

Robótica

Industrial I

Modelagem, Utilização e Programação

João Maurício Rosário

Robótica

Industrial I

Modelagem, Utilização e Programação



Copyright © 2010 by Editora Baraúna SE Ltda

Ilustração da capa: Alvaro Joffre Uribe Quevedo

Projeto Gráfico
Aline Benitez

Revisão
Priscila Loiola

CIP-BRASIL. CATALOGAÇÃO-NA-FONTE
SINDICATO NACIONAL DOS EDITORES DE LIVROS, RJ

R798r

Rosário, João Maurício

Robótica industrial I : modelagem, utilização e programação /
João Mauricio Rosário. - São Paulo : Baraúna, 2010.

ISBN 978-85-7923-145-2

1. Robótica. 2. Robos industriais. 3. Mecatrônica. I. Título.

10-1713.

CDD: 629.892

CDU: 681.865.8

20.04.10 05.05.10

018770

Impresso no Brasil
Printed in Brazil

DIREITOS CEDIDOS PARA ESTA EDIÇÃO À EDITORA BARAÚNA
www.EditoraBarauna.com.br

Rua João Cachoeira, 632, cj.11
CEP 04535-002 Itaim Bibi São Paulo SP
Tel.: 11 3167.4261

www.editorabarauna.com.br

Sumário

OBJETIVOS DESTE LIVRO	19
CAPÍTULO 1: <i>Introdução à Robótica</i>	21
1.1 - Introdução	25
1.2 - Histórico	26
1.3 - A utilização de robôs dentro do processo de Automação ..	31
1.4 - Fatores de Desenvolvimento de Robôs Industriais ..	33
1.5 - Vantagens e Desvantagens da Robótica Industrial. .	33
1.5.1 - Principais Vantagens	35
1.5.2 - Custo de Aquisição	35
1.5.3 - Impacto Social	36
1.6 - Conceitos Básicos de um Robô	40
1.6.1- Braço Mecânico	40
1.6.2 - Sensoriamento e Programação.....	41
1.6.3 - Aspectos relacionados à utilização de Robôs Industriais	42
1.6.3.1 - Classificação de um Sistema Automatizado ..	42
1.6.3.2 - Funcionalidades e Habilidades de um Robô....	42
1.7 - Principais Aplicações de Robôs Industriais	43
1.7.1 - Utilização de Robôs Industriais no Brasil e no mundo ..	43
1.7.2 - Indústria Automobilística	44
1.7.3 - Formação de Profissionais em Robótica no Brasil ..	45
1.8 - Conclusão	46
1.9 - Referências Bibliográficas	47
 CAPÍTULO 2: <i>Aspectos Construtivos</i>	49
2.1 - Sistemas Robóticos.....	53
2.1.1 - Juntas Robóticas	55
2.1.2 - Tipos de Juntas	56
2.1.3 - Graus de Liberdade.....	59
2.2 - Classificação de Manipuladores Robóticos	59
2.2.1 - Estrutura Cinemática	59
2.2.2 - Geometria do Robô	61
2.2.2.1 - Robô de Coordenadas Cartesianas	62

2.2.2.2 - Robô de Coordenadas Cilíndricas	63
2.2.2.3 - Robô de Coordenadas Polares (Esféricas)	64
2.2.2.4 - Robô de Coordenadas de Revolução (Articulado).....	64
2.2.2.5 - Robô SCARA.	64
2.2.3 - Estudo Comparativo da Área de Trabalho para diferentes configurações de robô.....	66
2.2.4 - Avaliação de configurações robóticas na capacidade de realização de tarefas	67
2.2.5 - Construção de Vínculos	68
2.3 - Sensores	69
2.4 - Acionamento e Controle de Robôs.	71
2.4.1 - Tipos de Acionamento de um braço robótico	74
2.4.2 - Formas de Acionamento de um braço robótico	74
2.4.2.1 - Acionamento Elétrico.	74
2.4.2.2 - Acionamento Hidráulico	76
2.4.2.3 - Acionamento Pneumático	77
2.4.2.4 - Comparaçāo de utilização de Acionamentos.	78
2.4.3 - Classificação pela forma de conexāo: Drivers Direto e Indireto	79
2.4.4 - Atuadores	79
2.5 - Programação de Robôs.....	80
2.6 - Precisão e Repetibilidade	81
2.7 - Garras e Ferramentas	81
2.7.1 - Garra de dois dedos.	84
2.7.2 - Garra de três dedos	84
2.7.3 - Garra para preensāo de objetos cilíndricos.....	84
2.7.4 - Garra para objetos frágeis	85
2.7.5 - Garra Articulada	85
2.7.6 - Garras a vácuo e acionamento eletromagnético	86
2.7.7 - Adaptador Automatizado de Garra	86
2.8 - Robôs com Estrutura Paralela.	87
2.8.1 - Comparaçāo entre Manipulador serial e Paralelo..	88
2.8.2 - Manipuladores Paralelos	90
2.8.2.1 - Manipuladores de Três Graus de Liberdade	92
2.8.2.2 - Manipuladores de Quatro Graus de Liberdade..	94
2.8.2.3 - Manipuladores de Cinco Graus de Liberdade..	95

2.8.2.4 - Manipuladores de Seis Graus de Liberdade	96
2.8.3 - Exemplos de aplicação dos manipuladores paralelos	97
2.8.3.1 - Aplicações Espaciais	97
2.8.3.2 - Aplicações Médicas	97
2.8.3.3 - Aplicações Industriais.	98
2.9 - Avaliação de Desempenho de Robôs Industriais	99
2.9.1 - Precisão, Repetibilidade e Desempenho.	100
2.9.2 - Características de Desempenho	103
2.9.3 - Características de postura	104
2.9.4 - Precisão de postura (AP)	105
2.9.5 - Repetibilidade de Postura (RP)	107
2.9.6 -Variação multidirecional na precisão de postura (vAP)	107
2.9.7 - Deslocamento nas características de postura	107
2.10 - Conclusão	108
2.11 - Referências Bibliográficas	111
 CAPÍTULO 3: <i>Modelagem Cinemática</i>	115
3.1 - Introdução	119
3.2 - Sistemas de Referência	121
3.3 - Sistemas de Coordenadas utilizados em Células Robotizadas	122
3.4 - Modelo Geométrico.	129
3.5 - Transformação de Coordenadas	131
3.6 - Apresentação de Exemplos	132
3.6.1 - Robô Elementar (1 GL) - Pêndulo simples .	132
3.6.2 - Robô com 2 GL - Pêndulo duplo	133
3.7 - Matrizes de Transformação Homogênea.	134
3.8 - Sistemática de Denavit-Hartenberg (DH)	136
3.8.1 - Algoritmo Resumido.	140
3.8.2 - Obtenção da Matriz de Transformação Homogênea ${}^{i-1}A_i$	142
3.8.3 - Matriz de Transformação T	144
3.9 - Exemplos de obtenção dos Parâmetros de DH. . .	145

3.10 - Modelagem Através do Método de Vetores Locais	150
3.10.1 - Metodologia	150
3.10.2 - Comparação do modelo cinemático com o obtido através de DH.	158
3.11 - Obtenção da Matriz de Orientação	160
3.11.1 - Ângulos de Euler.	161
3.11.2 - Ângulos de RPY	164
3.11.3 - Exemplo	167
3.11.4 - Quartenions	169
3.11.5 - Exemplo de Aplicação.	170
3.12 - Referências Bibliográficas	173

CAPÍTULO 4: *Modelagem Cinemática Exemplos*

Práticos	175
4.1 - Robô Stanford	179
4.1.1 - Parâmetros de Denavit-Hartenberg (DH) . .	180
4.1.2 - Matrizes de Transformação Homogenêa (MTH) .	180
4.1.3 - Matriz posição-orientação final 0T6	181
4.2 - Robô Puma 560 TM	181
4.2.1 - Parâmetros de Denavit Hartenberg (DH) . .	182
4.2.2 - Matrizes de Transformação Homogenêa (MTH) .	182
4.2.3 - Cálculo do Vetor de Posição e da Matriz de Orientação	183
4.2.3.1 - Matriz posição-orientação final 0TN . . .	183
4.2.3.2 - Orientação final (elemento terminal) . . .	183
4.2.3.3 - Posição final (elemento terminal)	184
4.2.3.4 - Verificação do modelo	184
4.3 - Robô Manutec r3 TM	184
4.3.1 - Parâmetros de Denavit-Hartenberg (DH) . .	187
4.3.2 - Matrizes de Transformação Homogenêa (MTH) .	187

4.3.3 - Cálculo do Vetor de Posição e da Matriz de Orientação	188
4.3.4 - Cálculo Matriz Posição-Orientação Final 0T_N	188
4.4 - Manipulador Submarino Kraft	189
4.4.1 - Parâmetros de Denavit-Hartenberg (DH)	190
4.4.2 - Matrizes de Transformação Homogenêa (MTH)	191
4.4.3 - Cálculo do Vetor de Posição e da Matriz de Orientação	192
4.4.3.1 - Orientação final	192
4.4.3.2 - Posição final	192
4.5 - Robô ABB IRB-1400 TM	192
4.5.1 - Parâmetros de Denavit-Hartenberg (DH) e Workspace	193
4.5.1.1 - Parâmetros de DH	193
4.5.1.2 - Limites angulares e velocidades/acelerações mínimas e máximas	194
4.5.2 - Matrizes de Transformação Homogenêa (MTH)	194
4.5.3 - Cálculo do Vetor de Posição e da Matriz de Orientação	195
4.5.3.1 - Definição de Parâmetros e Variáveis	195
4.5.3.2 - Vetor Orientação e Quaternions associados	196
4.5.4 - Posição Final	197
4.5.5 - Implementação Computacional do Modelo Cinemático	197
4.6 - Referências Bibliográficas	200
 CAPÍTULO 5: <i>Modelagem Cinemática Inversa</i>	201
5.1 - Introdução	205
5.1.1 - Métodos analíticos	205
5.1.2 - Métodos numéricos iterativos	206
5.2 - Descrição Matemática de um Robô com N GL	207

5.3 - Procedimento para obtenção do Modelo Cinemático Inverso	208
5.3.1 - Modelagem Cinemática Direta	208
5.3.2 - Modelagem Cinemática Inversa	209
5.3.3 - Matriz Jacobiana	210
5.3.4 - Exemplos	211
5.3.4.1 - Manipulador 1 GL.	212
5.3.4.2 - Manipulador 2 GL.	212
5.4 - Inversão da Matriz Jacobiana	213
5.4.1 - Cálculo da Matriz Jacobiana	214
5.4.2 - Controle de Posição de um Manipulador a partir da Inversão da Matriz Jacobiana	215
5.5 - Exemplos de Aplicação.	217
5.5.1 - Dispositivo Robótico Cartesiano 3 GL	217
5.5.1.1 - Parâmetros de Denavit-Hartenberg (DH)	218
5.5.1.2 - Matrizes de Transformação Homogêneas (MTH)	218
5.5.1.3 - Modelo Cinemático Direto	219
5.5.1.4 - Modelo Cinemático Inverso.	220
5.5.2 - Dispositivo Robótico 2R + 1T	221
5.5.2.1 - Parâmetros de Denavit-Hartenberg (DH)	222
5.5.2.2 - Matrizes de Transformação Homogêneas (MTH)	222
5.5.2.3 – Matriz de Transformação Final	222
5.5.2.4 - Implementação em Matlab TM	223
5.5.2.5 - Cálculo do Jacobiano	223
5.6 - Referências Bibliográficas	227
CAPÍTULO 6: <i>Geração de Trajetórias</i>	229
6.1 - Introdução	233
6.2 - Arquitetura de Controle e Geração de Movimentos de um Robô	235
6.3 - Controle de Trajetórias.	236

6.3.1 - Controle Ponto-a-Ponto (PTP)	236
6.3.2 - Controle por Trajetória Contínua	237
6.4 - Geração de Trajetórias no Espaço de Juntas . .	238
6.4.1 - Aceleração em Degrau.	239
6.4.2 - Aceleração em Polinômio Cúbico	242
6.5 - Algoritmo Numérico para Geração de Trajetórias .	243
6.6- Implementação de Algoritmo	245
6.7 - Discretização do caminho	250
6.7.1 - Discretização Linear	251
6.7.2 - Discretização em semicírculo.	252
6.8 - Exemplos de Simulação	256
6.8.1 - Condições utilizadas para Simulação	256
6.8.2 - Apresentação de Resultados da Simulação .	259
6.9 - Interpolação e filtragem de pontos de passagem no espaço da juntas	269
6.9.1 Interpolação Linear da Trajetória	270
6.9.1.1 Algoritmo implementado	270
6.9.1.2 - Exemplo.	271
6.9.2 Filtragem da Trajetória Interpolada	272
6.9.2.1 Filtragem na forma triangular.	273
6.9.2.2 Filtragem na forma retangular	274
6.9.2.3 Exemplo.	275
6.9.3 - Equação para a obtenção da trajetória filtrada e interpolada.	275
6.9.4 - Exemplo de Aplicação e resultados	276
6.10 - Conclusão	279
6.11 - Referências Bibliográficas	280

CAPÍTULO 7: <i>Análise de Desempenho, Capacidade e Precisão</i>	281
7.1 - Introdução	285
7.2 - Critérios Utilizados na Seleção de Robôs	285
7.3 - Precisão e Repetibilidade	288

7.4 - Características de Desempenho	292
7.4.1 - Postura	292
7.4.1.1 - Precisão de Postura (AP)	293
7.4.1.2 - Repetibilidade de Postura (R_p)	294
7.4.1.3 - Variação Multidirecional na Precisão de Postura (vA_p)	295
7.4.1.4 - Deslocamento nas Características de Postura (dA_p)	295
7.4.2 - Resolução Espacial	298
7.4.3 - Repetibilidade	298
7.4.4 - Complacência	299
7.5 - Exemplo de Especificação de Robôs Industriais	299
7.5.1 - Robô industrial FANUC ArcMate 100iBE .	300
7.5.2 - Robô industrial FANUC ArcMate 100iBE .	302
7.5.3 - Características de outros Robôs industriais .	304
7.6 - Referências Bibliográficas	306
 CAPÍTULO 8: <i>Programação de Robôs Industriais</i>	. 307
8.1 - Introdução	311
8.2 - Estrutura de Controle de um Robô Industrial .	312
8.3 - Programação de Tarefas em Robôs Industriais	312
8.3.1 - Considerações Iniciais	312
8.3.2 - Programação de Robôs Industriais	316
8.3.3 - Painel de Acionamento e Controle	318
8.3.4 - Elementos de Supervisão e Controle de um Robô Industrial	321
8.3.4.1 - Unidade de Armazenamento	321
8.3.4.2 - Calibração de um Robô e Acessórios	321
8.4 - Métodos de Programação de Robôs Industriais .	323
8.4.1 - Programação por aprendizagem	323
8.4.2 - Método de Aprendizagem (Teaching)	325
8.4.3 - Programação PTP (Point to point)	327

8.4.4 - Programação CP (Controlled Path)	328
8.4.5 - Programação por Aprendizagem Direta	328
8.4.6 - Programação Master-Slave.	330
8.5 - Linguagem de Programação de Robôs	331
8.5.1 - Níveis de programação	332
8.5.2 - Vantagens da utilização de programação através de linguagens	333
8.5.3 - Procedimentos Básicos para Implementação e Execução de Programas	333
8.5.3.1 - Aspectos Relacionados à Segurança	333
8.5.3.2 - Planejamento do Programa	334
8.5.4 - Programação de um Robô utilizando linguagem própria	335
8.5.5 - Programação Estruturada de Robô Industrial	341
8.6 - Programação off-line de Robôs Industriais	344
8.7 - Conclusão	349
8.8 - Referências Bibliográficas	350
CAPÍTULO 9: <i>Programação Off-line de Robôs</i>	351
9.1 - Introdução	355
9.2 - Softwares de Simulação e Programação Off-line	356
9.3 - Sistemas de Coordenadas e Modelagem Cinemática	358
9.3.1 - Movimentação Angular das Juntas	359
9.3.2 - Movimento na Direção Cartesiana	360
9.3.3 Movimento de Reorientação da Ferramenta	361
9.4 - Programação através de Visualização Gráfica	363
9.5 - Exemplos de Aplicações.	364
9.5.1 - Comentários da Implementação	369
9.5.2 - Outros Exemplos	370
9.6 - Conclusão	375
9.7 - Referências Bibliográficas	376

CAPÍTULO 10: <i>Integração de Robôs em Células de Manufatura</i>	379
10.1 - Implementação de Software de Programação Off-Line e Simulação	383
10.1.1 - Pacote Computacional Desenvolvido	384
10.1.2 - Implementação de Bibliotecas	384
10.1.3 - Descrição das Bibliotecas de um Programa Off-Line	385
10.1.4 - Módulo de Simulação (SIMULA)	387
10.1.5 - Módulo de geração de ferramentas, obstáculos e ambiente de atuação (OBSTÁCULO)	388
10.1.6 - Módulo de geração de trajetórias (TRAJETÓRIA)	388
10.2 - Integração de Robôs	389
10.3 - Programação Off-Line utilizando Softwares Comerciais	390
10.4 - Integração de Células Robotizadas	391
10.5 - Integração de Diferentes Dispositivos a partir de E/S.	395
10.6 - Sistemas Robóticos Cooperativos	397
10.7 - Modelagem e Simulação Off-Line	398
10.7.1 - Modelagem	398
10.7.2 - Simulação	399
10.7.3 - Exemplo	400
10.8 - Exemplo: Célula Integrada de Manufatura ..	402
10.9 - Identificação de Posicionamento de Robôs ..	404
10.9.1 - Metodologia de Calibração	406
10.9.2 - Infraestrutura necessária	408
10.9.2.1 - Dispositivos Mecânicos	408
10.9.2.2 - Hardware e Software	409
10.9.3 - Software de Identificação de Pontos no Paleta e Calibração de Ferramenta	410
10.9.4 - Aplicativos Desenvolvidos	411

10.9.4.1 - Sistema de Calibração	411
10.9.4.2 - Ajuste de Parâmetros de Calibração	412
10.9.4.3 - Procedimento de Ajuste Focal	413
10.9.4.4 - Operação de calibração	416
10.9.4.5 - Aplicativo para Calibração do Palete.	419
10.10 - Conclusão	419
10.11 - Referências Bibliográficas	420

CAPÍTULO 11: <i>Exemplo de Implementação de uma Célula Colaborativa Robotizada</i>	423
11.1 - Introdução	427
11.2 - Célula Colaborativa Robotizada.	428
11.3 - Plataforma de Posicionamento	431
11.3.1 - Encoders Incrementais	432
11.3.2 - Sensores Indutivos.	433
11.3.3 - Sistema de Acionamento	434
11.4 - Modelagem Cinemática	435
11.4.1 - Cinemática Direta.	436
11.4.1.1 - Matrizes de Rotação	438
11.4.1.2 - Vetores Locais	438
11.4.2 - Cinemática Inversa	439
11.5 - Estrutura de Acionamento e Controle	441
11.6 - Simulador Virtual	442
11.6.1 - Módulo Geração de Trajetórias	443
11.6.2 - Módulo de Acionamento	444
11.6.3 - Módulo Modelo Cinemático.	444
11.6.4 - Interface Gráfica	445
11.6.4.1 - Velocidade do Motor	446
11.6.4.2 - Deslocamento do motor.	447
11.6.4.3 - Sinal de Controle	448
11.6.4.4 - Movimento no Centro da Mesa	448
11.6.4.5 - Estudo do erro	450

11.7 - Implementação de Sistema Colaborativo em Automação	452
11.7.1 - Ambientes Colaborativos para Ensino e Pesquisa baseados em WEB	452
11.7.2 - Plataforma de Validação	453
11.7.3 - Estrutura Proposta	454
11.7.4 - Implementação do Sistema Supervisório	456
11.7.4.1 - Descrição Funcional	456
11.7.4.2 - Protocolo de Comunicação e Topologia de Implementação	457
11.7.4.3 - GRAFCET Funcional	459
11.7.4.4 – Definição de Variáveis do Sistema	461
11.7.4.5 - Programas Implementados nos Robôs ABB	468
11.7.5 - Telas Implementadas em Ambiente LabView™	470
11.8 - Sistema de Supervisão e Controle da Plataforma com 3 GL	472
11.8.1 - Descrição da Parte Operativa	474
11.8.2 - Sistema de Supervisão e Controle	477
11.8.2.1 - Descrição do Sistema Operativo	477
11.8.2.2 - Interface de Controle	478
11.8.3 - Telas Implementadas	479
11.8.3.1 - Tela de Inicialização	479
11.8.3.2 - Modo de Calibração	480
11.8.4 - Geração de Arquivos	483
11.8.4.1 - Modo Aprendizado	483
11.8.4.2 - Movimentação a partir de arquivo padrão EXCELTM	485
11.8.4.3 - Modo Controle	485
11.8.4.4 - Tela de Controle (modo automático)	486
11.8.4.5 - Tela de Status	487
11.8.4.6 - Tela de Monitoramento	487

11.8.4.7 - Tela de Ajuste de Parâmetros do Controlador de Posição PID.....	489
11.8.4.8 - Tela de Modo de Segurança	489
11.8.4.9 - Geração de Arquivo de Banco de Dados.	491
11.9 - Conclusão	491
11.10 - Referências Bibliográficas.....	492

OBJETIVOS DESTE LIVRO

Este livro é fruto de um trabalho de aproximadamente 25 anos de experiência nesta área, se consolidando a partir do oferecimento da disciplina Modelagem e Controle de Manipuladores Robóticos dentro do programa de Pós-graduação da Faculdade de Engenharia Mecânica da UNICAMP. O texto contribui não somente para os profissionais da área de Engenharia, mas todas as pessoas interessadas em conhecer e se aprofundar na área de Mecatrônica.

O contexto apresentado neste livro é dividido em capítulos distintos enfatizando uma formação em robótica industrial com ênfase na Modelagem, utilização e programação de robôs industriais, destinado a introduzir conceitos sobre Aspectos Construtivos de Manipuladores Industriais com ênfase nas características construtivas, modelagem cinemática direta e inversa, critérios de utilização de robôs industriais, programação estruturada de robôs, programação off-line e Integração de Sistemas Robóticos em linhas de Produção Industrial.

Este livro apresenta uma abordagem sistêmica, mas de fácil compreensão, fornecendo aos leitores uma visão ampla do assunto, permitindo a formação de massa crítica a nível industrial ou acadêmico, sendo apresentados e analisados exemplos de aplicações industriais reais, evidenciando, assim, conceitos e componentes descritos nos diferentes capítulos desta obra.

João Maurício Rosário
Janeiro de 2010

CAPÍTULO 1

Introdução à Robótica



"Unfortunately, there was no lobby for anything else."

CAPÍTULO 1

Introdução à Robótica

A quantidade de robôs de uso industrial vem aumentando nos últimos anos, pela implementação de robôs nas indústrias automobilísticas, particularmente, em suas linhas de montagem.

Ao mesmo tempo, a utilização de robôs vem se tornando cada vez mais frequente também na medicina: área cirúrgica (cardíaca, oftalmológica, etc.), teleoperação, na reabilitação através de próteses de pernas e braços, em operações em ambientes adversos (exploração submarina, cavernas, espacial, etc.).

Neste capítulo é realizada uma introdução à robótica industrial, apresentando uma revisão histórica sobre a evolução da robótica até os nossos dias, focando em particular a robótica de manipulação industrial, a partir da descrição de exemplos de utilizações de robôs industriais em instalações produtivas.

1.1 - Introdução

Na sociedade atual, há uma crescente necessidade de se realizar tarefas com eficiência e precisão. Existem também tarefas a serem realizadas em lugares onde a presença humana se torna difícil, arriscada e até mesmo impossível, como o fundo do mar ou a imensidão do espaço. Para realizar essas tarefas, se faz cada vez mais necessária a presença de dispositivos mecatrônicos (robôs), que realizam essas tarefas sem risco de vida. A Robótica é a área que se preocupa com o desenvolvimento de tais dispositivos, é uma área multidisciplinar, altamente ativa e que busca o desenvolvimento e a integração de técnicas e algoritmos para a criação de robôs.

