centre of the tool contact sphere along the tool axis, TA from the tool tip TE is computed,

OFFSET = f + (TCRAD-r)*COSA

Therefore,

TCCENT = TE + OFFSET*TA

Notes: NONE

Referenced by: AVOID
REMSCN

Subsidiaries: DOTF

6.1.35 TECALC

Language: FORTRAN 77

Type: Subroutine

Purpose: Compute TE given current contact point R and tool axis TA.

Calling sequence:

```
CALL TECALC(R,TA,U,PSIDE,PTHICK,TOLCOM,TUL,TE)
```

Arguments:

Variable	Type	Description
R	D.P.	Input surface contact point/derivatives (see CNSURF)
TA	D.P.	Input tool axis
U	D.P.	Local forward direction
PSIDE	D.P.	Input 1 or -1, defines cutter side of surface
PTHICK	D.P.	Part geometry thick specification
TOLCOM	D.P.	Tolerance band compensation
TUL	D.P.	Cutter parameters
TE	D.P.	Output tool end

Global Variables:

Variable	Block	Type	Description
TLHITE	TLLDAT	D.P.	Axial coordinate of cutter segment reference point
TANHI	TLLDAT	D.P.	Minimum slope for cutter segment TN
TANLO	TLLDAT	D.P.	Maximum slope for cutter segment TN
RI	TLLDAT	D.P.	Radial coordinate of cutter segment reference point
CORRAD	TLLDAT	D.P.	Radius of cutter arc segments
TCONT	TLLDAT	D.P.	Index to cutter segment - indicates if segment is present or not
DARRAY	DARRAY	C*120	Print Buffer

Local Variables:

Variable	Type	Description
OFFSET	D.P.	Tool offset from surface
VE	D.P.	Tool radial axis
VLEN2	D.P.	Length of VEC squared
V	D.P.	)
VN	D.P.	) Work vectors
CANG	D.P.	Cos(angle between tool normal and tool radial axes)
SANG	D.P.	Sine of same angle
TANG	D.P.	Tangent of same angle
BADMSG	D.P.	Diagnostic message

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

TECALC initially calculates the general case of the tool offset from the surface,

```
OFFSET=PSIDE*(PTHICK+TOLCOM)
```

Then, the local tool radial axis is determined, VEC = (TA^SN)^TA.
If TA is parallel to SN then U (the local forward direction) is used to determine tool radial axis, VEC = (TA^U)^TA.
VEC is adjusted to take account of the tool side of the surface (PSIDE).

Next TECALC computes the tangent of the angle between tool normal and tool radial axis, TANG.

This is checked against TANHI and TANLO to determine which segment of the cutter the angle is valid for.

If the angle is valid for segments 3,4,5 of cutter i.e. corner radius and adjacent point circle segments, in which case the tool end coordinates are determined using the relevant tool segment data from the TLLDAT block.

```
TE = SP + (OFFSET + PSIDE*CORRAD(i))*SN
       + RI(i)*VEC - TLHITE(i)*TA
```

If the angle is valid for the tool tip segment then TECALC will output a warning and return

```
TE = SP + OFFSET*SN
```

If the angle is valid for the top of the upper line segment (i=7) then TECALC will output a warning and return

```
TE = SP + OFFSET*SN + RI(7)*VEC -TLHITE(7)*TA
```

Otherwise the angle is an invalid angle for this cutter, in which case TECALC will output a warning and not compute a new TE.

Notes: NONE

Referenced by: AVOID

Subsidiaries: BAD
CFORM
CPRINT

CROSS
CROSSV
DOTV
VNORM

6.1.36 TXHULL

Language: FORTRAN 77

Type: Subroutine

Purpose: To compute the center and radius of the spherical 'hull' that encloses the tool.