A Robótica envolve matérias como engenharia mecânica, engenharia elétrica, inteligência artificial, entre outras, com uma perfeita harmonia, que se faz necessária para se projetar essas maravilhosas tecnologias. Temos hoje robôs em várias áreas de nossa sociedade: robôs que prestam serviços, como desarmamento de bombas, robôs com a nobre finalidade da pesquisa científica e educacional e até mesmo os robôs operários, que se instalaram em nossas fábricas e foram responsáveis pela “Segunda Revolução Industrial”, revolucionando a produção em série, substituindo a carne e o osso pelo aço, agilizando e fornecendo maior qualidade aos produtos.

Décadas atrás, os robôs faziam parte apenas da ficção científica, fruto da imaginação do homem. No início dos anos 60, os primeiros robôs começaram a ser usados com o objetivo de substituir o homem em tarefas que ele não podia realizar por envolverem condições desagradáveis, tipicamente contendo altos níveis de calor, ruídos, gases tóxicos, esforço físico extremo, trabalhos tediosos e monótonos. Existem duas tendências, nos últimos 20 anos, que garantem a evolução dos robôs:

1. O constante aumento dos níveis salariais dos empregados;

- O extraordinário avanço tecnológico no ramo de computadores, que induz à redução dos preços do robô e uma significativa melhoria em seu desempenho.

A utilização de robôs industriais são cada vez mais frequentes em ambientes conhecidos, acessíveis e inacessíveis, como também em ambientes desconhecidos, exigindo a integração dos mesmos dentro de um processo que envolve não só o robô como também outros dispositivos associados à aplicação.

Em ambientes industriais, como não poderia deixar de ser, a exigência da integração dos mesmos requer uma condição semelhante às citadas. A robotização vem da necessidade de se obter um processo, por exemplo, mais qualificado, garantindo repetições e precisões que fatalmente levariam à fadiga em ambientes insalubres ou não. Algumas atenuantes devem ser buscadas, fazem necessário para propiciar uma inserção consciente e correta na utilização de robôs em ambientes industriais.

1.2 - Histórico

Uma das maiores fantasias do homem é construção de uma máquina com “Inteligência Artificial”, capaz de agir e pensar como ele. No entanto, este desejo esconde em seu subconsciente a vontade de possuir um “escravo metálico” que satisfaça todos os seus desejos, este sonho humano está perto de se tornar realidade com o espantoso avanço da tecnologia.

A palavra robô tem origem da palavra tcheca robotnik, que significa servo. O termo robô foi utilizado inicialmente por Karel Capek em 1923, nesta época a ideia de um “homem mecânico” parecia vir de alguma obra de ficção. Não é só do homem moderno o desejo de construir tais robôs, existem alguns fatos históricos que nos mostram que a ideia não é nova, por exemplo, existem inúmeras referências sobre o “Homem Mecânico” construído por relojoeiros com a finalidade de se exibir em feiras.

Temos relatos também da realização de várias “Animações Mecânicas” como o leão animado por Leonardo da Vinci e seus esforços para fazer máquinas que reproduzissem o voo das aves. Porém estes dispositivos eram muito limitados, pois não podiam realizar mais do que uma tarefa, ou um conjunto reduzido das mesmas. A ideia de se construir robôs começou a tomar força no início do século XX com a necessidade de aumentar a produtividade e melhorar a qualidade dos produtos. Foi nesta época que o robô industrial encontrou suas primeiras aplicações. O pai da robótica industrial pode ser considerado George Devol. As tabelas 1.1 e 1.2 apresentam uma descrição dos principais fatos históricos que contribuíram para o desenvolvimento da Robótica.

Período	Fato Histórico
IV A.C. Grécia	Aristóteles relatava os primeiros princípios da robótica, referentes à utilização de instrumentos dedicados a trabalhos determinados, sem auxílio das mãos, que reduziriam os esforços do homem, enfatizando o conceito de mestre e escravo.
Século XIII	Inicia-se a revolução industrial; com a evolução de novas fontes, novos mecanismos e instrumentos. Com os novos conceitos industriais, se torna possível a evolução da maquinaria capaz de controlar uma série de ações sequenciadas.
Século XIX	No final do séc. XIX é que se inicia o desenvolvimento da máquina. As máquinas começam a ser usuais e exposições de máquinas sempre a promover os últimos eventos tecnológicos. O motor elétrico é introduzido na indústria. A máquina substitui o homem.
1ª Guerra Mundial	A 1ª Guerra Mundial trouxe muitas mudanças. O poder da máquina mostrou-nos a sua forma negativa e destrutiva.

1921	Foi introduzida pela primeira vez a palavra “Robot” pelo dramaturgo Karel Capek na peça teatral “Rossum’s Universal Robots” (R.U.R.), em que os robôs foram criados para substituir o homem nos trabalhos pesados. O robô começou a ser visto como uma máquina “humana” com inteligência e personalidade individual.
1928	Um robô mecânico abre uma exposição de modelos técnicos em Londres.
1940	O célebre escritor americano de ficção científica, Isaac Asimov, estabeleceu 4 Leis para a robótica: 1 ^a Lei: “Um robô não pode ferir um ser humano, ou permanecer passivo deixando um ser humano exposto ao perigo”. 2 ^a Lei: “Um robô deve obedecer às ordens dadas pelos seres humanos, exceto se tais ordens estiverem em contradição com a 1 ^a Lei”. 3 ^a Lei: “Um robô deve proteger sua existência na medida em que essa proteção não estiver em contradição com a primeira e a 2 ^a Lei”. 4 ^a Lei: “Um robô não pode causar mal à humanidade nem permitir que ela própria o faça”. (Esta última lei foi escrita por Asimov somente em 1984). Entretanto, cabe observar que os robôs têm braços e articulações capazes de trabalhos repetitivos e autônomos, mas não no sentido de sensibilidade para se controlarem a si próprios e resolver os problemas que poderão surgir.

Tabela 1.1: Histórico da Robótica (IV AC – 1960).

Período	Fato Histórico
1959	Devol e Joseph F. Engelberger desenvolveram o primeiro robô industrial pela Unimation Inc. Este robô tinha como função uma variedade de tarefas executadas automaticamente. Diferia dos autômatos já que poderia ser reprogramada e remodelada para outras tarefas com um nível de custos pouco elevado.
1960	Nos anos 60, tornou-se significativo o fato de a flexibilidade destas máquinas aumentar, utilizando diferentes tipos de sensores. É a partir de agora que a investigação sobre a robótica começa a incidir no tema Robótica Móvel.
1962	Ernest desenvolveu um computador capaz de controlar uma mão mecânica com sensores tácteis (MH-1). Esta invenção conseguia mover e “sentir” blocos, tornando possível através do controle da mão o empilhamento de blocos sem ajuda humana. Tomovic e Boni desenvolveram um protótipo equipado com um sensor de pressão que quando “sentia” o objeto transmitia informações referentes ao tamanho do objeto para um computador, transmitindo um sinal para um motor que iniciava a ação sobre diferentes moldes.
1963	American Machine e Foundry Company (AMF) introduziram uma versão de um robô comercial (VERSATRAN). Desenvolvimento de diversos projetos de braços para manipuladores, tais como o braço Roehampton e o braço Edinburgh.
1968	Melarthy e outros, no Laboratório de Inteligência Artificial de Standford, desenvolvem um computador com “mãos, olhos, pernas, ouvidos (utilizando manipuladores, câmaras de vídeo e microfones), demonstrando capacidade de identificação, reconhecimento e manipulação de blocos espalhados sobre uma mesa. Pieper estudou o problema de Cinemática de um manipulador controlado por computador.

1969	O Homem pisa no solo lunar pela 1ª vez. Nesta altura já eram utilizados manipuladores para recolher amostras e executar pequenas tarefas perante o comando de um controle remoto. O modo de tele-operação servia para efetuar escavações e outras tarefas de grau de complexidade reduzido.
1970	A Robótica começou a incidir na pesquisa do uso de sensores para facilitar operações manuais.
1971	Kahn and Roth analisaram a dinâmica e o controle de braços flexíveis usando controladores do tipo bang-bang (near minimum time).
1973	Em Standford, Balles e Paul utilizaram um sensor visual e um sensor de peso demonstrando um braço controlado através de computador para montagem de bombas de água do automóvel.
1974	Cincinnati Milacron introduziu o primeiro robô industrial controlado com computador – T3 -"The Tomorrow Tool" – Ferramenta do Futuro, que movia objetos numa linha de montagem. Inoue, no Laboratório de Inteligência Artificial, aprofundou estudos na utilização de sensores de peso (força) e no Draper Laboratory Nevins foram investigadas diferentes técnicas de sensoriamento.
1975	Will e Grossman na IBM desenvolveram um manipulador controlado por computador com sensores tácteis e de peso para realizar a montagem mecânica de 20 partes de uma máquina de escrever.
1980	General Motors, em Detroit, introduziu um robô industrial com "inteligência" eletrônica capaz de reconhecer diferentes componentes numa tela transportadora e com capacidade de escolha do que necessitava.

Tabela 1.2: Histórico da Robótica (1960-1980).

Atualmente, devido aos inúmeros recursos que os sistemas de microcomputadores nos oferece, a robótica atravessa uma época de contínuo crescimento que permitirá, em um curto espaço de tempo, o desenvolvimento de robôs inteligentes, fazendo, assim, a ficção do homem antigo se tornar a realidade do homem atual.

1.3 - A utilização de robôs dentro do processo de Automação

O processo de automação inicia-se através do planejamento de um novo processo ou de um já existente, a partir de um estudo de viabilidade, em que deverão ser considerados as limitações e ganhos na utilização de robôs industriais ou dispositivos automatizados dedicados. Dentre as principais vantagens e inconvenientes envolvendo a utilização de robôs como ferramenta de integração em automação podemos citar algumas, tais como:

a) Custo / benefício

Representa um fator preponderante, cuja amortização do robô é fator decisório na sua aquisição. No mercado brasileiro estima-se que o “pay back” esteja em torno de 36 meses (Tremonti, 2003), em contraposto ao mercado norte americano, que está entre 24 e 12 meses com uma forte tendência de queda (Nagler, 2002). Esse peso é negativo para o mercado nacional e só haverá uma mudança do panorama de mercado a partir do instante em que o número de robôs instalados venha a aumentar e naturalmente haja queda do preço, consequentemente forçando a amortização de forma mais amena.

b) Arquitetura de integração do equipamento

Toda unidade de controle possui um sistema “proprietário”, onde somente o fabricante possui liberdade de abri-lo e alterá-lo. A princípio, esta limitação pode levar a uma dificuldade no processo de integração robô / periféricos.

Nos últimos tempos esse conceito vem ganhando um apelo mais forte no sentido de tornar a arquitetura padronizada, conceito de “aberta”, usufruindo sistemas padronizados disponíveis facilmente no mercado, como, por exemplo, o sistema operacional Windows®, utilizados em algumas interfaces homem-máquina e protocolos de comunicação como TCP/IP, devicenet, profibus, interbus, entre outros, somando aí inúmeros ganhos técnicos e econômicos dos integradores e usuários.

Obviamente, tomando-se as devidas proporções em relação à tecnologia desenvolvida, como os algoritmos de controle dos movimentos do robô, interface entre o braço do robô e seus acionamentos e etc., propicia uma maior flexibilidade e mais facilidade de integração, principalmente na questão da interface com outros equipamentos que possuem sistemas “abertos”, através dos protocolos de comunicação que começam a dividir uma mesma tendência, fortalecendo um sistema padronizado que naturalmente diminui o custo de atualização e novos implementos.

c) Planejamento de células robotizadas

É uma ferramenta decisiva dentro do processo de implementação ou alteração de células de manufatura automatizada, devendo ser analisada de forma detalhada para compreensão e planejamento de um novo processo ou alteração de uma já existente, em que através do planejamento e concepção de células robotizadas é possível a simulação gráfica do robô e do ambiente, com o objetivo de especificar equipamentos, envelope de trabalho, e gerar trajetórias para os elementos robóticos.

A descrição de situações de layout e integração com outros equipamentos periféricos simulando e programando de forma gráfica com a visualização, a partir do dimensionamento e projeto mecânico, possibilita o processo de identificação de elementos, método gráfico para modelagem do ambiente. É conhecida como uma das possibilidades de programação off-line, prevendo situações inusitadas e garantindo uma estratégia segura de implementação de um processo automatizado e robotizado.

d) Custos envolvidos

Os inconvenientes mais significativos na utilização de robôs industriais são os altos custos envolvidos na aquisição, implantação, manutenção e principalmente em relação à mão de obra qualificada no uso dessa tendência tecnológica cada vez mais real, mas que são sensivelmente amortizados por meio do planejamento e popularização como contribuição para o processo.

1.4 - Fatores de Desenvolvimento de Robôs Industriais

Na época em que foram lançados os primeiros robôs industriais, na década de 1960, os robôs eram caros e acessíveis a pouquíssimas empresas existentes em países mais desenvolvidos, principalmente no Japão e nos Estados Unidos. No entanto, a partir de 1976 começaram a baixar os preços de uma forma extremamente acelerada.

O grande responsável por esta brutal redução de custos que ocorreu na informática e na robótica é a microeletrônica. Com o avanço desta disciplina, por exemplo, foi possível colocar toda a capacidade do ENIAC, o primeiro computador a válvula desenvolvida em 1950, em uma pastilha de silício de menos de $0,5 \text{ cm}^2$. Ressaltando que isso se consegue com velocidade de processamento muito superior e a um custo infinitamente menor.

Desta forma, os microprocessadores influenciaram diretamente a capacidade de todas as máquinas industriais, tendo impacto decisivo nas tecnologias associadas à robótica, permitindo que a capacidade de processamento de informações se multiplicasse de forma estrondosa, além de baratear o custo dos robôs, tornado-os mais acessíveis.

1.5 – Vantagens e Desvantagens da Robótica Industrial

O crescimento de tecnologia relacionado à robótica gerou grandes benefícios. A automação possibilita grandes incre-

mentos na produtividade do trabalho, possibilitando que as necessidades básicas da população possam ser atendidas. Além de aumentar a produção, os equipamentos automatizados possibilitam uma melhora na qualidade do produto, uniformizando a produção e eliminando perdas e refugos.

A automação também permite a eliminação de tempos mortos, ou seja, permite a existência de “operários” que trabalhem 24 horas por dia sem reclamarem, que leva a um grande crescimento na rentabilidade dos investimentos.

A microeletrônica permite flexibilidade ao processo de fabricação, ou seja, permite que os produtos sejam produzidos conforme as tendências do mercado, evitando que se produzam estoques de produtos invendáveis.

As características citadas acima mostram que a microeletrônica possibilita que não haja nem escassez nem desperdício, com melhor qualidade de vida e de produção, aliada a um menor esforço.

Sem dúvida a automação industrial foi e é um grande impulsionador da tecnologia de robótica. Cada vez mais se tem procurado aperfeiçoar os dispositivos, dotando-os com inteligência para executar as tarefas necessárias. Por exemplo, usando Redes Neurais procura-se a linearização de acionamentos eletromecânicos; com Fuzzy Logic pode-se fazer o planejamento de trajetória para robôs redundantes; ou utilizando Sistemas Especialistas é possível a detecção de vazamento de água a partir da aquisição remota de consumo.

Há alguns anos, foi concebida a ideia de que sistemas mecânicos poderiam ser controlados por operações numérico-aritméticas. As máquinas-ferramentas CNC (Controle Numérico Computadorizado) são máquinas operadas, e suas velocidades são controladas por computadores conectados aos motores das máquinas.

1.5.1 – Principais vantagens

As vantagens na utilização de robôs industriais são numerosas, destacando-se entre elas o aumento da produtividade, a melhoria e consistência na qualidade final de um produto, minimização de operações, uma menor demanda de contratação de mão de obra especializada, que é difícil de encontrar, a contabilidade no processo, a facilidade na programação e uso dos robôs, a operação em ambientes difíceis e perigosos ou em tarefas desagradáveis e repetitivas para o ser humano e, finalmente, a capacidade de trabalho sem interrupções por longos períodos.

Na prática, a aplicação de robôs na indústria requer uma solução confiável e robusta que desempenhe de forma consistente as funções predeterminadas, exigindo que indústrias que desejam automatizar todos os seus processos, e que possuam problemas complexos para resolver, estabeleçam a aquisição de robôs de precisão com alto custo, que muitas vezes não trabalham como o especificado.

Em algumas aplicações os processos alcançam 100% de robotização, tal como na manipulação de materiais diversos, soldagem por resistência por pontos e pintura na indústria automobilística. E esse sucesso não é por serem processos simples; pelo contrário, são processos complexos, mas é um bom exemplo da relação custo-benefício, além de substituir a mão de obra humana em um trabalho repetitivo, difícil e, em muitas vezes, de alto risco [2].

1.5.2 – Custo de Aquisição

O principal fator que impede a adoção em massa de sistemas robotizados industrialmente é seu alto custo inicial. O tempo que leva para se recuperar o investimento em um robô depende dos custos de compra, instalação e manutenção. Este tempo não é fixo, depende da fábrica onde o robô será instalado e de sua aplicação. O preço de um robô será determinado pelas suas dimensões,

grau de sofisticação e complexidade, exatidão e confiabilidade. Na especificação de Sistemas Automatizados utilizando dispositivos robóticos devemos levar em consideração as seguintes condições:

- número de empregados substituídos pelo robô;
- número de turnos realizados por dia;
- produtividade comparada a seu custo;
- custo de projeto e manutenção;
- custo dos equipamentos periféricos.

1.5.3 – Impacto Social

As transformações que ocorrem causadas pelo advento dos robôs muitas vezes podem não estar visíveis para grande parte das pessoas que não convivem no ambiente fabril, contudo a ascensão da robótica nas fábricas faz parte da mesma tendência que vem determinando, nos últimos anos, a crescente automatização dos bancos, do comércio e das empresas em geral, causados pelo advento da informática.

Nos últimos tempos, através da automação, observou-se o decréscimo do nível de emprego nas atividades industriais. Em curto prazo, automação levanta problemas como o desemprego, necessária reconversão e treinamento pessoal, consequências da redução de horas de trabalho e, questões relativas ao aumento de salários em atividades de maior produtividade.

Em comparação com a utilização de robôs em substituição às mãos de obra humanas se veem algumas vantagens, tais como a capacidade de fazer trabalhos árduos de maneira ininterrupta, sem suspensão de atividades por motivos trabalhistas; e possibilidade de manter uma qualidade uniforme na produção e não necessitar de condições ambientais especiais para o desenvolvimento do trabalho, como ar condicionado, luz e silêncio. Por outro lado, o robô tem o aprendizado, memória e movimentos limitados se comparados ao ser humano.

No que se refere ao meio fabril, por um lado às indústrias recrutam robôs e computadores guiados por uma necessidade

crucial para sobrevivência no mercado, para conquistar maior produtividade e qualidade para seus produtos, de forma barata, e assim assegurar competitividade frente aos concorrentes. Por outro lado, os trabalhadores ficam aterrorizados com a possibilidade de perda de emprego, causados pelos impactos que os robôs exercem sobre tal nível. Certamente os robôs se instalaram no lugar dos homens, muitas vezes um robô substitui dezenas ou até centenas de homens em uma linha de produção.

Este temor de desemprego vem aumentando a cada dia que passa. A queda nos custos dos robôs, tornando-os acessíveis para muitos setores da indústria, fez com que eles (os robôs) pudessem competir com a mão de obra barata, como a existente nos países do terceiro mundo, ameaçando o emprego de muitos trabalhadores. Muitas empresas multinacionais, que se instalavam em países subdesenvolvidos para utilizar-se do recurso “mão de obra barata”, já estão pensando em reverter essa tendência e concentrar suas operações nos seus próprios países de origem, utilizando robôs para baratear seus custos.

O uso de robôs para as indústrias passa a ser uma questão de sobrevivência, assim, resistir ao uso dos robôs é uma batalha perdida, principalmente devido à forma acelerada com que eles caem de preço. Além disso, o sucesso que as empresas e países usuários de robôs vêm obtendo é alto. O Japão, por exemplo, em 10 anos conseguiu quadruplicar a sua produção de automóveis, mantendo praticamente a mesma força de trabalho.

Um estudo conduzido no Japão em 1983 mostrou que existiam no país no início de 1981, cerca de 25000 robôs, cujo valor médio de mercado era de US\$ 17000. Desses robôs espera-se uma vida útil de seis anos, desde que trabalhem 22 horas por dia, durante os sete dias da semana. Fazendo-se as contas, percebe-se que nesses seis anos o robô trabalhará cerca de 48000 horas. Isso equivale ao que o operário médio japonês consegue trabalhar em 30 anos, já que trabalha apenas 40 horas por semana. O custo do operário médio para as empresas japonesas é de US\$ 13000 por ano. Pode-se notar a vantagem que os robôs possuem sobre os operários.

Quando se fala em desemprego, é necessário ressaltar que não existem somente os empregos destruídos. Existem também os empregos modificados. Habilidades pacientemente adquiridas por trabalhadores são, para alguns, bruscamente desqualificadas, porque foram tornadas inúteis pelo movimento do braço do robô.

Não resta dúvida que o que deve ser feito não é impedir o advento dos robôs, pois isto seria praticamente impossível. Por outro lado, não se deve assistir passivamente à sua chegada. O caminho é lutar para que sejam implantadas medidas que contraponham os seus possíveis impactos negativos.

Por exemplo, um estudioso inglês, chamado Tom Stoenier, diz que o caminho seria que os governos adotassem programas maciços de educação gratuita em todos os níveis. Isto aumentaria a capacitação dos indivíduos, o que possibilitaria que descobrissem novas formas de utilização dos robôs e dos computadores e, consequentemente, novos serviços e produtos viessem a ser inventados ou gerados.

Segundo um outro estudo feito por uma entidade sindical inglesa, a APEX, só existe uma forma não de se evitar desemprego, mas de se atenuar os seus males. Essa saída é através de uma atuação conjunta entre governo, sindicatos e empresários no sentido de estudarem cada setor da economia e cada ramo de atividade, estabelecendo impostos para as empresas que obtiverem ganhos em produtividade conseguidos às custas da automatização de funções outrora manuais. Esses impostos teriam uma destinação específica, qual seja a criação de empregos públicos em áreas como saúde, educação, etc.

Uma outra instituição muito importante que se tem dedicado a estudar este assunto é a OIT- Organização Internacional do Trabalho, organismo vinculado à ONU que já publicou várias pesquisas a respeito. As recomendações da Organização para que se consiga reduzir essas altas taxas de desemprego são as seguintes:

- 1) Reduzir as jornadas de trabalho para 30 horas semanais.
- 2) Criação de empregos no setor de serviços sociais, como saúde e educação.

Uma constante nos estudos referentes às possíveis forças atenuantes do desemprego é a referência à importância dos sindicatos de trabalhadores. Existem inúmeros exemplos provenientes de países com Alemanha, Suécia e França, entre outros da Europa Ocidental, onde atividades industriais foram automatizadas mas, por pressão dos sindicatos dos trabalhadores, as empresas não puderam dispensar seus empregados e tiveram de reaproveitá-los em outras funções. Isso evidencia uma outra arma importante no combate ao desemprego provocado pela máquina: o poder sindical, o qual, por sua vez, é dependente de uma sociedade civil forte. A existência de uma sociedade civil forte é importante não apenas para garantir sindicatos autônomos que possam defender seus filiados, mas também para permitir que toda essa discussão acerca da melhor distribuição dos frutos do progresso técnico seja feita de uma forma democrática, com participação de todos os segmentos da sociedade.

Uma outra forma muito importante para se conseguir atenuar os impactos negativos da robotização é o investimento maciço na indústria da construção civil, já que este é o segmento da economia com maiores condições de se transformar em absorvedouro de mão de obra não especializada. Além disso, a construção civil ainda demorará bastante até passar a ter os seus processos sendo realizados de forma robotizada ou automatizada. Ressaltando que quando se fala em construção civil não se está pensando somente em rodovias, pontes ou edifícios, mas principalmente em habitações populares, escolas e postos de saúde.

Um outro instrumento muito utilizado em praticamente todos os países desenvolvidos para atenuar os males do desemprego é o salário-desemprego.

1.6 – Conceitos Básicos de um Robô

1.6.1- Braço Mecânico

Um robô consiste em um braço mecânico motorizado, programável, que apresenta algumas características antropomórficas (figura 1.1) e um cérebro na forma de um computador que controla seus movimentos. O computador guarda em sua memória um programa que detalha o curso que o braço seguirá quando o programa estiver funcionando. O computador envia sinais ativando motores que movem o braço e a carga no final dele, que é mantida sob controle pelo atuador (“end effector”).

O braço mecânico é um manipulador projetado para realizar diferentes tarefas e ser capaz de repeti-las. Para realizar determinadas tarefas, o robô move partes, objetos, ferramentas e dispositivos especiais segundo movimentos e pontos pré-programados. A figura 1.2 mostra a estrutura de um robô industrial fazendo uma analogia com o ser humano.

Dois aspectos importantes do funcionamento de um braço mecânico correspondem ao sensoriamento ambiente e a como se realiza a programação do mesmo.

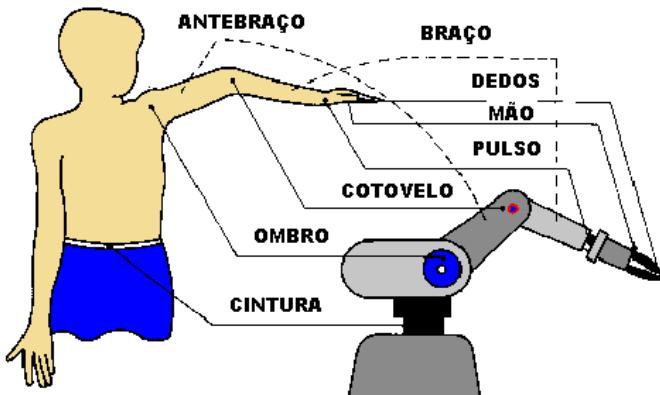


Figura 1.1: Conceitos Básicos de um robô.

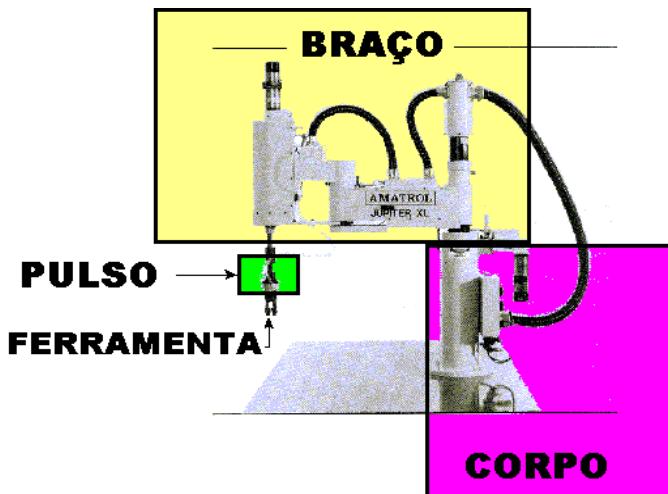


Figura 1.2: Estrutura de um robô em analogia com o ser humano.

1.6.2 - Sensoriamento e Programação

Para realizar certas tarefas os robôs precisam de habilidades sensoriais similares às do homem. Os modelos avançados de robôs estão equipados com sensores, mas sua capacidade ainda é limitada, assim como sua capacidade de movimentação, já que os robôs ficam fixos em um local, ou tem um espaço restrito para se mover. O controle de um robô deverá ser através da programação de um computador, onde o mesmo deverá possuir as seguintes características:

- Memória para guardar os programas;
- Conexões para os controladores dos motores;
- Conexões para entrada e saída de dados e para ativar os programas operacionais;
- Unidade de comunicação controlada por um humano.

1.6.3 – Aspectos relacionados à utilização de Robôs Industriais

1.6.3.1 - Classificação de um Sistema Automatizado

- Automação rígida: máquinas que são projetadas para executar uma função específica. Nestes sistemas, qualquer mudança na operação padrão demanda uma mudança no hardware da máquina e em sua configuração. Geralmente utilizados para um produto particular e de difícil adaptação a outro produto.
- Automação flexível: utilização de máquinas reprogramáveis com linguagem de programação estruturada, onde a configuração de uma célula de manufatura para outra pode ser alterada de modo fácil e rápido.

1.6.3.2 – Funcionalidades e Habilidades de um Robô

- **Robôs de 1ª Geração** - São incapazes de obter qualquer informação sobre o meio. Podem realizar apenas movimentos pré-programados e as informações que eles retornam sobre o ambiente de operação é mímina.
- **Robôs de 2ª Geração** - Possui todas as características da 1ª Geração, acrescentando uma detalhada comunicação com seu ambiente. Esta comunicação é atingida através de sistemas de sensoriamento e identificação. Necessita de computadores mais rápidos, com maior capacidade de memória e também de um grande avanço na capacidade de sensoriamento.

1.7 – Principais Aplicações de Robôs Industriais

Os robôs industriais têm fundamentalmente dois grandes grupos de aplicações, que obviamente se dividem em vários outros mais, que são [3]:

- a. Manipulação de materiais diversos;
- b. Fabricação

Em ambos os casos o robô industrial modifica seu ambiente, seja mudando as peças de lugar, seja criando um ambiente novo mediante a fabricação. Embora não se inclua a montagem de conjuntos mecânicos em tais grupos de aplicações, é evidente que a montagem constitui o topo do desenvolvimento tecnológico na indústria, e é conhecido que na fabricação de um produto a montagem ocupa 53% do tempo e, supõe-se, 22% do trabalho total.

1.7.1 – Utilização de Robôs Industriais no Brasil e no mundo

De acordo com dados da SOBRACON (Sociedade Brasileira de Comando Numérico, Automatização Industrial e Computação Gráfica), em 1995 existiam 550 robôs em operação nas indústrias brasileiras [8]. Dados de 1998 registram 1800 robôs, dos quais 65% estavam instalados na indústria automobilística brasileira [14]. No ano 2000 existiam cerca de 5000 robôs no Brasil [16], um número ainda muito baixo comparado com o de países industrializados (0,6% do total de robôs no mundo). A Asea Brown Boveri (ABB) detém em média de 33% do mercado brasileiro seguido da FANUC (18%), KUKA (13%) e robôs de outras marcas (36%) [15]. É importante notar o crescimento de aproximadamente 900% no número de robôs nas indústrias brasileiras nos últimos cinco anos, devido principalmente a investimentos privados realizados majoritariamente pelas indústrias automobilísticas [18].

As aplicações dos robôs nas indústrias brasileiras são diversas. Em termos percentuais, por exemplo, os robôs da ABB são utilizados para soldagem por resistência por pontos (33%), manipulação de materiais/paletização (25%), soldagem por arco (18%), pintura (10%) e outras aplicações como corte a jato de água, corte por gás, acabamento e montagem (14%) [17].

1.7.2 - Indústria Automobilística

Como mencionado anteriormente, quase 65% dos robôs existentes no Brasil em 1998 encontravam-se na indústria automobilística, sendo utilizados majoritariamente para realizar a soldagem por resistência por pontos. Outras aplicações dos robôs neste tipo de indústria são soldagem por arco, pintura, etc.

Muitos ainda veem aos robôs como os destruidores de empregos, entretanto, em 1996 a indústria automobilística da Ford, que possuía 120 robôs da ABB e produzia 1000 carros por dia, teve que “importar” 40 engenheiros ingleses e alemães para trabalhar com os outros 100 brasileiros [11]. Ou seja, tal indústria não gerou desemprego, já que era nova, e tampouco pôde criar mais do que os 100 postos de trabalho criados, porque na época não havia no Brasil suficientes profissionais capacitados para operar os robôs. A figura 1.3 mostra um exemplo de aplicação robótica na indústria automobilística.

O índice de automatização da indústria automobilística é de 90% em geral, embora em algumas tarefas de produção se chegue aos 100%. Como exemplo de comparação, o carro japonês Charade Sedan, da Daihatsu, que cerca de 80% dele é realizado através de robôs e 20% por trabalhadores japoneses.

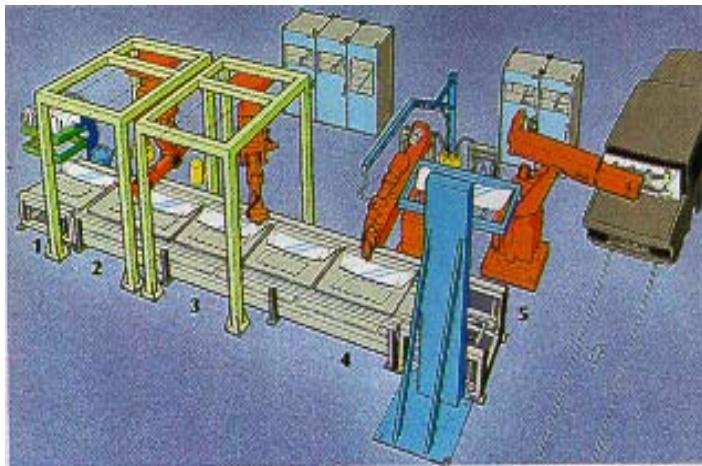


Figura 1.3: Exemplo de Célula Robotizada de Montagem de Veículos.

1.7.3 - Formação de Profissionais em Robótica no Brasil

Com o propósito de criar mão de obra especializada em robótica, existem vários grupos de pesquisa vinculados a Centros de Pesquisa e Universidades brasileiras. Em termos de preparação de mão de obra para atuar em sistemas robótizados, existe o SENAI (Serviço Nacional de Aprendizagem Industrial) que oferece cursos de formação profissional na área de robótica. Estudantes de nível básico de algumas escolas têm seu primeiro contato com robôs utilizando kits de robôs em cursos de robótica pedagógica.

A ABB, líder de robôs na Europa e EUA e com 60% do mercado de robôs de Brasil, também contribuem para a formação de profissionais nesta área, pois criou o primeiro centro de treinamento em automatização e robótica da América Latina, com aulas teóricas e práticas [10].

1.8 - Conclusão

Os principais fatores de crescimento do uso de robôs na indústria são motivados pelo aumento do custo da mão de obra, pelo aumento da produtividade e qualidade, pela melhoria das condições de segurança e qualidade de vida na realização de tarefas perigosas, além da queda do custo dos robôs. Estudos realizados pela ABB Robotics [12] mostram que em um ano pode-se obter o retorno do investimento realizado em robôs, já que o custo da mão de obra cresce cerca de 5% ao ano, enquanto que o custo dos robôs decresce mais que 5% ao ano. Um robô de soldagem utilizado na indústria automobilística, que em 1994 custava US\$ 200.000,00, atualmente custa cerca de US\$ 30.000,00 [14]. É importante destacar que se devem agregar ao preço final custos relativos à instalação, configuração, treinamento e testes do robô (cerca de US\$ 12.000,00).

A robótica do futuro constitui uma matéria multidisciplinar, que requer conhecimentos provenientes de diversos campos: projeto mecânico, eletrônica de potência, integração de grande escala e engenharia de software, e continuará sendo influenciada pelos avanços em acionamentos, controle, mecanismos, programação e sensores [3].

De acordo com alguns pesquisadores [3,7], o desafio tecnológico está na montagem de conjuntos de alto valor agregado, de uma forma econômica e mediante o emprego de sensores diversos. Isto supõe resolver problemas que até hoje não estão completamente resolvidos, como a integração multisensorial, a aprendizagem, o emprego cooperativo de sistemas multirrobôs, a adaptação às condições do ambiente, etc.

1.9 - Referências Bibliográficas

Scheinman, V., "Ideas on Implementing Modular Robot Systems", Technical Paper of Advanced Cybernetics Group, Inc., <http://www.advancedcybernetics.com/bmodula.htm>, 17/6/1998.

Nof, S. Y., *Handbook of Industrial Robotics*, John Wiley & Sons, 1985.

Armada, M.A., Control, de Robots, XV Curso de Automática en la Industria, Aguadulce (Almería), Junio de 1995.

Introducing Robotics, Technical Specifications of Pegasus 11 Articulated Servo Robot System, Edacom Tecnologia, São Caetano do Sul, Brasil.

Bastos, TT., Seguimiento y Análisis de Entornos de Soldadura por Arco Automatizada Mediante Ultrasonidos, Tesis Doctoral, Universidad Complutense de Madrid, Espanha, 1994.

Basañez, L., "MultiSensor Integration in Robotics", Workshop on Robotics and CIM, Lisbon, Portugal, September 1315, 1989.

Spong, M.W., Vidyasagar, M., *Robot Dynamics and Control*, John Wiley & Sons, Inc., 1989.

"Eu, Robô", Jornal O Globo, 11/12/95, Rio de Janeiro, Brasil.

Engelberger, J.F., "Robotics in the 21th Century", Scientific American, September 1995.

"Cresce Uso de Robôs na Indústria Nacional", Jornal A Tribuna, 02/01/1997, Vitória, Brasil.

"Adeus à Lanterna", IstoÉ, 27/3/1996.

"Conceito Empresarial ABB Robotics", ABINEE TEC'93, 1993.

“Delft Instruments Medical Imaging, 835 Sensor System. Template Description”, 1990.

“O Brasil na Era dos Robôs”, Época, 29/06/1998.

IFR International Federation of Robotics, 2000.

“Sales Representatives of Robot Manufacturers in Brazil (ABB, FANUC, KUKA)”, 2000.

ABB International Report, 2000.

Romano, V. F., “Brazilian Investments and Applications in Robotics”, ASI’2000 & IIMB’2000, Bordeaux, France, 20 September 2000.

CAPÍTULO 2

Aspectos Construtivos



CAPÍTULO 2

Aspectos Construtivos

Um robô é um sistema mecânico, de geometria variada, formada por corpos rígidos, articulados entre si, destinados a sustentar e posicionar/orientar o órgão terminal, que, dotado de garra mecânica ou ferramenta especializada, fica em contato direto com o processo. A mobilidade do manipulador é o resultado de uma série de movimentos elementares, independentes entre si, denominados graus de liberdade do robô [3].

Neste capítulo são apresentados aspectos construtivos de manipuladores robóticos, abordando configurações industriais básicas, mecanismos de manipulação e preensão de objetos e aplicações, enfatizando o estudo do braço mecânico do robô, seus tipos de juntas e graus de liberdade, tipos de articulações, sua área de trabalho (workspace) e formas de acionamento.

2.1 - Sistemas Robóticos

Um sistema robótico consiste normalmente nos seguintes elementos: manipulador (sistema mecânico), efetuador final, sistema de acionamento e controle de atuadores, computador e um sistema de visão ou outra classe de sensor.

O manipulador mecânico é constituído de vários links interligados através de articulações. Na figura 2.1 é apresentado um manipulador robótico do tipo serial onde o link da base é fixo e as articulações entre os links são atuadas. Ele possui as seguintes características:

- Um link é fixo a um sistema de referência, enquanto os outros são designados como links de saída.
- Algumas articulações no manipulador são atuadas, outras são passivas.
- O número de articulações atuadas é igual ao número de graus de liberdade do manipulador.

O efetuador final é um dispositivo adicionado ao link de saída do manipulador para apreensão, transporte ou manipulação de peças de trabalho. O efetuador final pode ser visto como uma interface entre o manipulador mecânico e o ambiente de atuação do manipulador.

O controlador pode ser constituído de um simples controlador PID de baixo nível a um controlador de alto nível baseado no modelo de um sistema de controle inteligente. Encoders são instalados nas juntas do manipulador para medir o movimento relativo entre dois corpos ligados, permitindo, assim, a implementação de um sistema de controle realimentado independente para cada junta do manipulador.



Figura 2.1: Elementos de um Sistema Robótico.

O braço do robô executa movimentos no espaço, transferindo objetos e ferramentas de um ponto para outro, instruído pelo controlador e informado sobre o ambiente por sensores. Na extremidade do braço existe um atuador usado pelo robô na execução de suas tarefas. Todo braço de robô é composto de uma série de vínculos e juntas, onde a junta conecta dois vínculos permitindo o movimento relativo entre eles, como mostrado na Figura 2.2. Todo robô possui uma base fixa e o primeiro vínculo está preso a esta base. A mobilidade dos robôs depende do número de vínculos e articulações que o mesmo possui.

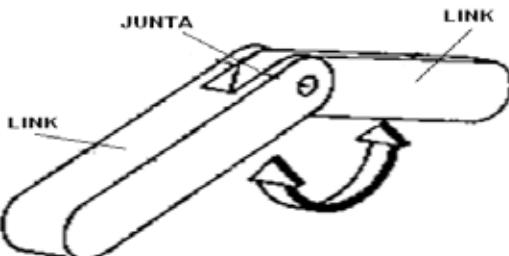


Figura 2.2: Junta e vínculos em um braço de robô.

2.1.1 - Juntas Robóticas

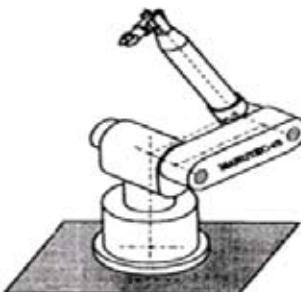
O braço manipulador de um robô é capaz de se mover para várias posições porque possuem uniões ou juntas, também denominadas eixos, que permitem ao manipulador executar tarefas diversas. O movimento da junta de um robô pode ser linear ou rotacional. O número de juntas de um robô determina seus graus de liberdade; a maioria dos robôs possui de três a seis eixos.

Estes eixos podem ser divididos em duas classes: eixo do corpo e eixo da extremidade do robô. Os eixos da base do corpo do robô permitem mover seu órgão terminal para uma determinada posição no espaço. Estes eixos são denominados cintura, ombro e cotovelo (*wrist, shoulder e elbow*). Os eixos da extremidade do robô permitem orientar seu órgão terminal e são denominados *roll, pitch e yaw* (figura 2.3).

Um robô industrial possui seis graus de liberdade, três para posicionamento da ferramenta terminal e três para orientação, compatível com tarefas que sejam realizadas dentro de seu volume de trabalho; com menos de seis graus de liberdade não se alcançam todos os pontos de um ambiente de trabalho. Um robô com mais de seis juntas é denominado robô redundante, ou seja, tem mais graus de liberdade do que o mínimo requerido para a execução da tarefa. No próximo capítulo serão apresentados aspectos referentes à modelagem cinemática de manipuladores robóticos.



a) Robô Stäubli 90X



b) Robô Manutec Siemens

Figura 2.3: Exemplo de Robôs Industriais.

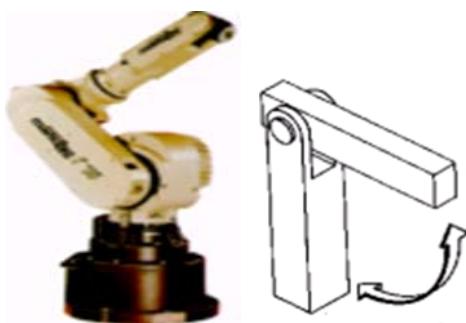
2.1.2 - Tipos de Juntas

Os braços robóticos podem ser constituídos de juntas:

- a. **Deslizantes** (figura 2.4a): Este tipo de junta permite o movimento linear entre dois vínculos. É composto de dois vínculos alinhados um dentro do outro, onde um vínculo interno escorrega pelo externo, dando origem ao movimento linear.
- b. **Rotativas** (figura 2.4b): Esta conexão permite movimentos de rotação entre dois vínculos. Os dois vínculos são unidos através de dobradiça comum, com uma parte podendo se mover num movimento cadenciado em relação à outra parte. As juntas rotativas são utilizadas em diversas ferramentas e dispositivos, tais como tesouras, limpadores de para-brisa e quebra-nozes.



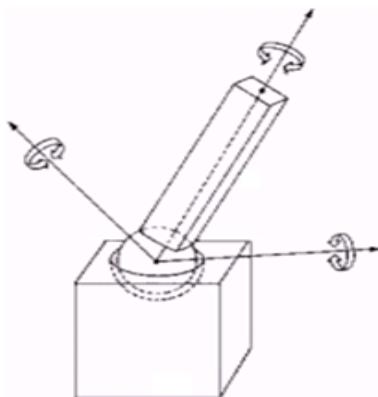
a) Juntas Deslizantes – Pórtico Cartesiano.



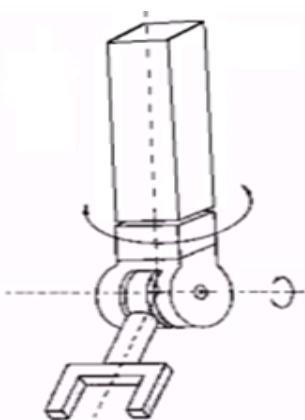
b) Junta Rotativa – Robô Industrial 6R.

Figura 2.4: Tipo de juntas robóticas e Aplicações.

- c) **Bola-encaixe** (figura 2.5): Esta conexão se comporta como uma combinação de três juntas de rotação, permitindo movimentos de rotação em torno dos três eixos (figura 2.5a). Para se ter o mesmo efeito de uma junta bola e encaixe, muitos robôs incluem três juntas rotacionais separadas, cujos eixos de movimentação se cruzam em um ponto (figura 2.5b).



a) Bola Encaixe.



b) Rotacional (3R).

Figura 2.5: Junta de bola-encaixe.

A maior parte dos braços robóticos é interligada através de juntas deslizantes e rotacionais, embora alguns incluam o de bola-encaixe, pouco utilizada em robótica, devido à dificuldade de acionamento.

2.1.3 - Graus de Liberdade

O número de articulações em um braço robótico é geralmente associado ao seu número de graus de liberdade. Quando o movimento relativo ocorre em um único eixo, a articulação tem um grau de liberdade. Quando o movimento é por mais de um eixo, a articulação tem dois graus de liberdade. A maioria dos robôs industriais tem entre 4 a 6 graus de liberdade, enquanto um ser humano tem sete graus de liberdade do ombro até o pulso.

2.2 - Classificação de Manipuladores Robóticos

Os manipuladores robóticos podem ser classificados dependendo de vários critérios, dentre eles, o número de graus de liberdade, estrutura cinemática, geometria do espaço de trabalho e características do movimento, tipo de controle e acionamento [4]. É usual classificar os robôs de acordo com o tipo de junta, ou mais exatamente pelas três juntas mais próximas da base do robô. Os robôs podem também ser classificados em relação ao espaço de trabalho (workspace), grau de rigidez, extensão de controle sobre o curso do movimento e através das aplicações adequadas ou inadequadas para cada tipo de robô.

2.2.1 - Estrutura Cinemática

Um manipulador pode ser classificado de acordo com a forma de sua estrutura cinemática, ou seja:

- a) **Serial ou manipulador de cadeia aberta:** possui a forma de cadeia cinemática aberta (figura 2.6b).
- b) **Paralelo:** constituído de cadeias cinemáticas fechadas (figura 2.6a).

- c) **Híbrido:** constituído de cadeias cinemáticas abertas e fechadas (figura 2.7).



a) Manipulador Paralelo - Plataforma de Stewart



b) Manipulador Serial KR-3 KS (KukaTM)

Figura 2.6: Manipulador Paralelo e Serial.

A figura 2.7 apresenta um manipulador robótico desenvolvido na Robotics Research Center do Nanyang Technological University. Este manipulador possui uma base composta de uma estrutura paralela com três graus de liberdade e um braço robótico com quatro graus de liberdade disposto sobre esta base.



Figura 2.7: Manipulador Híbrido.

2.2.2 - Geometria do Robô

Os diferentes graus de liberdade de um robô podem ser encontrados em várias combinações de configurações rotacionais e lineares, dependendo da aplicação. Estas combinações são denominadas geometria do robô.

Existem cinco classes principais de robôs manipuladores, segundo o tipo de juntas de rotação ou de revolução (R) ou de translação ou prismáticas (P), permitindo diferentes possibilidades de posicionamento no volume de trabalho.

As cinco classes ou geometrias principais de um robô são: cartesiana, cilíndrica, esférica (ou polar), de revolução (ou articulada) e SCARA (Selective Compliant Articulated Robot for Assembly), que também são denominados *sistemas geométricos coordenados*, posto que descrevem o tipo de movimento que o robô executa.