Calling sequence:

```
      CALL TXHULL(RC,RCNORM,R1,R2,R3,TLEN,TOOLAX,TOOLTP,
     1            TOOLCE,CT,HRAD,HCENT)
```

Arguments:

Variable	Type	Description
RC	D.P.	Point of contact
RCNORM	D.P.	Surface normal at point of contact
R1	D.P.	Half diameter of cutter
R2	D.P.	Corner radius
R3	D.P.	Offset of corner readius center from tool axis
TLEN	D.P.	Tool length
TOOLAX	D.P.	Tool axis
TOOLTP	D.P.	Tool end (not used)
TOOLCE	D.P.	Tool center
CT	D.P.	Curvature of enclosing sphere
HRAD	D.P.	Radius of spherical 'hull'
HCENT	D.P.	Centre of spherical 'hull'

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
A	D.P.	)
B	D.P.	) Work Variables
C	D.P.	)
I	INTEGER	Loop count
RA	D.P.	Radial distance of point of contact from tool axis
RR1	D.P.	Work variable = R1 squared
TR	D.P.	Maximum radius of curvature of tool at R
W1	D.P.	Vector from contact point to tool center

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

TXHULL first calculates the radial distance RA of the contact point from the tool axis. If RA if greater than the tool radius or RA is less that the offset of the corner radius from the tool axis, i.e. if the point of contact is on the cylindrical side of the tool or the end flat, then the tool sphere radius is taken as 1.D5.

Otherwise the tool sphere radius, HRAD is calculated as half the distance between the contact point R and a point on top of tool opposite to R. If HRAD is greater than 1.D5 it is limited to 1.D5.

If HRAD is not limited, then the maximum radius of curvature of the tool at R, TR is calculated. If this exceeds 1.D5 then the tool sphere radius is taken as 1.D5. Otherwise it is taken as the maximum of HRAD and TR.

Finally, the curvature of the tool CT and the center of the tool sphere HCENT are calculated. HCENT is taken to lie on the surface normal from the point of contact.

$$\underline{HCENT} = \underline{RC} + HRAD*\underline{RCNORM}$$

Notes: NONE

Referenced by: CURINT

Subsidiaries: DOTF
VECMOD

6.1.37 VECMOD

Language: FORTRAN 77

Type: Subroutine

Purpose: To calculates modulus XMOD of vector X.

Calling sequence:

```
CALL VECMOD(X,XMOD)
```

Arguments:

Variable	Type	Description
X	D.P.	VECTOR
XMOD	D.P.	LENGTH OF VECTOR (MODULUS)

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Given 3-d vector X, calculates the length of the vector XMOD

Notes: NONE

Referenced by: ALGO1
CALCT
HULCUT
NORMA
REMSCN
SHELLA
TXHULL

Subsidiaries: NONE

6.2 Module GOUGECHK

6.2.1 DEVCAL

Language: FORTRAN 77

Type: Subroutine

Purpose: Calculate the signed deviation of a point Q from the chord joining two points P0 and P1. The sign of the deviation being determined by the direction of TN.

Calling sequence:

```
CALL DEVCAL(Q,TN,P0,P1,DEV)
```

Arguments:

Variable	Type	Description
Q	D.P.	Point whose deviation from chord is required
TN	D.P.	Vector indicating direction of positive deviation
P0	D.P.	Point at one end of chord
P1	D.P.	Point at other end of chord
DEV	D.P.	Signed deviation of Q from chord P0P1

Global Variables: NONE

Variable	Block	Type	Description

Local Variables:

Variable	Type	Description
BADMSG	C*20	DIAGNOSTIC MESSAGE

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Consider,

$\underline{r}(u) = \underline{P0} + u(\underline{P1}-\underline{P0})$ Parametric equation of the chord

$\underline{H}(u) = \underline{Q} - \underline{r}(u)$ Equation of vector from point $\underline{r}(u)$ on the chord to $\underline{Q}$

$h(u) = \underline{H}(u).\underline{H}(u)$ Length of vector $\underline{H}$

$= (\underline{Q} - \underline{r}(u)).(\underline{Q} - \underline{r}(u))$

By substituting for $\underline{r}(u)$ and differentiating wrt u it can be shown that the minimum value for h(u) occurs when

$$u = \frac{(\underline{P1}-\underline{P0}).(\underline{Q}-\underline{P0})}{(\underline{P1}-\underline{P0}).(\underline{P1}-\underline{P0})}$$

The sign of the deviation is determined by considering $\underline{H}(u).\underline{TN}$ and the magnitude of the deviation is the length of the minimum value of $\underline{H}(u)$.