O código usado para estas classificações consiste em três letras, referindo-se ao tipo de junta (R = revolução, P = deslizante ou prismática) na ordem em que ocorrem, começando de junta mais próxima à base.

2.2.2.1 - Robô de Coordenadas Cartesianas

Um robô de coordenadas cartesianas, ou robô cartesiano (figura 2.8), pode se movimentar em linhas retas, em deslocamentos horizontais e verticais. As coordenadas cartesianas especificam um ponto do espaço em função de suas coordenadas X, Y e Z. Estes robôs têm três articulações deslizantes, sendo codificados como PPP.

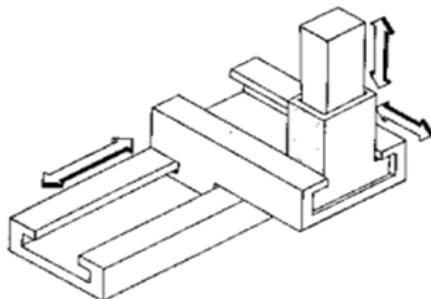


Figura 2.8: Eixos de um robô cartesiano
(PrismáticoPrismáticoPrismático - PPP).

Os robôs cartesianos se caracterizam pela pequena área de trabalho, mas com um elevado grau de rigidez mecânica e são capazes de grande exatidão na localização do atuador. Seu controle é simples devido ao movimento linear dos vínculos e devido ao momento de inércia da carga ser fixo por toda a área de atuação.

2.2.2.2 - Robô de Coordenadas Cilíndricas

Um robô de coordenadas cilíndricas combina movimentos lineares com movimentos rotacionais, descrevendo um movimento final em torno de um envelope cilíndrico. Normalmente, este tipo de robô possui um movimento rotacional na cintura (wrist) e movimentos lineares. Os graus de liberdade deste robô consistem de uma junta de revolução e duas juntas deslizantes, sendo codificado como RPP, como mostra a figura 2.6.

A área de trabalho destes robôs é maior que a dos robôs cartesianos, mas a rigidez mecânica é ligeiramente inferior. O controle é um pouco mais complicado que o modelo cartesiano, devido a vários momentos de inércia para diferentes pontos na área de trabalho e pela rotação da junta da base.

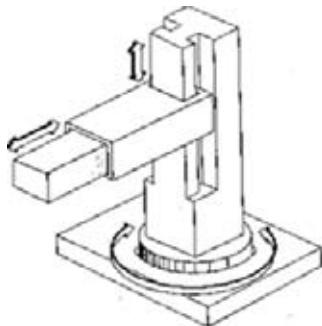


Figura 2.6: Eixos de um robô de coordenadas cilíndricas (RPP).

2.2.2.3 - Robô de Coordenadas Polares (Esféricas)

Um robô de coordenadas polares ou esféricas possui dois movimentos que são rotacionais, na cintura e no ombro (waist e shoulder), e um terceiro movimento que é linear. Estes três eixos descrevem um envelope esférico. Estes robôs possuem duas juntas de revolução e uma deslizante, sendo codificado como RRP, como na figura 2.7a.

Estes robôs têm uma área de trabalho maior que os modelos cilíndricos, mas perde na rigidez mecânica. Seu controle é ainda mais complicado devido os movimentos de rotação.

2.2.2.4 - Robô de Coordenadas de Revolução (Articulado)

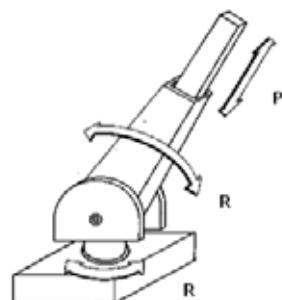
Um robô de coordenadas de revolução (articulado) possui juntas e movimentos que se assemelham aos de um braço humano. O robô PUMA (Programmable Universal Machine for Assembly) é um dos projetos mais populares de robôs articulados e foi projetado inicialmente para cumprir com os requerimentos da indústria automobilística. Estes robôs caracterizam-se por possuir três juntas de revolução, sendo codificados por RRR, como mostra a figura 2.7b.

Um robô com juntas de revolução tem uma área de atuação maior que qualquer tipo de robô, tendo uma baixa rigidez mecânica. Seu controle é complicado e difícil, devido às três juntas de revolução e devido a variações no momento de carga e momento de inércia.

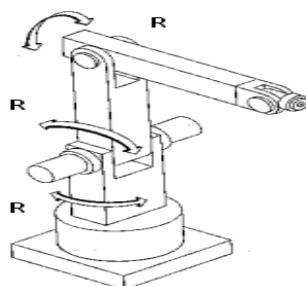
2.2.2.5 - Robô SCARA

O robô SCARA é uma configuração recente utilizada para tarefas de montagem, como o seu nome sugere, caracterizando-se por possuir duas juntas de revolução e uma deslizante, sendo codificado RRP. Embora a configuração pos-

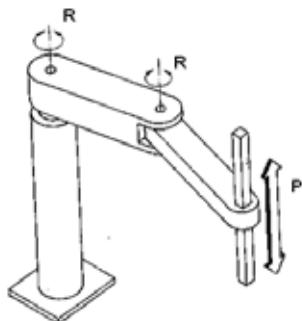
sua os mesmos tipos de juntas que uma configuração esférica (Rotacional-Rotacional-Prismática - RRP), ela se diferencia da esférica tanto pela sua aparência quanto pela sua faixa de aplicação, como mostra a figura 2.7c.



a) RRP



b) RRR



c) RRP

Figura 2.7: Eixos de coordenadas de Robôs.

A área de atuação deste tipo de robô é menor que no modelo esférico, sendo apropriado para operações de montagem, devido ao movimento linear vertical do terceiro eixo.

2.2.3 - Estudo Comparativo da Área de Trabalho para diferentes configurações de robô

A análise matemática elementar para cálculo da capacidade dos robôs é apresentada na figura 2.8. Um estudo comparativo de diferentes configurações do robô é resumido na tabela 2.1.

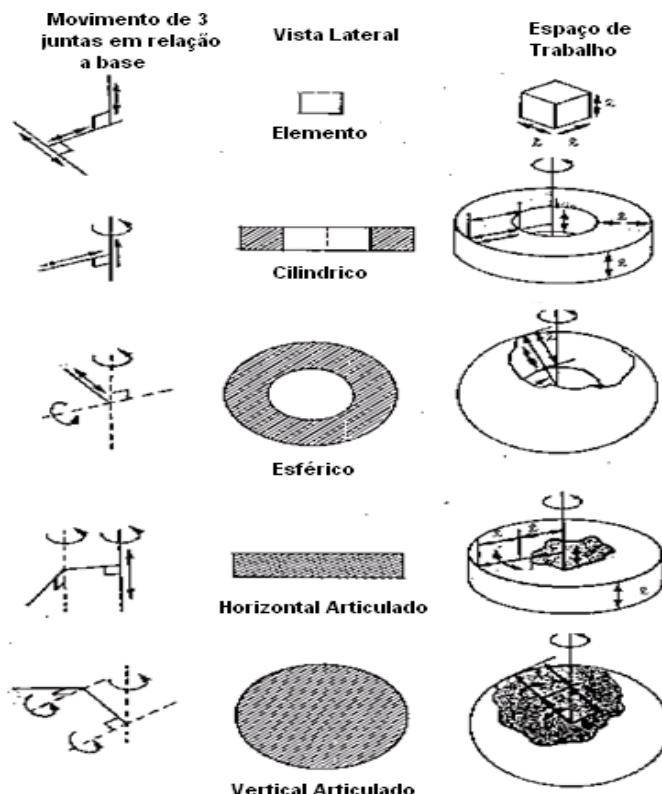


Figura 2.8: Comparação da Área de Trabalho dos tipos de configurações de robôs.

Tipo de Configuração das Juntas Robóticas	Capacidade de Área de Trabalho
Cartesiano	Alcançam qualquer ponto de um cubo de lado L
Cilíndrico	Alcançam qualquer ponto em um cilindro de altura L e raio $2L$, exceto os pontos do cilindro interno de raio L e altura L
Esférico	Alcançam qualquer ponto de uma esfera de raio $2L$, exceto a esfera interna de raio L
Articulado	Alcançam qualquer ponto de um cilindro de raio $2L$ e altura L
SCARA	Alcançam qualquer ponto de uma esfera de raio $2L$

Tabela 2.1: Comparação da Área de Trabalho

dos tipos de configurações de robôs.

2.2.4 - Avaliação de configurações robóticas na capacidade de realização de tarefas

A avaliação dos tipos de articulações e seu arranjo permitem ao projetista estimar a área de atuação do robô, rigidez mecânica e facilidade de controle do braço, possibilitando qual tarefa será mais adequada para cada tipo de robô.

O movimento das articulações capacita o robô a mover seu atuador para qualquer ponto na sua área de atuação, mas não habilitando o controle da orientação do atuador no espaço; cuja importância não se restringe somente ao alcance da peça, mas também em conduzir o atuador a certa altitude em relação à peça. Essa tarefa pode ser realizada adicionando-se articulações para o pulso do braço, dando um maior grau de liberdade. A partir disso, o robô fica habilitado a realizar os seguintes movimentos (figura 2.9):

- **Roll (R)** : Movimento de rotação no sentido horário e anti-horário.
- **Pitch (P)** : Movimento para cima e para baixo.
- **Yaw (Y)** : Movimento para a esquerda e para a direita.

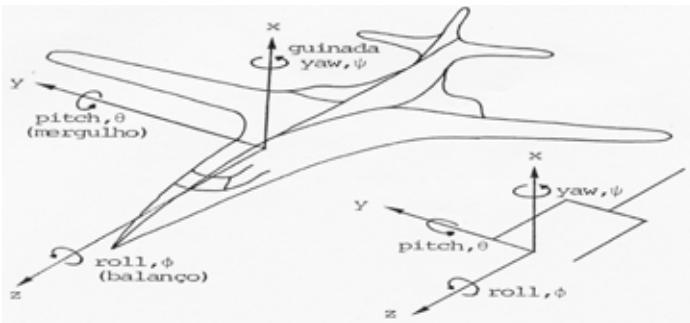


Figura 2.9: Definição dos ângulos de orientação RPY.

2.2.5 - Construção de Vínculos

Um importante fator na construção dos vínculos é a carga que o mesmo suporta, o peso do próprio braço e o grau de rigidez do mesmo. Um braço pesado necessita de um motor maior, tornando o custo do robô mais elevado. Um braço de baixa rigidez reduz a precisão do robô devido às vibrações e resposta à tensão. Para aumentar a rigidez mecânica do braço sem aumentar seu peso, frequentemente usa-se uma estrutura oca. A utilização deste tipo de estrutura tem uma maior rigidez mecânica quando comparada com uma construção maciça utilizando a mesma massa de material. Essa comparação é mostrada na figura 2.10.

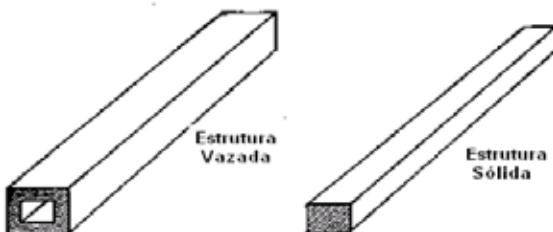


Figura 2.10: Estruturas para a construção de vínculos.

2.3 - Sensores

O uso de sensores permite que um robô possa obter informação sobre o comportamento do mesmo e sobre seu ambiente de atuação com a finalidade de realizar a operação para o qual foi destinado, além de poder modificá-la. Os sensores podem ser agrupados em duas categorias principais: sensores internos ou proprioceptivos, que fornecem informações sobre as variáveis do próprio robô, e sensores externos ou exteroceptivos, cujo objetivo é obter informação do ambiente ao redor do robô.

A maior parte dos robôs industriais é do tipo convencional, também chamado de robô de primeira geração, que desenvolve tarefas pré-programadas repetitivas, necessitando unicamente de sensores internos (proprioceptivos), situados nas juntas do robô, para a realização das tarefas.

Estes sensores podem ser codificadores óticos (*encoders*) do tipo incremental ou absoluto, sincros, resolvers, potenciômetros multi-voltas, tacômetros, etc. Entre estes, os codificadores óticos incrementais estão entre os sensores mais utilizados, devido a seu baixo custo e por proporcionar uma precisão suficiente para a maioria das aplicações.

Encoders são dispositivos eletromecânicos que convertem a rotação angular do eixo do robô em pulsos de saída na forma de ondas quadradas. Assim, pode-se conhecer o sentido de rotação, a posição e a velocidade de todos os eixos do robô. A figura 2.11 detalha este tipo de sensor.

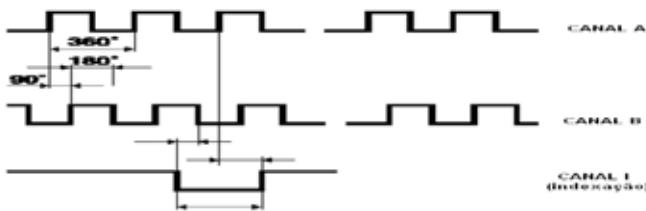
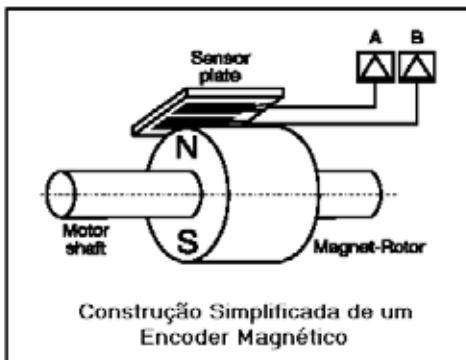
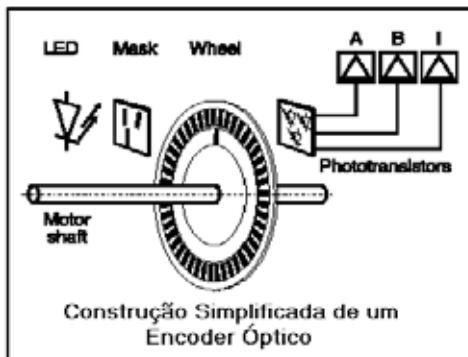


Figura 2.11: Encoders Incrementais – Princípio de Funcionamento.

Entretanto, embora estes sensores possam proporcionar informação sobre o estado interno do robô, não possuem informação do ambiente externo ao robô. Tal falta de informação do ambiente de trabalho restringe as aplicações a casos particulares onde as peças objeto da tarefa estão posicionadas

dentro das tolerâncias operacionais do robô. Este é o caso típico da indústria automobilística, que utiliza soldagem por resistência, em que as peças a processar são extremamente precisas ou apresentam características acomodatícias, o que permite o emprego de sistemas de sujeição para fixar suas dimensões. Entretanto, em várias aplicações de robôs na indústria, é difícil realizar um sistema de sujeição eficaz [5].

O aparecimento da nova geração de robôs permitiu interconectar sensores exteroceptivos com o controlador do robô, podendo utilizar tais sensores para obter propriedades relevantes do ambiente e controlar o sistema na realização da tarefa. O emprego de sensores externos possibilita um controle em malha fechada que permite a realização de aplicações complexas, tais como agarrar objetos aleatoriamente posicionados e orientados, seguir objetos em movimento em um ambiente 3D, realizar a montagem de dispositivos mecânicos, realizarem a inspeção/controle de qualidade de peças, efetuarem a busca e identificação de objetos, além de se poder ter um controle preciso do manipulador [3,6].

Os sensores externos, normalmente encontrados nos robôs industriais, são sensores de segurança para proteção humana (cortinas de luz, ultrassom, barreiras mecânicas, dispositivos sensíveis à pressão, etc.), sensores de contato, sensores óticos para determinação de distância a obstáculos e reconhecimento de objetos, sensores indutivos, capacitivos, de efeito Hall, ultrassônicos e laser para determinação de proximidade às peças a manipular.

2.4 - Acionamento e Controle de Robôs

Os eixos de um robô são acionados por atuadores. Um atuador converte algum tipo de energia em movimento mecânico. Os três tipos de energia mais comuns para acionar os atuadores de um robô são Pneumático, Elétrico e Hidráulico.

Os atuadores hidráulicos possuem alto torque e velocidade de resposta, sendo adequados para atuar sobre cargas

pesadas. Entretanto, requer equipamentos periféricos, como bombas, o que implica na necessidade de manutenção frequente, além de gerar grande ruído. Os atuadores pneumáticos são mais baratos e simples, entretanto, não podem ser controlados com precisão. Atualmente os motores elétricos, CA ou CC, são os mais atrativos para ser empregados em robótica, pois são mais baratos e silenciosos [8].

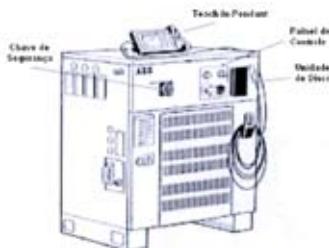
O controle dos atuadores dos robôs normalmente é realizado através de dois métodos: servo-controlados ou não servo-controlados. Um robô não servo-controlado utiliza chaves mecânicas no final do curso de cada junta (*limit switches*), de modo que o controlador unicamente conhece quando um atuador está em uma das duas posições: iniciando ou finalizando seu movimento. O posicionamento de cada eixo é controlado por paradas mecânicas ajustáveis em lugar de ser pelo controlador. Este tipo de robô é de baixo custo e pode realizar tarefas de sequência fixa ou variável, por exemplo, pegar objetos e colocá-los em um determinado lugar (*pick and place*).



a) Ferramentas e Acessórios.



b) Teach-in-Pendant.



c) Unidade de Controle.

Figura 2.12: Principais Componentes de um Robô.

Por outro lado, os robôs servo-controlados, ao utilizarem sensores internos (ou proprioceptivos) podem conhecer a posição na qual se encontra cada eixo, além de sua velocidade. Por sua vez, o controlador pode controlar a quantidade de energia a ser fornecida aos atuadores com a finalidade de permitir ao robô mover com velocidade variável e parar em qualquer posição. Estes robôs podem realizar tarefas através do modo ensino e repetição (*playback*) ou por programação de alto nível. A figura 2.12 mostra os principais componentes de um robô servo-controlado, composto pelo robô, módulo de controle (computador e software), os dispositivos de acionamento de potência, placas de entrada/saída de dados e os periféricos (vídeo, unidade de disco e “comando manual”).

O software que controla o robô pode ser digitado diretamente no teclado do módulo de controle, utilizando uma das várias linguagens de programação de robôs (por exemplo, para o PUMA utiliza-se o VAL II, que também é um sistema operacional) [5]. A figura 2.13 mostra uma ilustração de uma célula automatizada de usinagem.

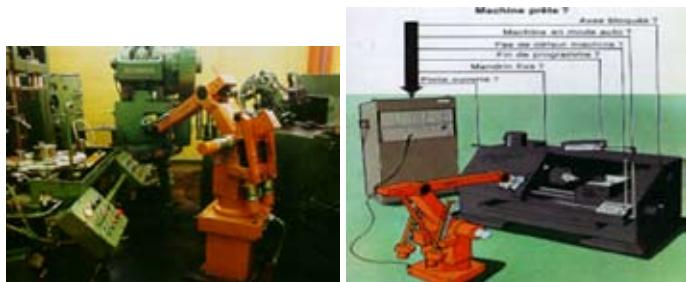


Figura 2.13: Célula Automatizada de usinagem.

2.4.1 - Tipos de Acionamento de um braço robótico

Existem diversos tipos de *drivers* de açãoamento de braços robóticos que são classificados genericamente através do:

- **Movimento:** Drivers de Rotação e de Deslizamento;
- **Acionamento:** Drivers Elétrico, Hidráulico, Pneumático;
- **Tipo de conexão:** Drivers Direto e Indireto.

O sistema de açãoamento pode ser classificado pela forma movimento em:

- a) **Driver de rotação:** Consiste em um motor, que quando conectado à sua fonte de energia, o eixo do motor responde em um movimento de rotação.
- b) **Driver deslizante:** Consiste em um cilindro hidráulico ou pneumático. O movimento linear também pode ser produzido por um movimento rotativo usando correias ou hastes empurradas pelo motor, fazendo uma conversão de movimento rotativo em linear.

2.4.2 - Formas de Acionamento de um braço robótico

O sistema de açãoamento pode ser classificado pela sua forma de transformação de energia em Açãoamento Elétrico, Hidráulico e Pneumático.

2.4.2.1 - Açãoamento Elétrico

Este tipo de açãoamento utiliza motores elétricos que podem ser: motor de corrente contínua, motor de passo e motor de corrente alternada. Muitos robôs novos têm drivers de motor corrente contínua devido ao alto grau de precisão e simplicidade

de controle do motor elétrico. A figura 2.14 mostra o acionamento completo referente a uma junta de um robô industrial e suas principais vantagens e desvantagens de utilização.

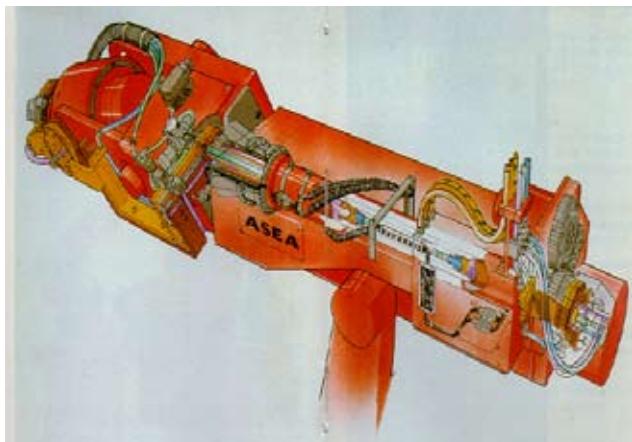


Figura 2.14: Sistema de Acionamento Elétrico de uma junta robótica.

VANTAGENS

1. **Eficiência calculada, controle preciso;**
2. **Envolve uma estrutura simples e fácil manutenção;**
3. **Não requer uma fonte de energia cara;**
4. **Custo relativamente pequeno.**

DESVANTAGENS

1. **Não pode manter um momento constante nas mudanças de velocidade de rotação.**
2. **Sujeitos a danos para cargas pesadas suficientes para parar o motor.**
3. **Baixa razão de potência de saída do motor e seu peso, necessitando um motor grande no braço.**

2.4.2.2 - Acionamento Hidráulico

Este tipo de acionamento utiliza uma Unidade Hidráulica, composta dos seguintes elementos: motor de movimento rotativo e cilindro para movimento deslizante. A unidade de acionamento hidráulico provoca movimento em pistões que comprimem o óleo. O controle é feito através de válvulas que regulam a pressão do óleo nas duas partes do cilindro e que impulsionam o pistão. A Figura 2.15 apresenta uma unidade de acionamento hidráulica e principais vantagens e desvantagens na utilização desse tipo de acionamento.

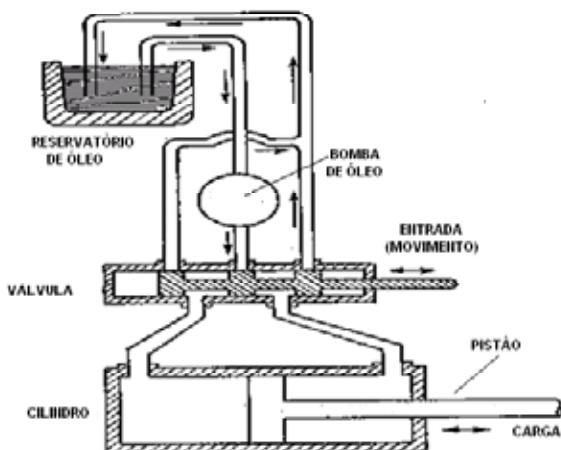


Figura 2.15: Unidade de acionamento hidráulico.

VANTAGENS

1. **Momento alto e constante sob uma grande faixa de variação de velocidade.**
2. **Precisão de operação (menor que o elétrico e maior que o pneumático).** O óleo não é compressível e não há variação de seu volume quando se varia a pressão.
3. **Pode manter um alto momento para um longo período de tempo, quando parado.**

DESVANTAGENS

- 1. Requer uma fonte de energia cara.**
- 2. Requer uma manutenção cara e intensa.**
- 3. Requer válvulas de precisão caras.**
- 4. Está sujeito a vazamento de óleo.**

2.4.2.3 - Acionamento Pneumático

Este tipo de acionamento é similar ao hidráulico e é composto de: motores pneumáticos de movimento rotativo e cilindros pneumáticos de movimento deslizante. Na Figura 2.17 pode-se considerar a mesma para acionamento pneumático, utilizando ar ao invés de óleo. Possui um alto grau de precisão nas paradas. São utilizados em sistemas automáticos simples, mas pouco utilizado em robôs devido à alta compressibilidade, o que reduz a habilidade de realizar controle preciso. É muito utilizado em movimentos de preensão, tanto para abrir como para fechar as garras. A figura 2.16 apresenta exemplos de garras pneumáticas e principais vantagens e desvantagens desse tipo de acionamento.



Figura 2.16: Garras Robóticas de Acionamento Pneumático.

VANTAGENS

- 1. Podem operar em velocidades extremamente altas.**
- 2. Custo relativamente pequeno e fácil manutenção.**
- 3. Podem manter um momento constante em uma grande faixa de velocidade.**
- 4. Pode manter alto o momento por longos períodos de tempo sem danos, quando parado.**

DESVANTAGENS

- 1. Não possui alta precisão.**
- 2. Esta sujeito a vibrações quando o motor ou cilindro pneumático é parado.**

2.4.2.4 - Comparação de utilização de Acionamentos

A tabela 2.2 apresenta um quadro comparativo, resumindo vantagens e desvantagens de utilização dessas formas de acionamento.

Acionamento / Característica	Elétrico	Hidráulico	Pneumático
Acionamento / Característica	Alta	Média - Alta	Baixa
Capacidade Transferência de carga	Pequena e média (20 Kg)	Pesada (> 1000 Kg)	Pequena e média (20 Kg)
Velocidade	Alta	Média - Alta	Alta
Custo	Alta	Médio - Alto	Baixo

Tabela 2.2: Quadro Comparativo – Formas de Acionamento.

2.4.3 - Classificação pela forma de conexão: Direto vs Direto e Indireto

No caso do acionamento direto, o motor é montado diretamente na junta que ele irá mover. Se o motor é montado longe da junta, próximo da base, o acionamento é indireto; neste caso há elementos de transmissão como correntes, correias, diferenciais e engrenagens. Vantagens do acionamento indireto sobre o direto:

- Redução do peso do braço mecânico;
- Permite mudanças na velocidade de rotação das juntas.
- Desvantagens do driver indireto sobre o direto:
 - Falta de precisão da operação da junta devido à liberdade mecânica dos pontos de conexão entre os dispositivos de transferência;
 - Perdas consideráveis de potência.

2.4.4 – Atuadores

O atuador (end effector) é todo um sistema montado na extremidade do vínculo mais distante da base do robô, cuja tarefa é agarrar objetos, ferramentas e/ou transferi-las de um

lugar para outro. São exemplos de atuadores a pistola de solda, garras e pulverizadores de tintas. A operação do atuador é o objetivo final na operação de um robô, assim todos os demais sistemas (unidades drives, controles, etc.) são projetados para habilitar sua operação.

2.5 - Programação de Robôs

Para que os robôs possam realizar sua missão, é necessário programá-los. Um robô pode executar as tarefas por meio de programas realizados diretamente no computador que controla o robô, utilizando uma das várias linguagens de programação de robôs, ou utilizar o modo repetição “*playback*” para programá-lo, utilizando o “comando manual”.

Este método é utilizado para simplificar a programação dos robôs, pois quando o robô está no modo ensino (“*teach*”) pode-se movê-lo da forma desejada através das teclas do comando manual. Pode-se também editar programas, selecionar velocidades, mudar parâmetros da tarefa (por exemplo, inserir e/ou apagar parâmetros de soldagem), etc.

Uma vez programado, o robô repetirá automaticamente os movimentos entre os pontos gravados. Se um ponto necessita ser corrigido, pode-se executar o programa para trás passo-a-passo, alcançar o ponto desejado e corrigi-lo. Assim, o resultado é uma importante economia de tempo. Nos últimos anos, a programação dos robôs industriais evoluiu consideravelmente, e atualmente eles podem ser programados através de comando vocal, sistemas gráficos interativos, geração de planos de ação, realidade virtual, etc.

2.6 - Precisão e Repetibilidade

Dois importantes parâmetros característicos dos robôs são sua precisão e repetibilidade. Por precisão, entende-se a capacidade do robô de ir a uma posição desejada, com respeito a um sistema de referência fixo (normalmente a base do robô), com um erro determinado (por exemplo, ± 1 mm). Trata-se de precisão em posicionamento absoluto.

Repetibilidade pode ser entendida como a capacidade do robô de, uma vez conhecida e alcançada uma posição, e partindo-se da mesma condição inicial, voltar a ir (“repetir”) novamente a tal posição com um erro determinado. A maioria dos manuais dos robôs informa sobre a repetibilidade do robô e não a precisão absoluta, muito mais difícil de obter [3].

2.7 - Garras e Ferramentas

Os robôs são projetados para atuar sobre seu ambiente, mas para isto devem ir dotados em seu órgão terminais conforme mostra o quadro comparativo apresentado na tabela 2.3 e exemplificado na figura 2.17.

GARRAS E MÃOS MECÂNICAS

- Com sujeição por pressão
- Com sujeição magnética
- Com sujeição a vácuo
- Com sujeição de peças a temperaturas elevadas
- Resistentes a produtos corrosivos e perigosos
- Dotadas de sensores, etc.

FERRAMENTAS ESPECIALIZADAS

- Pistolas pulverizadoras (pintura, metalização)
- Soldagem por resistência por pontos
- Soldagem por arco
- Dispositivos de Furação
- Polidoras, etc.

Tabela 2.3: Quadro Comparativo – Garras e Ferramentas Especializadas.

A escolha de uma ferramenta é de extrema importância na execução de uma tarefa, entretanto isto exige que o mesmo seja adequadamente projetado e adaptado às condições do seu meio e área de trabalho, onde podemos classificar em garras e ferramentas específicas para cada tipo de aplicação.

A garra é comparável a mão humana, no entanto, ela não é capaz de simular seus movimentos, resultando na limitação dos movimentos a uma faixa de operações. A grande demanda tem levado ao desenvolvimento de garras que podem manusear objetos de diferentes tamanhos, formas e materiais. Estas garras são divididas em vários tipos de classe, descritos nos próximos itens.



a) Garra de Prensão.



b) Ferramenta Antropomórfica.



c) Exemplos de Ferramentas Dedicadas

Figura 2.17: Ferramentas acopladas ao órgão terminal de um robô.

2.7.1 - Garra de dois dedos

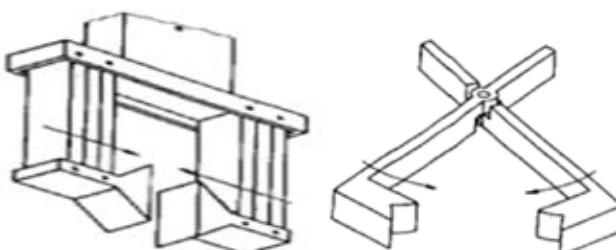
É o tipo mais comum e com grande variedade. São diferenciados um do outro pelo tamanho e/ou movimento dos dedos, como o movimento paralelo mostrado na figura 2.18a ou o movimento de rotação mostrado na figura 2.18b. A principal desvantagem desta garra é a limitação da abertura dos seus dedos, restringindo, assim, a sua operação em objetos cujo tamanho não exceda esta abertura máxima.

2.7.2 - Garra de três dedos

É similar às garras de dois dedos, porém permite uma preensão de objetos em forma circular, triangular e irregular com maior firmeza. Os dedos são articulados e formados por diversos vínculos, como mostra a figura 2.19a.

2.7.3 - Garra para preensão de objetos cilíndricos

Consiste de dois dedos com vários semicírculos chanfrados (figura 2.19b), que permitem à garra segurar objetos cilíndricos de vários diâmetros diferentes. As principais desvantagens deste tipo de garra são: seu peso que deve ser sustentado pelo robô durante a operação e a limitação de movimentos causada pelo comprimento da mesma.



a) Movimento Paralelo

b) Movimento de Rotação.

Figura 2.18: Garra de dois dedos - formas de movimentação de garras.

2.7.4 - Garra para objetos frágeis

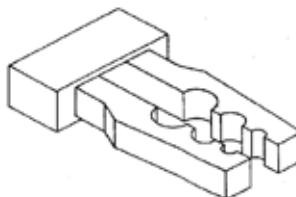
Garras própria para exercer certo grau de força durante a operação de segurar algum corpo, sem causar algum tipo de dano ao mesmo. Ela é formada por dois dedos flexíveis, que se curvam para dentro, de forma a agarrar um objeto frágil; seu controle é feito por um compressor de ar (figura 2.19c).

2.7.5 - Garra Articulada

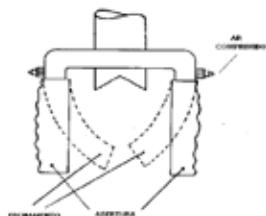
É projetada para agarrar objetos de diferentes tamanhos e formas. Os vínculos são movimentados por pares de cabos, em que um cabo flexiona a articulação e o outro a estende. Sua destreza em segurar objetos de formas irregulares e tamanhos diferentes se deve ao grande número de vínculo, conforme mostra a figura 2.19d abaixo.



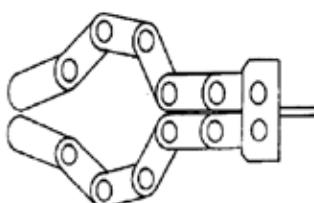
a) Três Dedos.



b) Objetos Cilíndricos.



c) Objetos Frágeis.



d) Garra Articulada.

Figura 2.19: Tipos de Garras de preensão de objetos.

2.7.6 - Garras a vácuo e acionamento eletromagnético

Garras a vácuo são projetadas para prender uma superfície lisa durante a ação do vácuo. Estas garras possuem ventosas de sucção conectadas a bomba de ar comprimido, que prendem superfícies como chapas metálicas e caixas de papelão. Para reduzir o risco de funcionamento irregular devido a perda de vácuo, é comum usar mais do que uma ventosa de sucção (figura 2.20).

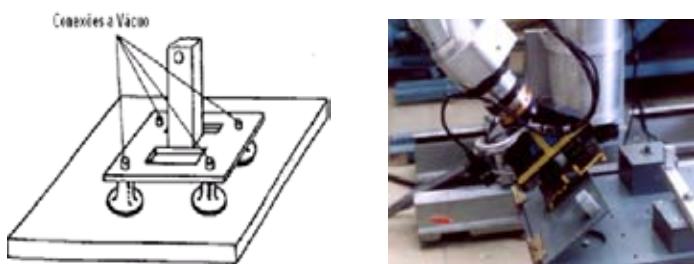


Figura 2.20: Garras a vácuo.

Garras eletromagnéticas são utilizadas para segurar objetos que podem ser magnetizados (aço e níquel) através de um campo magnético. Estes objetos devem possuir um lugar específico na qual a garra passa atuar. Ambos os tipos de garras descritos acima são muito eficientes, uma vez que eles podem segurar objetos de vários tamanhos e não necessitam de grande precisão no posicionamento da garra.

2.7.7 - Adaptador Automatizado de Garra

Surgiu da necessidade de se ter uma garra capaz de segurar todos os tipos de objetos. Então foi criada uma unidade chamada de trocador automatizado de ferramentas - *Automatic Gripper Changer*, que é um adaptador que permite que uma garra seja rapidamente ligada ou removida do braço do robô.

Esses adaptadores devem ser ligados ao braço do robô de um mesmo modo e deve se conectar de maneira idêntica suas unidades de acionamento, elétrica, mecânica ou pneumática. As principais desvantagens desse sistema são:

1. O peso adicional na extremidade do braço do robô;
2. Complicações tecnológicas são uma fonte potencial de mau funcionamento;
3. Acréscimo no custo do robô;
4. Tempo gasto na troca das garras.

Diante destes fatos verifica-se que o desenvolvimento e produção de garras fazem parte de um estágio importante no projeto de robôs para tarefas particulares. Normalmente, os fabricantes vendem robôs sem o atuador, as garras e as ferramentas são escolhidas e adaptadas pela equipe de engenharia que instala o robô no local de trabalho. Este é um estágio crítico da instalação, requerendo um alto nível de conhecimento e prática.

2.8 - Robôs com Estrutura Paralela

Os manipuladores robóticos com mecanismos paralelos em aplicações industriais têm sido muito empregados nas últimas décadas, principalmente como solução para sistemas de posicionamento, se evidenciado a implementação de metodologias para desenvolvimento de ferramentas de simulação e análise computacional para construção de protótipos baseadas nas condições de operação e funcionalidades estabelecidas através das especificações do projeto.

Assim, para a implementação de novos protótipos de um Manipulador Plataforma de Stewart (MPS) torna-se imprescindível a utilização de ferramentas dedicadas especificamente a este tipo de manipulador, que permita avaliar o seu desempenho através de ambiente de simulação antes de ser construído.

Neste item será apresentado um trabalho de revisão de literatura sobre manipuladores paralelos, com ênfase no Manipulador Plataforma de Stewart (MPS). Para uma

maior compreensão, neste trabalho de revisão bibliográfica serão discutidos conceitos e definições de manipuladores paralelos com a finalidade de estabelecer um marco de referência para a compreensão de seu princípio de funcionamento e principais aplicações industriais dos manipuladores paralelos.

2.8.1 - Comparação entre Manipulador serial e Paralelo

A comparação destes dois tipos de manipuladores é realizada basicamente em termos de suas características mecânicas e problemas de controle para determinar em que tipo de tarefas cada um pode ser utilizado com vantagem.

Em termos mecânicos, os manipuladores seriais são constituídos de atuadores nas suas partes móveis, implicando em massas e momentos de inércia relativamente altos (sistema rígido). Nos manipuladores paralelos todos os atuadores são montados próximos a base, possibilitando assim uma possível redução da massa nas suas partes móveis, implicando que os manipuladores paralelos apresentam características dinâmicas melhores em relação aos manipuladores seriais.

O volume de trabalho dos manipuladores paralelos é muito menor em relação a um manipulador serial. Em compensação, a estrutura mecânica de um manipulador paralelo é mais simples.

Para o controle de posição, a cinemática de um manipulador é um aspecto importante a ser considerado, em que a partir da cinemática direta, a posição e orientação do efetuador final são determinadas em função das variáveis articulares. O modelo cinemático inverso é necessário para o cálculo dos deslocamentos da juntas, quando são considerados os movimentos no espaço de trabalho, como, por exemplo, o movimento do efetuador final ao longo de uma trajetória.

Um sensor de força pode ser utilizado a partir do conhecimento das forças e torques obtidos através do modelo

estático direto. No controle de força no espaço de trabalho, o modelo estático inverso é requerido no cálculo dos torques e forças nos atuadores correspondentes aos momentos e forças no efetuador final. Os cálculos são muito mais simples quando realizados de forma direta, sem necessidade de achar a matriz inversa, consequentemente, o controle no espaço de trabalho é mais fácil de ser realizado para os manipuladores paralelos, enquanto a localização do efetuador final é mais simples nos manipuladores seriais. Assim, o controle de força no espaço de trabalho nos manipuladores seriais é mais fácil de ser implementado que nos manipuladores paralelos.

Os erros de posicionamento nas articulações são acumulativos nos manipuladores seriais, enquanto que nos manipuladores paralelos estes erros são a média de todos os erros. A capacidade de carga máxima nos manipuladores seriais é limitada para cada atuador que tem o torque de saída mínimo nos manipuladores seriais, enquanto que nos manipuladores paralelos a capacidade de carga é a soma das forças de todos os atuadores constituintes do manipulador.

O manipulador paralelo tem articulações passivas, as quais não estão associadas a um atuador, nos manipuladores seriais uma junta sempre está associada a um atuador. A tabela 2.4 apresenta uma comparação qualitativa das características mecânicas e modelagem dos manipuladores com arquitetura serial e paralela, onde podemos observar as principais diferenças entre essas duas arquiteturas e as principais vantagens e desvantagens na utilização das mesmas [13].

Podemos constatar que a utilização dos manipuladores seriais apresenta vantagens na realização de tarefas que requerem movimentos complexos e controle exato de força, como por exemplo uma montagem de pequenos objetos com precisão. Por outro lado os manipuladores paralelos podem ser aplicados para manipular objetos pesados em movimentos rápidos sem um grande range de orientação.

2.8.2 - Manipuladores Paralelos

Um robô paralelo é constituído de um efetuador com “n” graus de liberdade e de uma base fixa, interligados por, no mínimo, duas cadeias cinemáticas independentes. A atuação é realizada através de “n” atuadores simples.

Um manipulador paralelo generalizado é um mecanismo de cadeia cinemática fechada cujo efetuador está ligado à base através de várias cadeias cinemáticas independentes, podendo incluir mecanismos redundantes com mais atuadores que o número de graus de liberdade controlados do efetuador, bem como manipuladores trabalhando em cooperação. As principais características destes manipuladores são as seguintes:

No mínimo duas cadeias suportam o efetuador. Cada uma destas cadeias contém, no mínimo, um atuador simples. Há um sensor apropriado para medir o valor das variáveis associadas com a atuação (ângulo de rotação ou movimento linear). O número de atuadores é o mesmo que o número de graus de liberdade do efetuador. Um mínimo de duas cadeias permite uma melhor distribuição da carga sobre as cadeias.

NÍVEL	CARACTERÍSTICAS	MANIPULADOR SERIAL	MANIPULADOR PARALELO
MECANISMO	Inércia	Grande	Pequeno*
	VOLUME de trabalho	Grande*	Pequeno
	Aparência	Antropomórfica	Base Estrutural
	Fabricação	Difícil	Fácil*
CONTROLE	Controle de posição no espaço de trabalho	Difícil	Fácil*
	Controle de força no espaço de trabalho	Fácil*	Difícil
	Deteção de Forças	Difícil	Fácil*
	Erro de Posição	Acumulado	Médio *
Perto de pontos singulares	Erro de controle de Forças	Médio*	Acumulado
	Degeneração no controle de força	Diminuição de exatidão no posicionamento	
	Grande movimento no atuador	Grande força no atuador	
Dinâmica		Complicado*	Muito mais complicado

Tabela 2.4: Comparação entre Manipuladores Serial e MDS.

A literatura corrente traz uma série de arquiteturas mecânicas para robôs paralelos. Merlet as relacionou em robôs planares (manipuladores de dois e três graus de liberdade) e robôs de movimento espacial. Nesta revisão são abordados os manipuladores paralelos espaciais.

2.8.2.1 - Manipuladores de Três Graus de Liberdade

Manipuladores com três graus de liberdade em translação têm sido desenvolvidos para aplicações de manipulação. O mais conhecido robô deste tipo é o Delta, inicialmente desenvolvido por Clavel da *École Polytechnique Federale* de Lausanne em 1988, e atualmente comercializado pela Demaurex: Divisão de Tecnologia de Embalagens do Grupo Bosch, e pela ABB (*FlexPicker*). As aplicações para este tipo de robô estão concentradas na área de embalagem, manipulação de alimentos, indústria farmacêutica, médica e eletrônica, conforme mostra a figura 2.21.



a) Demaurex (máquinas de embalagem) b) Delta (ABB FlexPicker-IRB340)

Figura 2.21: Manipuladores com três graus de liberdade – translação.

Outro manipulador com três graus de liberdade é o orthoglide (figura 2.22a), que possui três atuadores lineares fixos montados ortogonalmente constituindo em três pernas idênticas.

ticas e uma plataforma móvel que se movimenta no espaço cartesiano com uma orientação fixa [14].

Manipuladores para orientação, ou seja, que permitem três rotações sobre um ponto, representa uma alternativa para o pulso com três juntas revolutas tendo eixos convergentes normalmente usados para robôs seriais. O mais simples gerador de movimentos rotativos sobre um ponto é constituído de um mastro central ligado à base e possuindo em sua extremidade uma junta bola-encaixe sobre a qual a base móvel é articulada. Um gerador de movimentos baseados em cadeias esféricas descrito por Asada foi desenvolvido por Gosselin [15,16] e equipe na Universidade Laval, no Canadá, e chamado *Agile Eye* para uma rápida orientação de câmeras (figura 2.22b).



a) Orthoglide.



b) 3 GL - rotação.

Figura 2.22: Manipuladores com três graus de liberdade.

2.8.2.2 - Manipuladores de Quatro Graus de Liberdade

Mecanismos com quatro graus de liberdade são encontrados há muito tempo na literatura. Em 1975, Koeverman apresentou um simulador de voo usando atuadores lineares e com a plataforma móvel submetida a restrições passivas, em que os graus de liberdade eram três rotações e uma translação sobre o eixo Z [17].

Uma aplicação para esse mecanismo é o manipulador **The Adept Quattro™ s650** (Adept Technology), sendo projetado para ser utilizado em operações Pick-and-place, tais como operações de manufatura, manipulação de materiais e montagem a alta velocidade. O manipulador Quadrupteron, criado pelo Dr. François Pierrot e sua equipe [18], foi inspirado no orthoglide, tem três graus de liberdade translacionais x-y-z e um de rotação no eixo z (figura 2.23).



a) The Adept Quattro™ s650



b) Quadrupteron (orthoglide)

Figura 2.23: Manipuladores com quatro graus de liberdade – 3 translações e 1 rotação.

2.8.2.3 - Manipuladores de Cinco Graus de Liberdade

O mecanismo implementado por Austad representa uma arquitetura híbrida para um manipulador de cinco graus de liberdade, que poderia ser interpretado como uma mistura entre um robô serial e um robô paralelo, em que um primeiro dispositivo paralelo controla a posição de um ponto específico e do efetuador e um segundo mecanismo paralelo garante duas rotações da plataforma móvel.

Outro mecanismo que podemos destacar é o de Zamanov, que desenvolveu uma arquitetura baseada no acoplamento de dois robôs paralelos planares. Esta estrutura permite o controle dos graus de liberdade da plataforma com exceção de rotação à normal à plataforma. Este último grau de liberdade pode ser controlado através de um atuador colocado sobre a plataforma. A figura 2.24 exemplifica estes dois mecanismos paralelos.

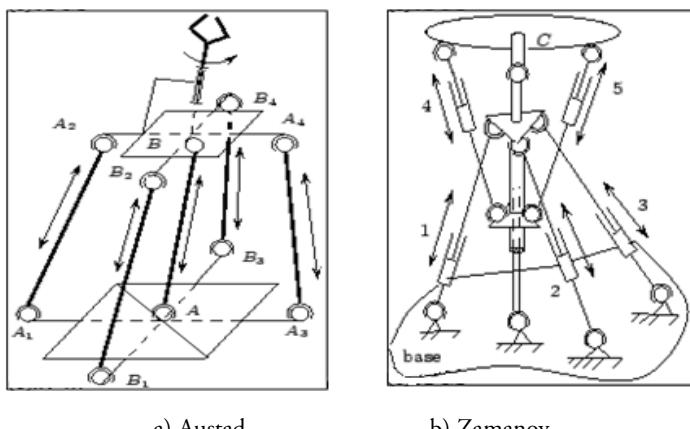


Figura 2.24: Manipuladores com cinco graus de liberdade.

2.8.2.4 - Manipuladores de Seis Graus de Liberdade

O manipulador HEXA consiste de seis cadeias cinemáticas muito leves. Todas estas cadeias são idênticas e constituídas de um braço, uma barra e de uma articulação esférica. O manipulador HEXA é fixado no teto, os seis motores são fixos na base, e cada um deles é associado diretamente ao braço. Todas estas seis cadeias cinemáticas terminam no efetuador final da plataforma móvel. O HEXA possui a capacidade de operação em altas velocidades, com baixas cargas na plataforma móvel. Este manipulador foi desenvolvido pela Toyota Machine Works [20].