Notes: NONE

Referenced by: GCHK

Subsidiaries: BAD
CPRINT
DOTF

6.2.2 GCHK

Language: FORTRAN 77

Type: Subroutine

Purpose: To perform GOUGCK gouge checking for the move from the previously accepted tool position to the current tool position. If gouging is detected then a suitable step reduction factor to avoid gouging is returned.

Calling sequence:

```
 CALL GCHK(SURF,SIDE,R ,TE ,TA ,NPAT ,UP ,VP ,
+          RL,TEL,TAL,NPATL,UPL,VPL,
+          TOLCOM,TOLI,TOLO,GFAC,IRET)
```

Arguments:

Variable	Type	Description
SURF	D.P.	CANONICAL FORM OF PS
SIDE	D.P.	1 OR -1 TO INDICATE RELATIONSHIP OF SURFACE NORMAL TO TOOL SIDE OF SURFACE
R	D.P.	CURRENT CONTACT POINT
TE	D.P.	CURRENT TOOL END
TA	D.P.	CURRENT TOOL AXIS
NPAT	INTEGER	PATCH NUMBER)
UP	D.P.	U PARAMETER) OF CURRENT CONTACT POINT
VP	D.P.	V PARAMETER)
RL	D.P.	PREVIOUS CONTACT POINT
TEL	D.P.	PREVIOUS TOOL END POSITION
TAL	D.P.	PREVIOUS TOOL AXIS
NPATL	INTEGER	PATCH NUMBER)
UPL	D.P.	U PARAMETER) OF PREVIOUS CONTACT POINT
VPL	D.P.	V PARAMETER)
TOLCOM	D.P.	TOLERANCE COMPENSATION VALUE
TOLI	D.P.	PS INTOL
TOLO	D.P.	PS OUTTOL
GFAC	D.P.	STEP REDUCTION FACTOR
IRET	INTEGER	RETURN FLAG

Global Variables:

Variable	Block	Type	Description
IBUG	IBUGG	INTEGER	Flag that forces debug printout

Local Variables:

Variable	Type	Description
BADMSG	C*20	Diagnostic message
DEV	D.P.	Deviations of parametric midpoint from the various chords
DEVMAX	D.P.	Maximum deviation

DEVMIN	D.P.	Minimum deviation
I	INTEGER	Loop count
OFFSET	D.P.	Offset of tool contact point from surface
ONE	D.P.	1.0D0
RM	D.P.	Parametric midpoint and its derivatives
SKRM	D.P.	Curvature of surface at parametric midpoint in direction of chord between previous and current contact points
TN	D.P.	Tool normal at parametric midpoint
TP	D.P.	Current tool contact point
TPL	D.P.	Previous tool contact point
XTP	D.P.	Point on tool at current position corresponding to previous tool contact point
XTPL	D.P.	Point on tool at previous position corresponding to current tool contact point
Z1EM10	D.P.	1.0D-10
ZERO	D.P.	0.0D0

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

GCHK first call MIDPNT to compute the coordinates and derivatives of the parametric midpoint on the surface between the current and previous surface points at which the tool contacted the surface. Then MIDKUR is called to compute the surface curvature at this midpoint in chord direction. GCHK then computes the tool contact point at the current and last tool locations and tool normal at midpoint. Note the tool contact point and surface contact point differ by an offset distance equal to the tolerance compensation value. This is because the cut vector is intended to be tangential to the surface between the two endpoints.

The next step is to compute the two tool points, XTP and XTPL that correspond to the tool contact point at the alternative tool location. Then GCHK calls DEVCAL to compute

a) the deviation of the midpoint RM from the chord between the two tool contact points TP and TPL
b) the deviations of the midpoint RM from the chord between the tool contact point and corresponding point for the previous location for both tool locations. (TP to XTPL and TPL to XTP)

The maximum and minimum deviations (DEVMAX, DEVMIN) are then determined.

GHCK the checks to see if the appropriate tolerance band violated.

If the surface curvature in the chord direction at the midpoint SKRM is positive (surface concave) and the absolute value of DEVMIN is greater

than TOLO then excess material will be left on, therefore a step reduction factor is computed equal to the minimum of TOLO/abs(DEVMIN) and 0.95, unless TOLO is less than 1.D-10 (OUTTOL=0) in which case a step reduction factor of 0.75 is returned. The flag IRET is set equal to -1 to indicate that excess material is left on.

If the surface curvature in the chord direction at the midpoint SKRM is negative (surface convex) and the maximum deviation DEVMAX is greater than TOLI then gouging of surface will occur, therefore a step reduction factor is computed equal to the minimum of TOLI/abs(DEVMAX) and 0.95, unless TOLI is less than 1.D-10 (INTOL=0) in which case a step reduction factor of 0.75 is returned. The flag IRET is set equal to 1 to indicate that gouging will occur.

Notes: NONE

Referenced by: SSPATH

Subsidiaries: BAD
CPRINT
DEVCAL
MIDKUR
MIDPNT
XTPCAL

6.2.3 MIDKUR

Language: FORTRAN 77

Type: Subroutine

Purpose: To calculate the curvature of the surface at point R in the direction of the chord between the two points P0 and P1.

Calling sequence:

```
CALL MIDKUR(R,P0,P1,SIDE,SK)
```

Arguments:

Variable	Type	Description
R	D.P.	Midpoint contact point and derivatives
P0	D.P.	Last contact point
P1	D.P.	Current contact point
SIDE	D.P.	Tool side of surface
SX	D.P.	Curvature of surface in direction of chord

GLOBAL Variables:

Variable	Block	Type	Description
IBUG	IBUGG	INTEGER	Flag that forces debug printout

Local Variables:

Variable	Type	Description
CH	D.P.	Direction of chord from P0 to P1
VEC	D.P.	Projection of CH onto surface tangent plane
BADMSG	C*20	Diagnostic message

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

MIDKUR calls KURSRF to compute the principle curvatures etc. at the midpoint, R, then computes the chord direction CH and projects this chord vector into surface tangent plane. Finally MIDKUR computes the curvature of surface in the direction of the chord.

Notes: NONE

Referenced by: GCHK

Subsidiaries: BAD
CPRINT
CROSS
DOTF
KURSRF
VNORM

6.2.4 MIDPNT

Language: FORTRAN 77

Type: Subroutine

Purpose: To evaluate the parametric midpoint on the surface between the current and prevoius cutter contact points.

Calling sequence:

```
CALL MIDPNT(SURF,NPAT,UP,VP,NPATL,UPL,VPL,RM)
```

Arguments:

Variable	Type	Description
SURF	D.P.	Canonical form of surface
NPAT	INTEGER	Patch number for current point
UP	D.P.	U parameter for current point
VP	D.P.	V parameter for current point
NPATL	INTEGER	Patch number for previous point
UPL	D.P.	U parameter for previous point
VPL	D.P.	V parameter for previous point
RM	D.P.	Midpoint and its derivatives

Global Variables:

Variable	Block	Type	Description
IBUG	IBUGG	INTEGER	Flag that forces debug printout

Local Variables:

Variable	Type	Description
B	D.P.	Patch coefficients
BADMSG	C*20	Diagnostic message
IFLAG	INTEGER	Patch flags
ITOP	INTEGER	
NPATM	INTEGER	Patch number for midpoint
U	D.P.	Global U parameter for midpoint
U1	D.P.	Global U parameter for current point
U2	D.P.	Global U parameter for previous point
UM	D.P.	U parameter for midpoint
V	D.P.	Global V parameter for midpoint
V1	D.P.	Global V parameter for current point
V2	D.P.	Global V parameter for previous point
VM	D.P.	V parameter for midpoint

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

MIDPNT computes the global parameters for the current and previous points by calling MESCON, from which the global parameters of midpoint are computed. Then MESCON is again called to compute the patch number and local parameters for the midpoint. LODPCH is called to load the patch coefficients and CNSURF to evaluate the coordinates of the midpoint.