Um Manipulador Plataforma de Stewart (MPS) é constituído por uma base fixa unida a uma base móvel por seis atuadores lineares, a distribuição espacial dos atuadores no MPS forma um octaedro [21]. O mecanismo desenvolvido opera com baixas velocidades, altas cargas na plataforma móvel e tem uma rigidez alta (figura 2.25).



a) HEXA



b) Manipulador Plataforma Stewart

Figura 2.25: Manipuladores paralelos de seis graus de liberdade.

2.8.3 - Exemplos de aplicação dos manipuladores paralelos

Manipuladores paralelos têm sido frequentemente utilizados nos últimos anos em diferentes aplicações no campo da engenharia, proporcionando soluções de posicionamento na indústria, utilização em diferentes aplicações em medicina e em variadas aplicações especiais, descritas a seguir.

2.8.3.1 - Aplicações Espaciais

Um exemplo é o robô CKCM, estudado pela NASA (*Goddard Space Flight Center*) por *Nguyen* [23]. Um robô hexápode também foi desenvolvido pelo Instituto Max Planck para ser usado com o telescópio UKIRT (*United Kingdom Infra-Red Telescope*) para movimentos lentos de foco. A companhia alemã Vertex também oferece uma estrutura paralela para uso com telescópios [23].

2.8.3.2 - Aplicações Médicas

Nesta área podemos destacar um endoscópio com dispositivo de fixação ativo usando um robô com três graus de liberdade atuados por fios que foi construído por Wendlant [24]. Outra aplicação médica para a estrutura paralela é o uso na assistência de pessoas deficientes para a movimentação de seus braços, como sugerido por Homma [25]. Estas estruturas também são utilizadas como suporte para microscópio e auxílio em cirurgias de precisão. A figura 2.26 apresenta algumas aplicações médicas utilizando manipuladores paralelos.



Figura 2.26: Manipuladores paralelos - Aplicações médicas.

2.8.3.3 - Aplicações Industriais

Atualmente, manipuladores paralelos são muito utilizados industrialmente devido às suas características de alta precisão de posicionamento e enorme rigidez. As principais aplicações destes manipuladores são em operações de montagem e desmontagem, manipulação de peças, soldagem a ponto e aplicações usando realimentação de força.

O modelo Delta representa o melhor exemplo de um manipulador de estrutura paralela usada industrialmente em tarefas de manipulação rápida de objetos. Nos dispositivos máquina-ferramenta de precisão um dispositivo paralelo do tipo plataforma de Stewart-Gough pode ser utilizado para posicionamento de peças mecânicas complexas durante a execução de operações de fresagem e usinagem em 3 e 5 eixos. A figura 2.27 apresenta algumas aplicações industriais utilizando esses manipuladores.



Figura 2.27: Manipuladores Paralelos - Aplicações Industriais.

2.9 - Avaliação de Desempenho de Robôs Industriais

A avaliação de desempenho de um robô industrial consiste na verificação da sua capacidade de realizar as funções para as quais foi construído. De uma forma geral as funções de um robô industrial são a movimentação de materiais, peças, ferramentas ou dispositivos. Desta forma, a avaliação de desempenho resulta em uma medida da eficácia do robô na realização desses movimentos.

Após definir parâmetros para quantificar a eficácia com que um robô executa uma tarefa, destaca-se a importância e a utilização da avaliação de desempenho, descreve-se as características para esta avaliação estabelecidas em normas internacionais, às condições de teste normalizadas e apresenta-se uma orientação para a seleção das características de desempenho a serem testadas para algumas aplicações dos robôs. Adicionalmente, descreve-se os procedimentos para a realização de testes comparativos entre robôs diferentes e indica-se os métodos de medição recomendados pelas normas para a avaliação de desempenho.

Uma medida da eficácia com que um robô realiza os movimentos inerentes a uma tarefa é a precisão com que os movimentos são realizados, como também a medida de seus movimentos repetitivos, através da análise de repetibilidade, que

é a capacidade de repetir o mesmo movimento. Além disso, o comportamento do robô durante seus movimentos, ou seja, seu desempenho dinâmico, também é uma informação importante para avaliar sua eficácia na execução de uma tarefa.

A avaliação de desempenho de um robô industrial é feita por meio da verificação da sua precisão, repetibilidade e desempenho dinâmico. São esses fatores que determinam se um robô é adequado à execução de uma tarefa. Assim, através da avaliação de desempenho pode-se verificar a adequação de um robô para a realização de uma tarefa industrial. Além disso, comparando desempenhos pode-se selecionar o robô mais apropriado para uma tarefa.

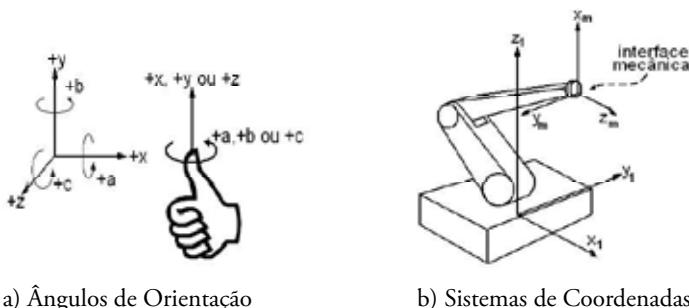
A avaliação de desempenho pode auxiliar ainda na escolha entre robôs de diferentes fabricantes, sempre que é possível comparar as avaliações realizadas pelos fabricantes e sua aceitação num ambiente industrial. Escolhido a partir das exigências da tarefa e das especificações do fabricante, é fundamental que no recebimento do robô essas especificações sejam avaliadas.

Além disso, durante o funcionamento do robô podem ocorrer desgastes que alteram suas características. Por isso avaliações de desempenho ao longo do tempo podem assegurar que o funcionamento do robô continua adequado para a realização de uma determinada tarefa, e verificação das dificuldades verificadas na avaliação de desempenho podem facilitar a manutenção.

2.9.1 - Precisão, Repetibilidade e Desempenho

A precisão, a repetibilidade e o desempenho dinâmico do robô são definidos a partir do elemento terminal existente na extremidade do robô, onde são fixadas as ferramentas e os dispositivos empregados na execução das tarefas (figura 2.28b). A posição da ferramenta é dada em termos de três coordenadas cartesianas e sua orientação através de três ângulos. De acordo com a norma ISO 9787 a representação da orientação da extremidade do robô é feita através dos ângulos a, b e c, respectivamente, em torno dos eixos x, y e z, cujo sentido positivo está indicado na figura 2.28a.

O conjunto formado pelas três coordenadas cartesianas e pelos três ângulos é definido como **postura**. A figura 2.28b mostra os sistemas de coordenadas fixados na base do robô (x_1, y_1, z_1) e na interface mecânica (x_m, y_m, z_m) de acordo com a norma ISO 9787. Neste caso a postura da interface mecânica é formada pelas três coordenadas cartesianas da origem do sistema (x_m, y_m, z_m) em relação ao sistema da base (x_1, y_1, z_1), e pelos três ângulos em torno dos eixos x_1, y_1 , e z_1 que determinam a orientação do sistema (x_m, y_m, z_m).



a) Ângulos de Orientação

b) Sistemas de Coordenadas

Figura 2.28: Posicionamento da ferramenta terminal.

A avaliação de desempenho é uma medida da eficácia com que o robô realiza tarefas com as ferramentas fixadas à sua interface mecânica. Por isso, essa avaliação é feita em relação a um **ponto de medição**, colocado a uma distância da interface mecânica para levar em conta a dimensão da ferramenta. Nesse ponto de medição é fixado um sistema de coordenadas cuja postura é o foco da avaliação de desempenho do robô.

A postura do ponto de medição é o resultado da combinação das posições de suas juntas, consequentemente, a repetibilidade e o desempenho dinâmico em uma dada postura são combinações da precisão, da repetibilidade e do desempenho dinâmico de cada uma de suas juntas.

A influência de cada junta nessa combinação varia ao longo do espaço de trabalho do robô. Devido a isso, a precisão, a repetibilidade e o desempenho dinâmico de um robô

industrial variam dentro do seu espaço de trabalho. As características de desempenho também variam com a velocidade e com carga aplicada na interface mecânica do robô. Assim, para avaliar o desempenho de um robô e compará-lo com o desempenho de outro é preciso conhecer as condições de teste utilizadas na avaliação de cada uma das características.

Enquanto alguns fabricantes desenvolveram condições de teste próprias, outros adotaram testes estabelecidos pelo *American National Standard Institute* (ANSI), e outros a norma internacional estabelecida pela *International Standard Organization* (ISO).

Os resultados variam muito de uma norma para outra, porque as condições de teste são diferentes e porque as fórmulas para calcular os resultados são diferentes. Para os mesmos dados de postura, por exemplo, as fórmulas empregadas nas normas americanas (ANSI) dão como resultado uma precisão maior do que as da ISO, enquanto a repetibilidade calculada segundo a ANSI é sempre menor do que a calculada de acordo com a ISO.

As normas americanas (ANSI) são voltadas principalmente à comparação de desempenho entre robôs de diferentes fabricantes, definindo métodos para avaliar o desempenho estático dos robôs industriais e avaliação do desempenho dinâmico. No Brasil, a Associação Brasileira de Normas Técnicas (ABNT) adota as normas ISO, sendo apresentados aqui mais detalhadamente os testes recomendados por essa instituição.

A ISO estabelece que a avaliação de desempenho deve ser realizada de acordo com a norma ISO 9283:1998 (Second edition) – “*Manipulating Industrial Robots – Performance Criteria and Related Methods*”. Os testes descritos nesta norma internacional permitem a avaliação de desempenho de robôs individuais e a comparação do desempenho entre robôs diferentes. A partir dessa norma, a precisão, a repetibilidade e o desempenho dinâmico de um robô industrial são avaliados pela medida dos parâmetros descritos na tabela 2.5

Medida de Desempenho de Robôs	Características
Precisão	Postura Variação multidirecional na precisão de postura À distância De percurso
Repetibilidade	Postura À distância De percurso
Avaliação do Desempenho Dinâmico	Tempo de estabilização Sobrepasso Desvios de canto Velocidades no percurso Tempo mínimo de posicionamento

Tabela 2.5: Parâmetros segundo a norma ISO 9283:1998.

Os desvios na precisão e a repetibilidade ao longo do tempo são medidos pelo deslocamento das características de postura, enquanto as variações na precisão e na repetibilidade entre robôs do mesmo modelo são caracterizadas pela intercambiabilidade.

A norma ISO 9283:1998 estabelece ainda uma característica para avaliar a flexibilidade do robô denominada de Flexibilidade Estática. Essas características podem ser usadas no todo ou em parte para avaliar o desempenho de um robô, observando que a norma não especifica quais delas devem ser empregadas para testar um robô em particular, apresentando apenas uma orientação para a seleção das características a serem testadas para algumas aplicações típicas.

2.9.2 - Características de Desempenho

A norma ISO 9283:1998 (segunda edição) define características para testar a postura, o percurso, o tempo mínimo de posicionamento e a flexibilidade estática dos robôs industriais.

2.9.3 - Características de postura

São grandezas que quantificam os erros entre uma **postura comandada** (especificada através da programação do robô) e a respectiva **postura atingida** (alcançada em resposta à postura comandada com o robô funcionando em modo automático).

Os erros podem ser causados pelo algoritmo de controle, pelas transformações de coordenadas, por diferenças dimensionais entre os componentes do robô e o seu modelo utilizado no sistema de controle, por dificuldades mecânicas como folgas, atrito, histerese, e por influências externas como a temperatura.

Nos robôs industriais a postura comandada pode ser especificada diretamente no robô mediante a gravação das coordenadas das juntas, através de uma caixa de comando ou de outra entrada manual de dados, e indiretamente através de um método de programação fora de linha. A forma pela qual a postura comandada é especificada influencia diretamente os resultados dos testes e, de acordo com a norma ISO 9283, deve estar claramente descrita no relatório dos testes.

Na figura 2.29 estão mostradas a postura comandada e a postura atingida, que devem ser medidas em relação a um sistema de coordenadas paralelo ao sistema da base (figura 2.28). Esta figura apresenta também o ponto de medição, neste caso tomado como o centro da ferramenta acoplada à interface mecânica.

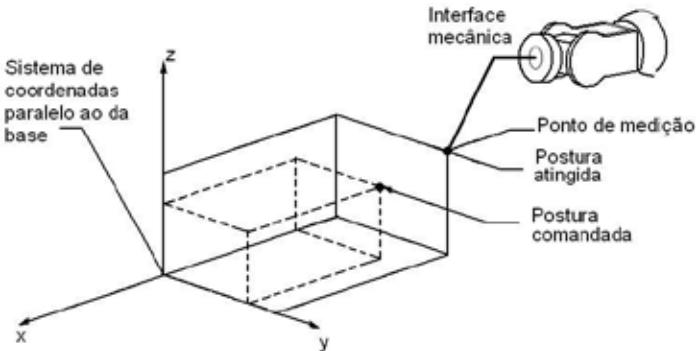


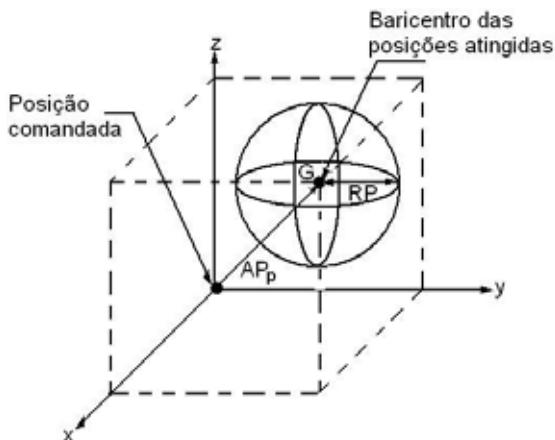
Figura 2.29: Postura comandada e postura atingida.

Segundo a norma ISO o desempenho é quantificado através da precisão de postura, da repetibilidade de postura, da variação multidirecional na precisão de postura, do deslocamento nas características de postura, da intercambiabilidade, da precisão e repetibilidade à distância, do tempo de estabilização, e do sobrepasso descritos a seguir.

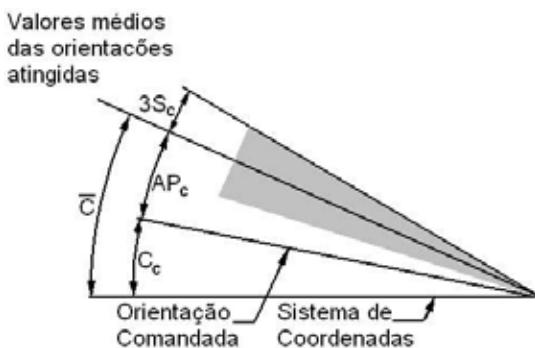
2.9.4 - Precisão de postura (AP)

É a diferença entre a postura comandada e a média das posturas atingidas quando a aproximação é realizada pela mesma direção. É dividida em:

- **Precisão de posicionamento (AP_p)**: a diferença entre uma posição comandada e o baricentro das posições atingidas (figura 2.30a);
- **Precisão de orientação (AP_o)**: a diferença entre a orientação angular comandada e a média das orientações angulares atingidas (figura 2.30b).



a) Posicionamento



b) Orientação

Figura 2.30: Precisão e Repetibilidade.

2.9.5 - Repetibilidade de Postura (RP)

- Expressa a proximidade das posturas atingidas após “n” visitas à mesma postura comandada na mesma direção. É quantificada através do:
- Raio da esfera centrada no baricentro do grupo de pontos atingidos (RP_p), calculado a partir da média dos raios de cada ponto atingido e do seu desvio padrão (figura 2.30a);
- Faixa de três desvios padrão das medidas angulares em torno dos valores médios dos ângulos de orientação (figura 2.30b).

2.9.6 -Variação multidirecional na precisão de postura (vAP)

Expressa a diferença entre as médias das posturas atingidas visitando a mesma postura comandada “n” vezes a partir de três direções ortogonais (figura 2.30). É quantificada por:

- vAP_p – a distância máxima entre os baricentros do conjunto de pontos atingidos ao final de cada percurso;
- vAP_a, vAP_b, vAP_c – o máximo desvio entre o valor médio dos ângulos atingidos ao final dos diferentes percursos.

2.9.7 - Deslocamento nas características de postura

Compreende o deslocamento na precisão e na repetibilidade. O deslocamento na precisão de postura (dAP) é a variação na precisão de postura no decurso de um tempo especificado. Da mesma forma, o deslocamento na repetibilidade de postura (dRP) é a variação na repetibilidade de postura ao longo de um determinado tempo. A tabela 2.5 apresenta uma classificação das principais características de desempenho para robôs industriais.

2.10 - Conclusão

A robótica requer conhecimentos provenientes de diversos campos como projeto mecânico, eletrônica de potência, integração de grande escala e engenharia de software, e continuará sendo influenciada pelos avanços em acionamentos, controle, mecanismos, programação e sensores. Sistemas de preensão constituem um elemento importante para análise de viabilidade e custos no desenvolvimento de sistemas robóticos.

Neste capítulo foram apresentados aspectos construtivos de manipuladores robóticos apresentando configurações industriais básicas, mecanismos de manipulação e preensão de objetos e aplicações, enfatizando o estudo do braço mecânico do robô, seus tipos de juntas e graus de liberdade, tipos de articulações, seu volume de trabalho (Workspace) e principais formas de acionamento.

Características	Aplicações Industriais					
	Solda Ponto	Movimentação de materiais	Montagem	Inspeção	Desbaste, Polimento e Corte	Pintura
Precisão e Repetibilidade de postura	X	X	X	X		X
Variação multidimensional na precisão de postura		X	X	X		
Deslocamento na precisão, repetibilidade de postura	X	X	X	X		X
Precisão, Repetibilidade à distância de postura	X	X	X	X		
Tempo de estabilização	X	X	X	X		
Sobrepasso	X	X	X	X		X
Precisão, Repetibilidade de percurso			X	X	X	X

Cont.

Desvios de canto			X	X	X		X
Precisão, Repetibilidade, Flutuação na velocidade de percurso				X	X	X	X
Tempo mínimo de posicionamento	X	X	X				
Flexibilidade estática	X	X	X		X		

Tabela 2.5: Orientação para a seleção das características de desempenho

2.11 - Referências Bibliográficas

- Scheinman, V., "Ideas on Implementing Modular Robot Systems", Technical Paper of Advanced Cybernetics Group, Inc., <http://www.advancedcybernetics.com/bmodula.htm>, 17/6/1992.
- Nof, S. Y., Handbook of Industrial Robotics, John Wiley & Sons, 1985.
- Armada, M.A., Control, de Robots, XV Curso de Automática en la Industria, Aguadulce (Almería), Junio de 1995.
- Introducing Robotics, Technical Specifications of Pegasus 11 Articulated Servo Robot System, Edacom Tecnologia, São Caetano do Sul, Brasil.
- Bastos, TT., Seguimiento y Análisis de Entornos de Soldadura por Arco Automatizada Mediante Ultrasonidos, Tesis Doctoral, Universidad Complutense de Madrid, Espailla, 1994.
- Basañez, L., "Multi-Sensor Integration in Robotics", Workshop on Robotícs and CIM, Lisbon, Portugal, September 1315, 1989.
- Spong, M.W., Vidyasagar, M., Robot Dynamics and Control, John Wiley & Sons, Inc., 1989.
- Engelberger, J.F., "Robotics in the 21th Century", Scientific American, September 1995.
- Delft Instruments Medical Imaging, 835 Sensor System - Template Description", 1990.
- IFR International Federation of Robotics, 2000.
- ABB International Report, 2000.
- Gosselin, C. M.; Perreault, L. & Vaillancourt, C., "Simulation and Computer-Aided Kinematic Desing of Tree-Degree-of-Freedom Spherical Parallel Manipulators", Journal of Robotic Systems 12(12), 857-869, 1999.

Arai, T. & B. Sheridan, "Characteristics and Mechanism Analysis of Parallel Link Manipulator", IFAC Robot Control, 119-124, 1988.

Pashkevich, A.; Chablat, D. & Wenger, P., "Kinematics and workspace analysis of a three-axis parallel manipulator: the Orthoglide", Robotica, Cambridge University Press 24, 39 – 49, 2006.

Gosselin, C. & Angles, J., "A global performance index for the kinematic optimization of robotic manipulators", Journal of mechanical design 3(113), 220-226, 1991.

Gosselin, C.; Pierre, E. & Gagne, M., "On the development of the Agile Eye", Robotics & Automation Magazine, IEEE 3(4), 29 – 37, 1996.

Koevermans, W.P et al., "Design and performance of the four DOF motion system of the NLR research flight simulator", Proc. of AGARD Conf., No 198, Flight Simulation 17, 1-11, 1975)

Nabat, V.; Pierrot, F.; Mijangos, M. R.; Arteche, J. A.; Zabalo, R. B.; Company, O., "Unlimited-rotation parallel robot with four degrees of freedom", Technical Report, Brevet International No. WO/2006/106165, 2006.

Stewart, D. (1965), 'A platform with six degrees of freedom', Proceedings of the IMechE 180(15), 371-385.

Robotics Research Center, Parallel Robot Design, Nanyang Technological University Modular Robotics & Robot Locomotion, 2005.

Bonev, I. (2003), 'The True Origins of Parallel Robots', ParallelMIC.

Yang, Y.; Rees, N. & Chuter, T., "Reduction of encoder measurement errors in UKIRT telescope control system using a Kalman filter", IEEE Transactions on Control Systems Technology 10, 149 – 157, 2002.

Nguyen, C.; Antrazit, S.; Zhout, Z. & C.E. Campbell, J., "Experimental Study of Motion Control and Trajectory Planning for a Stewart Platform Robot Manipulator", IEEE International Conference on Robotics and Automation 2, 1873-1878, 1991.

Wendlandt, J. M. & Sastry, S. S., "Design and Control of a Simplified Stewart Platform for Endoscopy", Proceedings of the 33rd conference on Decision and Control 1, 357-362, 1994.

Arai, T.; Cleary, K.; Nakamura, T.; Adachi, H. & Homma, K., "Design, Analysis and Construction of a Prototype Parallel Link Manipulator", IEEE International Workshop on Intelligent Robots and Systems', 205-212, 1990.

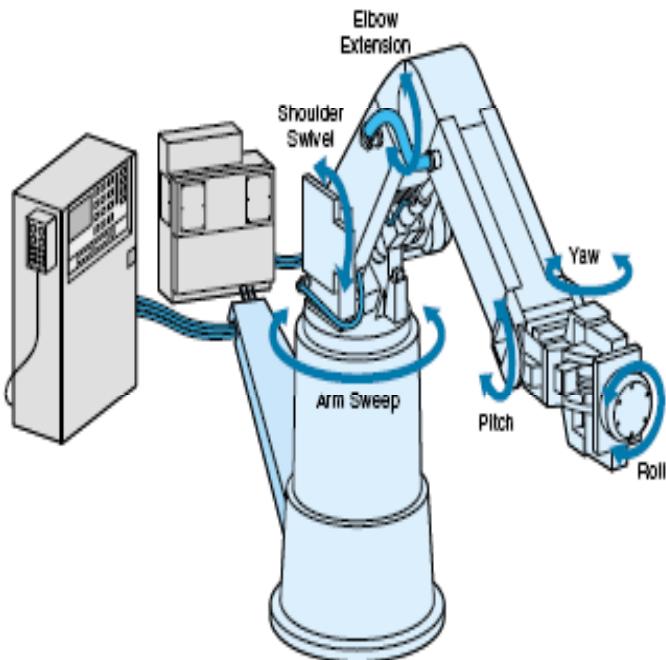
ISO 9787: "Manipulating Industrial Robots: Coordinate systems and motions", ISO Publications, França, 1990, 33 pp.

ISO 9283 Second edition 1998: "Manipulating Industrial Robots: Performance criteria and related test methods", ISO Publications, França, 1998, 33 pp.

ISO/TR 13309 First edition 1995: "Manipulating industrial robots: Informative guide on test equipment and metrology methods of operation for robot performance evaluation in accordance with ISO 9283", ISO Publications, França, 1995, 33 pp.

CAPÍTULO 3

Modelagem Cinemática



CAPÍTULO 3

Modelagem Cinemática

Um manipulador robótico é um dispositivo que tem por função posicionar e orientar uma ferramenta dedicada para uma operação fixa no seu elemento terminal. Assim, duas partes principais podem ser consideradas na estrutura de um manipulador: o braço constituído, no mínimo por três graus de liberdade utilizados para posicionamento do ponto de concentração dos referenciais de orientação e o punho, normalmente constituído por outros três graus de liberdade rotacionais, com a função de orientação do referencial terminal.

Neste capítulo é realizada uma revisão dos conceitos fundamentais sobre cinemática de robôs manipuladores incluindo a representação de Denavit-Hartenberg. A apresentação é direcionada a manipuladores robóticos antropomórficos com juntas prismáticas e rotacionais, permitindo que o mesmo possa se deslocar a partir de uma configuração de juntas inicial a outra configuração final desejada. Todos estes procedimentos apresentados poderão ser estendidos para qualquer configuração geométrica de robôs seriais. Durante este capítulo também são apresentados uma série de exemplos de modelagem manipuladores robóticos.

3.1 - Introdução

O número de robôs utilizados em instalações industriais vem aumentando, tendo em vista sua capacidade de realizar operações que exigem flexibilidade, rapidez e precisão.

Na maioria das aplicações industriais, a programação de tarefas de robôs é realizada por aprendizagem, consistindo no movimento individual de cada junta. Essa operação não necessita do modelo geométrico, tornando a programação de trajetórias de um robô no espaço de juntas, muito fácil, não necessitando de um conhecimento prévio do modelo, e apenas do conjunto de ângulos associados ao movimento angular de cada grau de liberdade do robô.

A movimentação coordenada dessas juntas será realizada através da operação de armazenamento de uma sequência de incrementos necessários para movimentação de cada junta de uma configuração inicial até seu posicionamento final, especificado a partir de um perfil de trajetórias fornecido. Essas informações (, servirão como sinal de referência para o controlador de posição de cada junta robótica (robô controlado a partir do sistema de coordenadas de juntas).

A trajetória de um robô é definida através de um conjunto de ângulos associados ao movimento angular de cada grau de liberdade do robô, que, após algoritmo de interpolação, servirá como sinal de referência para o controlador de posição de cada junta robótica que realizará uma comparação com os sinais provenientes dos transdutores de posição das juntas. Entretanto, muitas aplicações industriais exigem que o robô trabalhe de acordo com a posição e orientação do seu elemento terminal em relação ao sistema de coordenadas de trabalho, como por exemplo, um robô trabalhando em conjunto com uma máquina de comando numérico, numa célula automatizada com outros robôs, ou ainda quando o mesmo é dotado de um sistema de visão. Neste último caso, a interpretação das imagens se efetuará em relação ao sistema de coordenadas de trabalho (em duas ou três dimensões), e as infor-

mações extraídas das mesmas serão transmitidas ao Sistema de Supervisão após tratamento apropriado.

Como um robô é controlado através de suas variáveis articulares, a realização do controle do mesmo em relação ao sistema de coordenadas cartesianas implicará no desenvolvimento de metodologias para transformação de coordenadas. A transformação de coordenadas articulares para cartesianas é normalmente realizada em tempo real, em que a partir do conjunto de variáveis articulares serão obtidas a posição e orientação do elemento terminal de um robô.

O Supervisor de Controle é responsável pela geração dos sinais de referência individuais ao longo do tempo para cada junta do robô. Através de uma malha de controle de posição independente para cada junta, estes sinais são comparados com os valores atuais (obtidos através dos sensores de posição articulares), que faz com que a configuração de um robô seja controlada a partir de um valor desejado, independente do movimento desejado e da carga transportada pelo robô.

Entretanto, os valores das variáveis articulares utilizados como sinal de referência na malha de controle de posição das juntas quando comparados com os valores das juntas podem traduzir num erro, que aumenta com a sua velocidade de operação. Consequentemente, a implementação de um controlador de posição para um robô industrial exige o conhecimento da precisão cinemática do movimento do manipulador.

Para estabelecermos estratégias de controle de posição de juntas robóticas eficientes e precisas (erro próximo de zero), o movimento do robô é descrito através de equações diferenciais levando-se em consideração a sua arquitetura construtiva, a massa dos diferentes elementos, as inércias e tensor de inércia relacionado com a carga transportada, considerando também a modelagem completa de seu sistema de acionamento (motor-redutor), Paul, 1981.

3.2 – Sistemas de Referência

Um Sistema Articular pode ser representado matematicamente através de “n” corpos móveis C_i , com $i = 1, 2, \dots, n$, e de um corpo C_0 fixo, interligados por “n” articulações, formando uma estrutura de cadeia, sendo que estas juntas podem ser rotacionais ou prismáticas.

Para representar a situação relativa dos vários corpos da cadeia, é fixado a cada elemento C_i um referencial R. Podemos relacionar um determinado referencial R_{i+1} ($o_{i+1}, x_{i+1}, y_{i+1}, z_{i+1}$) com o seu anterior R_i (o_i, x_i, y_i, z_i), como também o sistema de coordenadas de origem da base (figura 3.1) através da equação (3.1), onde $A_{i,i+1}$ representa as matrizes de transformação homogênea de rotação e L_i o vetor de translação de uma origem a outra, onde $A_{i,i+1}$ é resultante do produto matricial global entre as diversas matrizes de transformações homogêneas relacionadas com rotações ou translações sucessivas das diferentes articulações conforme mostra a equação (3.2).

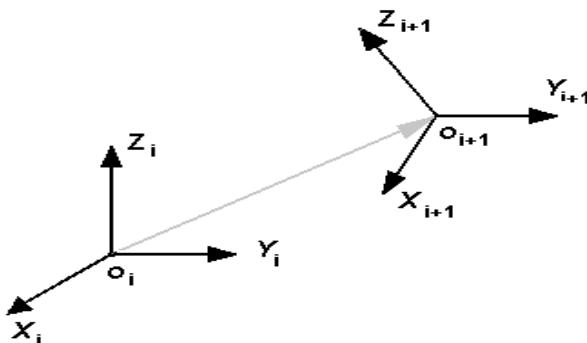


Figura 3.1: Sistema de Referência utilizado.

$$o_{i+1} = o_i + A_{i,i+1} * L_i \quad (3.1)$$

$$A_{i,i+1} = A_{1,2} \cdot A_{2,3} \cdot \dots A_{i,i+1} \quad (3.2)$$

onde

$$A_{i,i+1} = \begin{bmatrix} Nx_o & Sx_o & Ax_o \\ Ny_o & Sy_o & S\bar{y}_o \\ Nz_o & Sz_o & S\bar{z}_o \end{bmatrix} \quad (3.3)$$

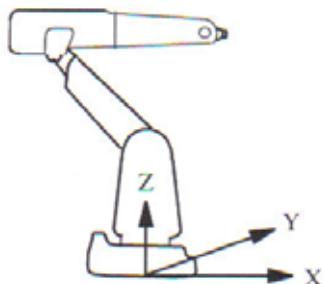
Qualquer rotação no espaço pode ser decomposta em um grupo de rotações elementares ao longo dos eixos X, Y e Z. A matriz de rotação elementar usada na equação de transformação é associada com a rotação elementar do referencial correspondente em relação ao seu anterior. Este procedimento matemático pode ser estendido para toda extensão do modelo. Assim sendo, a matriz de orientação de um ponto de interesse pode ser obtida pela equação (3.2).

Consequentemente, o posicionamento completo de um corpo rígido no espaço poderá ser facilmente obtido através da equação (3.1) que fornece o seu vetor posição, sendo que a equação (3.3) representa a matriz de orientação associada, podendo ser expressa através de componentes angulares associadas às três direções de rotação correspondentes aos eixos de referência do sistema de coordenadas (por exemplo, *Roll, Pitch, Yaw - RPY* ou *quaternions*).

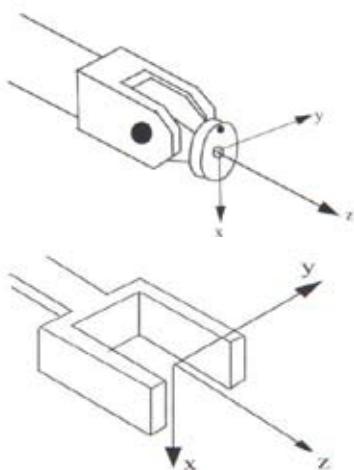
3.3 – Sistemas de Coordenadas utilizados em Células Robotizadas

Num robô industrial, os diferentes graus de liberdade podem ser associados a diversos sistemas de coordenadas servindo para descrever o movimento de cada grau de liberdade em relação a cada um desses sistemas. Por outro lado, o elemento final de um

robô é constituído de um sistema de referência designado Ponto Central da Ferramenta (TCP – Tool Center Point), utilizado para calibração de sua ferramenta terminal, e implementação de trajetórias em relação a um ponto de referência da ferramenta (posicionamento cartesiano ou orientação). Num robô industrial a escolha de um sistema de coordenadas adequado pode simplificar bastante a solução de seu modelo geométrico. A figura 3.2 apresenta o sistema de coordenadas da base e o sistema de coordenadas em relação ao TCP de um robô industrial.



a) Coordenadas da Base



b) Coordenadas do TCP

Figura 3.2: Sistemas de Referências Típicos de um Robô Industrial.

Os principais sistemas de coordenadas utilizados em células robotizadas industriais são os seguintes: Global, Base, Ferramenta, Objeto e Usuário, conforme mostra a figura 3.3.

O sistema de coordenadas global é definido a partir de um ponto de referência em relação ao sistema de coordenadas solidário à base do robô. A partir da utilização deste sistema de coordenadas, é possível relacionar a posição do robô a um ponto fixo da célula de trabalho onde o robô está atuando. O sistema de coordenadas global também é muito útil quando dois robôs estão trabalhando juntos ou quando utilizar um sistema de transferência robótico.

O sistema de coordenadas da base é associado à base de fixação de montagem do robô, enquanto o sistema de coordenadas da ferramenta é associado ao ponto central da ferramenta, estando associados ao posicionamento e orientação da mesma.

No sistema de coordenadas do usuário, o mesmo especifica um sistema de coordenadas de posicionamento de uma peça ou dispositivo externo ao manipulador, enquanto o sistema de coordenadas objeto especifica como um determinado objeto está posicionado em relação a um determinado dispositivo de fixação ou externo ao manipulador.

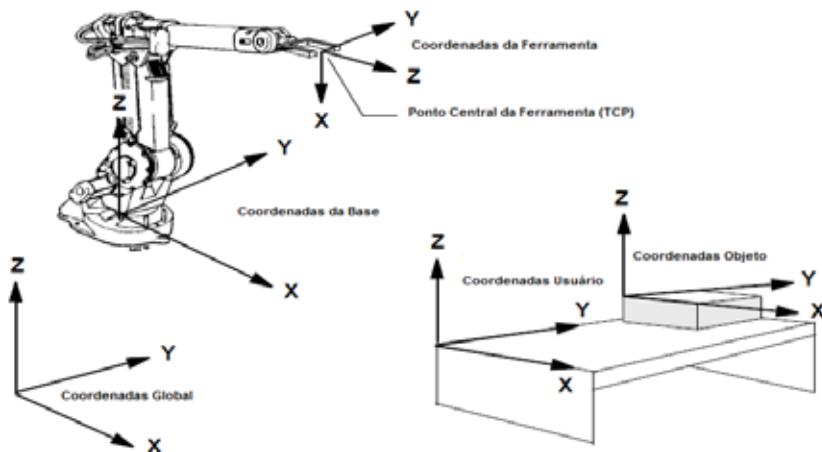


Figura 3.3: Sistemas de Coordenadas utilizados numa célula robotizada.

Num robô industrial cada posição deverá ser especificada em relação ao sistema de coordenadas objeto em relação ao posicionamento e orientação da ferramenta de trabalho. Isto implicará que no caso de alteração do tipo de ferramenta utilizada pelo robô o usuário não precisará de realizar grandes alterações no programa, fazendo apenas uma nova definição da mesma. Ao mesmo tempo, caso um dispositivo de fixação ou peça seja alterado, somente será necessário redefinirmos no programa o referencial relativo ao sistema de coordenadas do usuário ou do objeto.

Quando um robô está segurando um objeto de trabalho e trabalhando com uma ferramenta estacionária, é possível definirmos um TCP para esta ferramenta. Assim, quando essa ferramenta estiver ativa, o caminho programado e velocidade estão relacionados com o objeto de trabalho (figura 3.4). As figuras 3.5 e 3.6 apresentam alguns exemplos de utilização de sistemas de referência.

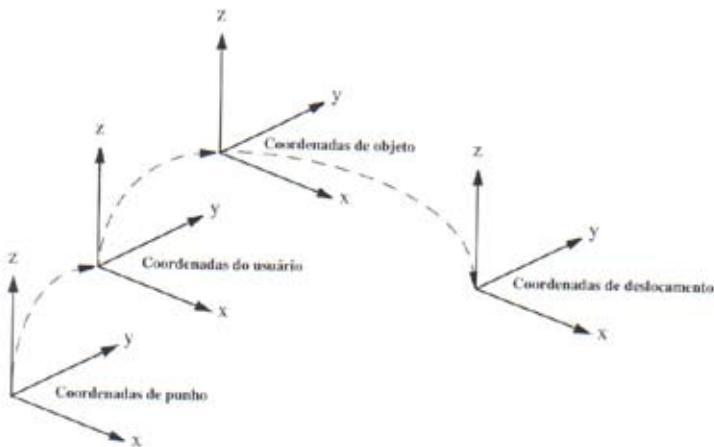
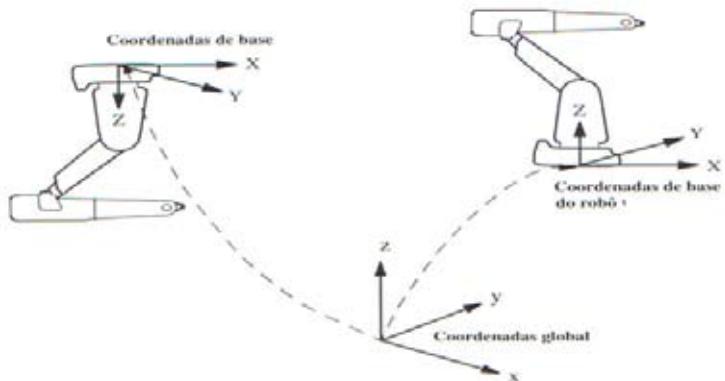
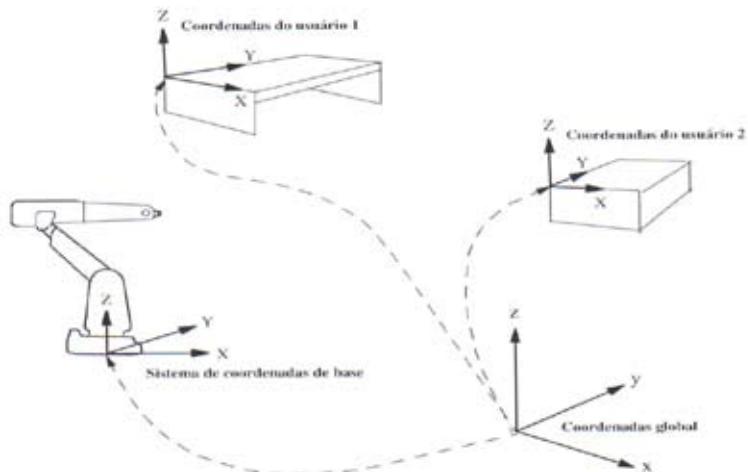


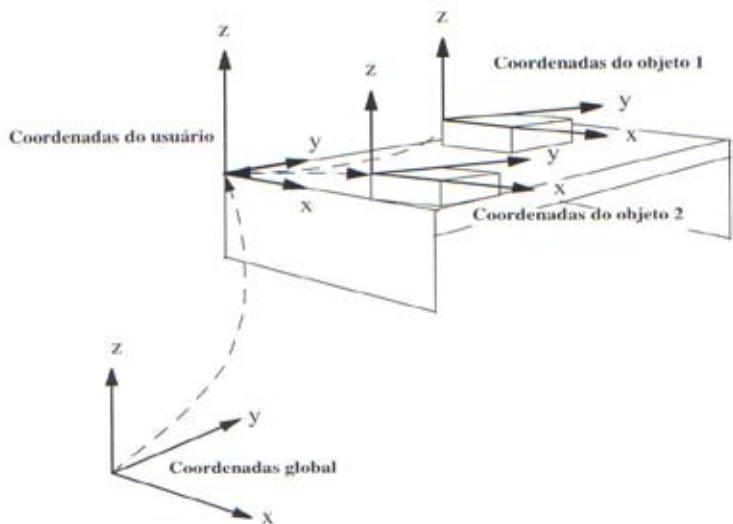
Figura 3.4: Sistemas de Coordenadas Ferramenta.



a) Definição de sistemas de coordenadas para dois robôs.

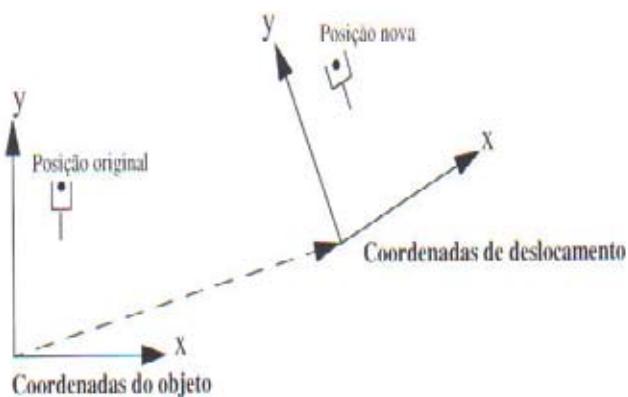


b) Definição de sistema de coordenadas do usuário.

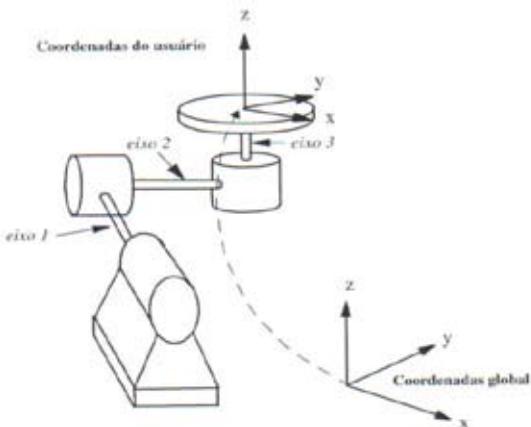


c) Definição de sistema de coordenadas do objeto.

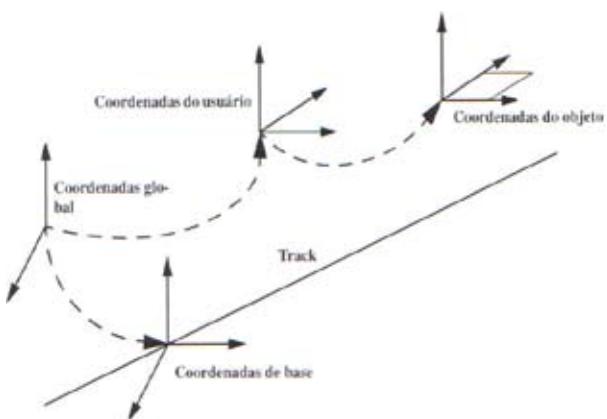
Figura 3.5: Exemplos de utilização de sistemas de referência.



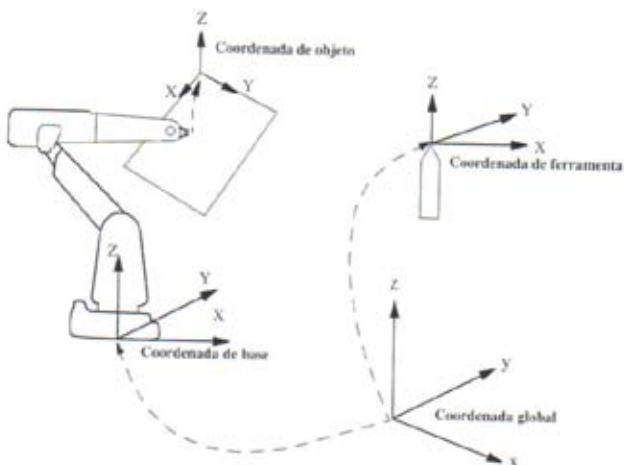
a) Sistema de Coordenadas do Deslocamento.



b) Eixos Externos Coordenados.



c) Movimentação do sistema de Coordenadas da base.



d) TCP's estacionários

Figura 3.6: Exemplos de utilização de sistemas de referência.

3.4 - Modelo Geométrico

O modelo geométrico é aquele que expressa a posição (\underline{x}) e orientação do elemento terminal (vetores \underline{n} , \underline{s} e \underline{a}) em relação a um sistema de coordenadas solidário à base do robô, em função de suas coordenadas generalizadas (angulares, no caso de juntas rotacionais), figura 3.7.

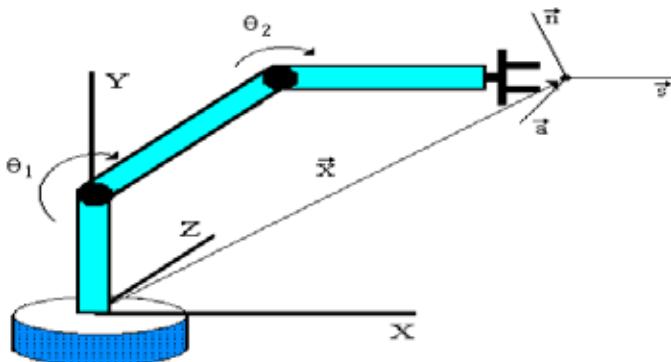


Figura 3.7: Modelo Geométrico de um Robô.

Esta relação pode ser expressa matematicamente a partir de uma matriz que relaciona o sistema de coordenadas da base com o sistema de coordenadas do último elemento. Esta matriz é chamada de matriz de transformação de coordenadas homogêneas do robô.

Embora a definição do modelo geométrico seja única, a maneira de obter a matriz de passagem homogênea do robô está associada ao sistema de referência utilizado. Existem diversas maneiras, e cada uma gera expressões diferentes que, embora equivalentes quantitativamente, podem diferir quanto ao número de operações aritméticas necessárias para fazer o cálculo numérico das mesmas. Um robô pode ser modelado através de vários métodos: através da sistemática de Denavit-Hartenberg ou a partir de vetores locais.

O modelo geométrico é definido função vetorial f , que exprime um vetor \underline{x} (espaço cartesiano) em função do vetor $\underline{\theta}$ (coordenadas angulares):

$$\underline{x} = f(\underline{\theta}) \quad (3.4)$$

onde

$\underline{\theta} = (\theta_1, \theta_2, \dots, \theta_n)$: vetor das posições angulares das juntas e
 $\underline{x} = (p_x, p_y, p_z, \psi, \theta, \phi)$: vetor posição, em que os três primeiros termos denotam a posição cartesiana e os três últimos a orientação do órgão terminal.

A função f , que é o modelo geométrico, é então a expressão analítica da composição dos movimentos das juntas para realizar o movimento do elemento terminal do robô. Esta relação pode ser expressa matematicamente pela matriz de transformação homogênea que relaciona o sistema de coordenadas solidárias a base do robô com um sistema de coordenadas associadas a sua ferramenta terminal. Esta matriz é obtida a partir do produto das matrizes de transformação, $A_{i,i-1}$, que relaciona o sistema de coordenadas de um elemento i com o sistema de coordenadas anterior $i-1$, isto é:

$$T_n = A_{0,1} \cdot A_{1,2} \cdot \dots \cdot A_{n-1,n} \quad (3.5)$$

$$T_n = [n \ s \ a \ p]$$

onde

$\underline{p} = [p_x, p_y, p_z]$: vetor posição e
 $\underline{n} = [n_x, n_y, n_z]$, $\underline{s} = [s_x, s_y, s_z]$ e $\underline{a} = [a_x, a_y, a_z]$: vetor ortonormal que descreve a orientação.