Notes: NONE

Referenced by: GCHK

Subsidiaries: CNSURF
CPRINT
LODPCH
MESCON

6.2.5 XTPCAL

Language: FORTRAN 77

Type: Subroutine

Purpose: To compute the point on the tool when positioned at TE corresponding to the point of contact at the last accepted tool position TEL and vice versa.

Calling sequence:

```
CALL XTPCAL(TP,TE,TA,TPL,TEL,TAL,XTP,XTPL)
```

Arguments:

Variable	Type	Description
TP	D.P.	Current tool contact point
TE	D.P.	Current tool end coords.
TA	D.P.	Current tool axis
TPL	D.P.	Previous tool contact point
TEL	D.P.	Previous tool end coords.
TAL	D.P.	Previous tool axis
XTP	D.P.	Point on tool at current position corresponding to tool contact point at previous position
XTPL	D.P.	Point on tool at previous position corresponding to tool contact point at current position

Global Variables:

Variable	Block	Type	Description
IBUG	IBUGG	INTEGER	Flag that forces debug printout

Local Variables:

Variable	Type	Description
FWD	D.P.	Unit forward vector
VN	D.P.	Vector normal to FWD and TA
VNL	D.P.	Vector normal to FWD and TAL
TF	D.P.	Vector orthogonal to VN and TA
TFL	D.P.	Vector orthogonal to VNL and TAL
TPV	D.P.	Vector from tool tip TE to surface contact point TP
TPVL	D.P.	Vector from tool tip TEL to surface contact point TPL
BADMSG	C*20	Diagnostic message
I	INTEGER	Loop count
X	D.P.	Projection of TPV onto TF
Y	D.P.	Projection of TPV onto VN
Z	D.P.	Projection of TPV onto TA
XL	D.P.	Projection of TPVL onto TF
YL	D.P.	Projection of TPVL onto VN

ZL D.P. Projection of TPVL onto TA

Files: NONE

Restrictions: NONE

Diagnostics: NONE

Method:

Compute unit forward vector from $\underline{TEL}$ to $\underline{TE}$.

$$\underline{FWD} = \frac{\underline{TE} - \underline{TEL}}{|\underline{TE} - \underline{TEL}|}$$

Compute two vectors normal to $\underline{FWD}$ and each $\underline{TA}$.

$$\underline{VN} = \underline{TA} \times \underline{FWD}$$

$$\underline{VNL} = \underline{TAL} \times \underline{FWD}$$

Compute orthogonal vectors $\underline{TF}$ and $\underline{TFL}$.

$$\underline{TF} = \underline{VN} \times \underline{TA}$$

$$\underline{TFL} = \underline{VNL} \times \underline{TAL}$$

Compute vectors from tool tip to surface contact point for each case.

$$\underline{TPV} = \underline{TP} - \underline{TE}$$

$$\underline{TPVL} = \underline{TPL} - \underline{TEL}$$

Project into local tool axis system.

$$X = \underline{TPV} \cdot \underline{TF} \qquad XL = \underline{TPVL} \cdot \underline{TFL}$$
$$Y = \underline{TPV} \cdot \underline{VN} \qquad YL = \underline{TPVL} \cdot \underline{VNL}$$
$$Z = \underline{TPV} \cdot \underline{TA} \qquad ZL = \underline{TPVL} \cdot \underline{TAL}$$

Compute corresponding points on tool.

$$\underline{XTP} = \underline{TE} + XL\,\underline{TF} + YL\,\underline{VN} + ZL\,\underline{TA}$$

$$\underline{XTPL} = \underline{TEL} + X\,\underline{TFL} + Y\,\underline{VNL} + Z\,\underline{TAL}$$

Notes: NONE

Referenced by: GCHK

Subsidiaries: CPRINT
CROSSV
DOTF
VNORM