Embora a definição do modelo geométrico seja única, a maneira de obter a matriz de passagem homogênea do robô está associada ao sistema de referência utilizado. Existem diversas maneiras, e cada uma gera expressões diferentes que, embora equivalentes quantitativamente, podem diferir quanto ao número de operações aritméticas necessárias para fazer o cálculo numérico das mesmas. Pode-se citar duas maneiras de se obter a matriz de passagem homogênea do robô: através da sistemática de Denavit-Hartenberg e através de vetores locais.

Através da sistemática de Denavit-Hartenberg serão obtidas a posição e a orientação finais do elemento terminal. O modelo do robô obtido desta forma é indicado para o uso em programas de geração de trajetórias, pois nestes programas o que interessa é apenas a posição e orientação do elemento terminal. Por outro lado, utilizando vetores locais, são obtidas as posições e as orientações de diversos pontos de interesse que podem ser utilizados para a construção gráfica do robô através de elementos primitivos.

3.5 - Transformação de Coordenadas

Numa determinada aplicação industrial, um robô pode ser controlado e programado a partir do sistema de coordenadas associadas ao seu elemento terminal. É muito mais natural expressarmos o deslocamento absoluto do elemento terminal de um robô que considerarmos a variação de suas coordenadas

articulares, embora a malha de controle de uma junta robótica seja estabelecida a partir da comparação de grandezas articulares, tornando-se necessário a realização de uma transformação geométrica apropriada para o estabelecimento da correspondência entre as variáveis articulares θ_i e as coordenadas absolutas do elemento terminal X_i . A figura 3.8 apresenta um esquema descrevendo o problema de transformação direta de coordenadas para um robô com “n” graus de liberdade.

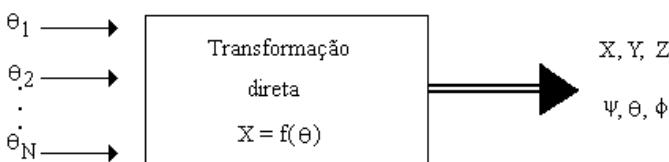


Figura 3.8: Transformação Direta de Coordenadas.

A operação que realiza a correspondência entre esses dois espaços é chamada de transformação de coordenadas. A transposição direta de coordenadas apresenta uma solução única, o mesmo não acontece com o problema inverso, que pode conduzir à soluções múltiplas.

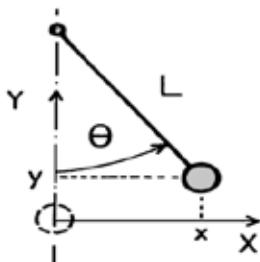
3.6 – Apresentação de Exemplos

Nesta seção serão apresentados dois exemplos de solução do problema cinemático inverso para robôs elementares (um e dois graus de liberdade). Nos capítulos seguintes serão abordados exemplos de robôs com mais graus de liberdade.

3.6.1 - Robô Elementar (1 GL) – Pêndulo simples

A figura 3.9 apresenta um robô elementar (pêndulo simples) com 1 GL (grau de liberdade) e de comprimento L (perfeitamente rígido), onde as coordenadas X e Y do elemento terminal são expressas em relação ao sistema de coordenadas

apresentadas. A partir de um dado valor θ ficam determinadas as coordenadas $X_T = (X, Y)^T$ do elemento terminal do robô em relação ao seu sistema de coordenadas. Esta operação é chamada transformação direta de coordenadas.



Modelo Matemático associado:

$$X = L \cdot \sin \theta$$

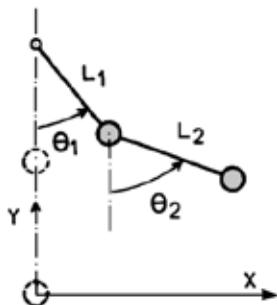
$$Y = L \cdot (1 - \cos \theta)$$

Figura 3.9: Robô com 1 grau de liberdade (pêndulo simples).

Para deslocarmos a extremidade do seguimento L do robô para uma posição desejada $M = (X_o, Y_o)^T$ basta utilizarmos a coordenada θ , ou seja, $\theta = \arcsin(X_o/L)$, com $Y_o \leq L$.

3.6.2 - Robô com 2 GL – Pêndulo duplo

A figura 3.10 apresenta um robô com dois graus de liberdade, constituído de dois pêndulos com comprimentos L_1 , L_2 , em que as coordenadas absolutas X e Y da extremidade de L_2 são expressas em relação aos sistemas de coordenadas apresentados.



Modelo Matemático associado:

$$X = L_1 \cdot \sin \theta_1 + L_2 \cdot \sin \theta_2$$

$$Y = L_1 \cdot (1 - \cos \theta_1) + L_2 \cdot (1 - \cos \theta_2)$$

Figura 3.10: Robô com 2 graus de liberdade (pêndulo duplo).

A transformação inversa de coordenadas consistirá na definição de um vetor $\theta = (\theta_1, \theta_2)^T$, a partir do posicionamento do robô num determinado ponto $M(X_o, Y_o)^T$, a partir da obtenção dos valores θ_1 e θ_2 expressos em função de X_o e Y_o .

3.7 – Matrizes de Transformação Homogênea

Denavit e Hartenberg (Paul, 1981) introduziram o conceito de coordenadas homogêneas à análise dos mecanismos. Um ponto P de um corpo rígido no espaço pode ser representado fisicamente por uma matriz \mathfrak{R}^3 e em coordenadas homogêneas por uma matriz \mathfrak{R}^4 ou MTH, que representa a posição e orientação da ferramenta do robô. Esta matriz pode ser subdividida em quatro partes como é descrito na equação

3.6: Matriz de rotação \mathbf{R}_{3x3} , Vetor de translação \mathbf{p}_{3x1} , Transformação em perspectiva \mathbf{O}_{1x3} , e fator de escala \mathbf{E}_{1x1} .

$$\mathbf{A} = \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{p}_{3x1} \\ \mathbf{O}_{1x3} & \mathbf{E}_{1x1} \end{bmatrix} \quad (3.6)$$

A figura 3.11 ilustra um sistema de coordenadas arbitrárias para representar a ideia de MTH de um ponto P no espaço.

Seja o vetor \mathbf{p}^0 cujo sistema de coordenadas é $O_0(X_0, Y_0, Z_0)$ e o vetor \mathbf{p}^1 cujo sistema de coordenadas é $O_1(X_1, Y_1, Z_1)$.

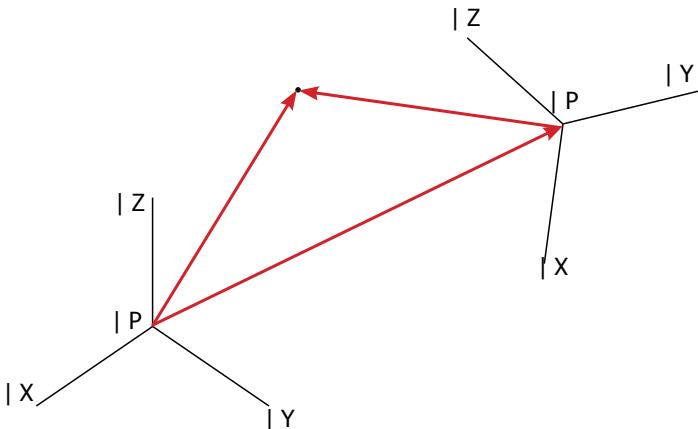


Figura 3.11: Representação de um ponto em relação a dois sistemas de coordenadas.

Seja o vetor \mathbf{r}_1^0 , que descreve a posição do referencial $O_1(X_1, Y_1, Z_1)$, com respeito ao referencial $O_0(X_0, Y_0, Z_0)$ e seja também \mathbf{R}_1^0 a matriz de rotação do referencial 1, respeito ao referencial 0. Através da análise geométrica, tem-se:

$$\begin{aligned}\mathbf{p}^0 &= \mathbf{r}_1^0 + \mathbf{R}_1^0 \mathbf{p}^1 \\ \mathbf{p}^0 &= \mathbf{R}_1^0 \mathbf{p}^1 + \mathbf{r}_1^0\end{aligned}\tag{3.7}$$

A posição e orientação do ponto P são representadas com a seguinte MTH:

$$\mathbf{A}_1^0 = \begin{bmatrix} \mathbf{R}_1^0 & \mathbf{r}_1^0 \\ \mathbf{O}^T & 1 \end{bmatrix}\tag{3.8}$$

A transformação inversa obtém-se pré-multiplicando os dois lados da anterior expressão por \mathbf{R}_1^{0T} porque $\mathbf{R}^T \mathbf{R} = \mathbf{I}$, então:

$$\begin{aligned}\mathbf{R}_1^{0T} \mathbf{p}^0 &= \mathbf{R}_1^{0T} \mathbf{r}_1^0 + \mathbf{R}_1^{0T} \mathbf{R}_1^0 \mathbf{p}^1 \\ \mathbf{p}^1 &= -\mathbf{R}_0^1 \mathbf{r}_1^0 + \mathbf{R}_0^1 \mathbf{p}^0\end{aligned}\tag{3.9}$$

A descrição matemática das matrizes de transformação homogênea será realizada inicialmente através do uso do método de Denavit-Hartenberg, após a obtenção dos quatro parâmetros θ_i , a_i , d_i e α_i , ou através da utilização vetores locais.

3.8 - Sistemática de Denavit-Hartenberg (DH)

A evolução no tempo das coordenadas das juntas de um robô representa o modelo cinemático de um sistema articulado no espaço tridimensional. A notação de Denavit-Hartenberg é uma ferramenta utilizada para sistematizar a descrição cinemática de sistemas mecânicos articulados com “n” graus de liberdade (Denavit, 1955).

Neste método, são obtidas apenas as coordenadas do elemento terminal do robô. O modelo obtido deste modo pode ser utilizado em programas de geração de trajetórias e de identificação de erros, entre outros, uma vez que são necessárias apenas as coordenadas do elemento terminal. Algoritmo para obtenção do sistema de coordenadas para o link utilizando a convenção de Denavit Hartenberg.

Na figura 3.12 podemos visualizar dois links conectados por uma junta que tem duas superfícies deslizantes uma sobre a outra remanescentes em contato. Um eixo de uma junta i ($i = 1, \dots, 6$) estabelece a conexão de dois *links*.

Estes eixos de juntas devem ter duas normais conectadas neles, uma para cada um dos links. A posição relativa destes

dois links conectados (link $i-1$ e link i) é dada por d_i , que é a distância medida ao longo do eixo da junta entre suas normais. O ângulo de junta θ_i entre as normais é medido em um plano normal ao eixo da junta. Assim, d_i e θ_i podem ser chamados, respectivamente, distância e o ângulo entre links adjacentes. Eles determinam a posição relativa de *links* vizinhos.

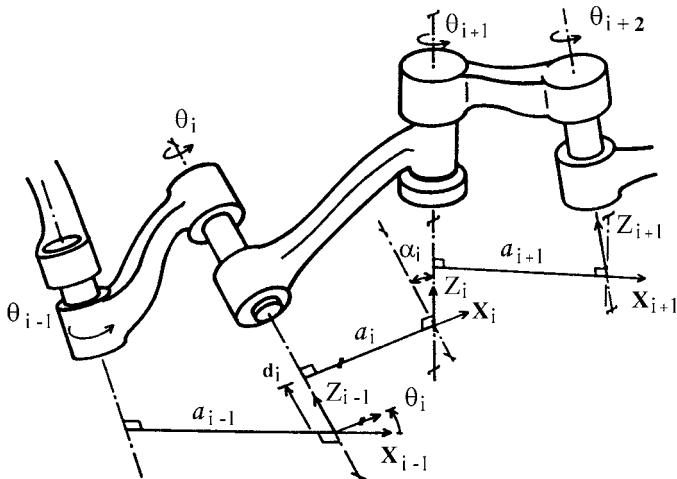


Figura 3.12: Notação de Denavit-Hartenberg.

Um *link* i poderá estar conectado, no máximo, a dois outros links (link $i-1$ e link $i+1$). Assim, dois eixos de junta são estabelecidos em ambos terminais de conexão. Os *links*, do ponto de vista cinemático, mantêm uma configuração fixa entre suas juntas que podem ser caracterizadas por dois parâmetros: a_i e α_i .

O parâmetro a_i é a menor distância medida ao longo da normal comum entre os eixos de junta (isto é, os eixos z_{i-1} e z_i para a junta i e junta $i+1$, respectivamente) Assim, a_i e α_i podem ser chamados, respectivamente, comprimento e ângulo de *twist* (torção) do *link* i . Eles determinam a estrutura do *link* i .

Assim sendo, quatro parâmetros (a_i , α_i , d_i , θ_i) são associados com cada link do manipulador. No momento

em que estabelecemos uma convenção de sinais para cada um destes parâmetros, estes constituem um conjunto suficiente para determinar a configuração cinemática de cada *link* do manipulador. Note que estes quatro parâmetros aparecem em pares:

- (a_i, α_i) que determinam a estrutura do *link* e os parâmetros da junta.
- (d_i, θ_i) que determinam a posição relativa de *links* vizinhos.

Para descrever a translação e rotação entre dois *links* adjacentes, Denavit e Hartenberg propuseram um método matricial para estabelecimento sistemático de um sistema de coordenadas fixo para cada link de uma cadeia cinemática articulada.

A representação de Denavit-Hartenberg (D-H) resulta na obtenção de uma matriz de transformação homogênea 4×4 , representando cada sistema de coordenadas do link na junta, em relação ao sistema de coordenadas do *link* anterior. Assim, a partir de transformações sucessivas, podem ser obtidas as coordenadas do elemento terminal de um robô (último *link*), expressas matematicamente no sistema de coordenadas fixo a base.

Assim sendo, um sistema de coordenadas cartesianas ortonormal (X_i, Y_i, Z_i) pode ser estabelecido para cada *link* no seu eixo de junta, em que $i = 1, 2, \dots, N$ (N número de graus de liberdade) mais o sistema de coordenadas da base. Assim, uma junta rotacional tem somente um grau de liberdade, e cada sistema de coordenadas (X_i, Y_i, Z_i) do braço do robô corresponde a junta $i+1$, sendo fixo no *link* i .

Quando o acionador ativa a junta i , o *link* i deve mover-se com relação ao *link* $i-1$. Assim, o i -ésimo sistema de coordenadas é solidário ao link i , se movimentando junto com o mesmo. Assim, o n -ésimo sistema de coordenadas se movimentará com o elemento terminal (*link n*). As coordenadas da base são definidas como o sistema de coordenadas O (X_0 ,

Y_0 , Z_0), também chamado de sistema de referência inercial. Os sistemas de coordenadas são determinados e estabelecidos obedecendo três regras:

- O eixo Z_{i-1} é colocado ao longo do eixo de movimento da junta i .
- O eixo X_i é normal ao eixo Z_{i-1} , e apontando para fora dele.
- O eixo Y_i completa o sistema utilizando a regra da mão direita.

Através destas regras podemos observar que:

A escolha do sistema de coordenadas é livre, podendo ser colocada em qualquer parte da base de suporte, enquanto que a posição do eixo Z_0 deverá ser a do eixo de movimento da primeira junta. O último sistema de coordenadas (n -ésimo) pode ser colocado em qualquer parte do elemento terminal, enquanto que o eixo X_i é normal ao eixo Z_{i-1} .

A representação D-H de um *link* rígido dependerá de quatro parâmetros associados ao link. Estes parâmetros descrevem completamente o comportamento cinemático de uma junta prismática ou de revolução. Estes quatro parâmetros são definidos a seguir:

- θ_i é o ângulo de junta obtido entre os eixos X_{i-1} e X_i no eixo Z_{i-1} (usar a regra da mão direita).
- d_i é a distância entre a origem do ($i-1$)-ésimo sistema de coordenadas até a interseção do eixo Z_{i-1} com o eixo X_i ao longo do eixo Z_{i-1} .
- a_i é a distância (off-set) entre a interseção do eixo Z_{i-1} com o eixo X_i até a origem o i -ésimo sistema de referência ao longo do eixo X_i (ou a menor distância entre os eixos Z_{i-1} e Z_i).

- α_i é o ângulo offset entre os eixos Z_{i-1} e Z_i medidos no eixo X_i (usando a regra da mão direita).

Para uma junta rotacional, d_i , a_i , e α_i são os parâmetros, variando o seu valor na rotação do *link* i em relação ao *link* $i-1$. Para uma junta prismática θ_i , a_i e α_i são os parâmetros, enquanto d_i é a variável de junta (deslocamento linear). A figura 3.13 ilustra o conceito de junta rotacional e prismática.

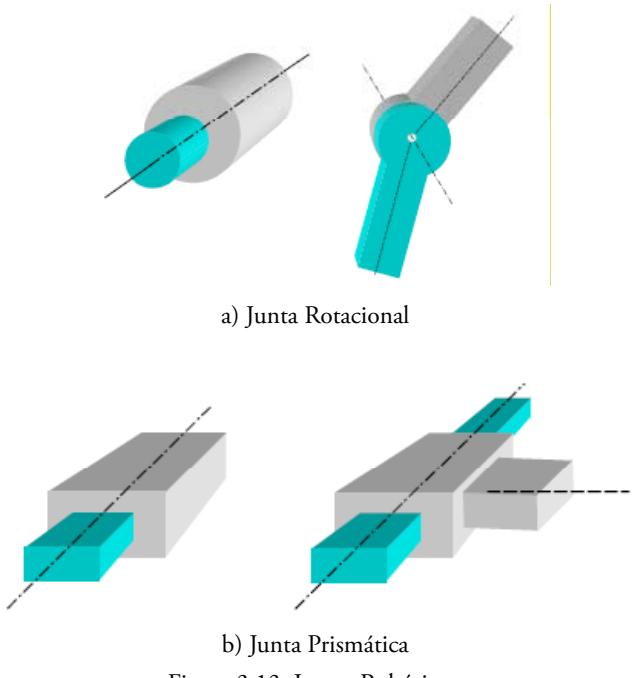


Figura 3.13: Juntas Robóticas.

3.8.1 - Algoritmo Resumido

Dado um manipulador com “n” graus de liberdade, o algoritmo descrito a seguir, determina um sistema de coordenadas ortonormais para cada *link* do robô, a partir do sistema de coordenada fixo à base de suporte (sistema inercial) até o seu elemento terminal. As relações entre os links adjacentes

podem ser representadas por uma matriz de transformação homogênea (4×4). O conjunto de matrizes de transformação homogêneo permite a obtenção do modelo cinemático do robô. A tabela 3.1 apresenta um algoritmo resumido para obtenção dos parâmetros de DH.

D1: Obtenção do sistema de coordenadas da base: <i>Estabelecer um sistema ortonormal de coordenadas (X_o, Y_o, Z_o) na base de suporte com o eixo Z_o colocado ao longo do eixo de movimento da junta 1 apontando para o ombro do braço do robô. Os eixos X_o e Y_o podem ser convenientemente estabelecidos e são normais ao eixo Z_o.</i>
D2: Inicialização e iteração: <i>Para cada i, $i = 1, \dots, N-1$, efetuar passos D3 até D6.</i>
D3: Estabelecer o eixo das juntas: <i>Alinhar Z_i com o eixo de movimento (rotação ou translação) da junta $i+1$. Para robôs tendo configurações de braço esquerdo-direito, os eixos Z_1 e Z_2 são apontados sempre para o ombro e o tronco do braço do robô.</i>
D4: Estabelecer a origem do i-ésimo sistema de coordenadas: <i>Situar a origem do iésimo sistema de coordenadas na interseção dos eixos Z_i e Z_{i-1} ou na interseção da normal comum entre os eixos Z_i e Z_{i-1} e o eixo Z_i.</i>
D5: Estabelecimento do eixo X_i: <i>Estabelecer $X_i = \pm(Z_{i-1} \times Z_i) / Z_{i-1} \times Z_i$ ou ao longo da normal comum entre os eixos Z_i e Z_{i-1} quando eles forem paralelos.</i>
D6: Estabelecimento do eixo y_i: <i>Determina-se $Y_i = \pm(Z_i \times X_i) / Z_i \times X_i$ para completar o sistema de coordenadas. (Estender os eixos Z_i e X_i se necessário para passos D9 a D12).</i>
D7: Estabelecer a direção do sistema de coordenadas: <i>Normalmente a n-ésima junta é uma junta rotativa. Estabelecer Z_n ao longo da direção do eixo Z_{n-1} apontando para fora do robô. Estabelecer X_n assim que ele é normal tanto aos eixos Z_{n-1} e Z_n. Determine Y_n para completar o sistema de coordenadas.</i>

D8: Encontrar os parâmetros das juntas e links: Para cada i , $i = 1, \dots, n$, efetuar passos D9 ao D12.

D9: Encontrar d_i : d_i é a distância da origem do ($i-1$)-ésimo sistema de coordenadas até a interseção do eixo Z_{i-1} e o eixo X_i ao longo do eixo Z_{i-1} . Ela é a variável de junta se a junta i é prismática.

D10: Encontrar a_i : a_i é a distância da interseção do eixo Z_{i-1} e o eixo X_i para a origem do i -ésimo sistema de coordenadas ao longo do eixo X_i .

D11: Encontrar θ_i : θ_i é o ângulo de rotação entre os eixos X_{i-1} e X_i sobre o eixo Z_{i-1} . Esta é a variável de junta se a junta é rotacional.

D12: Encontrar α_i : α_i é o ângulo de rotação entre os eixos Z_{i-1} e Z_i no eixo X_i .

Tabela 3.1: Algoritmo Resumido Denavit-Hartenberg (DH).

3.8.2 - Obtenção da Matriz de Transformação Homogênea ${}^{i-1}A_i$

Uma vez os sistemas de coordenadas D-H tenham sido estabelecidos, uma matriz de transformação homogênea pode facilmente ser desenvolvida relacionando o i -ésimo ao ($i-1$)-ésimo frame de coordenadas. A figura 3.6 mostra que um ponto r_i , expresso no i -ésimo sistema de coordenadas, pode ser expresso no ($i-1$)-ésimo sistema de coordenadas como r_{i-1} aplicando as transformações sucessivamente apresentadas a seguir:

1. Rotação no eixo Z_{i-1} de um ângulo de θ_i para alinhar o eixo X_{i-1} com o eixo X_i (o eixo X_{i-1} é paralelo ao eixo X_i e aponta para a mesma direção).
2. Translação uma distância de d_i ao longo do eixo Z_{i-1} para trazer os eixos X_{i-1} e X_i na coincidência.
3. Translação ao longo do eixo X_i uma distância de a_i para trazer as duas origens também como o eixo X na coincidência.
4. Rotação do eixo X_i um ângulo de α_i para trazer os dois sistemas de coordenadas na coincidência.

Cada uma destas quatro operações pode ser expressa através de uma matriz homogênea de rotação-translação, e o produto destas quatro matrizes de transformações elementares produzem uma matriz de transformação homogênea composta ${}^{i-1}A_i$, conhecida como matriz de transformação de D-H, para sistemas de coordenadas adjacentes, i e i-1.

$${}^{i-1}A_i = T_{z,d} T_{z,\theta} T_{x,a} T_{x,\alpha} \quad (3.10)$$

$$\begin{aligned} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (3.11)$$

A transformação inversa será:

$$[{}^{i-1}A_i]^{-1} = {}^iA_{i-1} = \begin{bmatrix} \cos\theta_i & \sin\theta_i & 0 & -a_i \\ -\cos\alpha_i \sin\theta_i & \cos\alpha_i \cos\theta_i & \sin\alpha_i & -d_i \sin\alpha_i \\ \sin\alpha_i \sin\theta_i & -\sin\alpha_i \cos\theta_i & \cos\alpha_i & -d_i \cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.12)$$

onde a_i , α_i , d_i são constantes, e θ_i é a variável de junta para uma junta rotativa.

Para uma junta prismática a variável de junta é d_i , enquanto a_i , α_i , θ_i são constantes. Neste caso, ${}^{i-1}A_i$ será definido como:

$${}^{i-1}A_i = T_{z,\theta} T_{z,d} T_{x,\alpha} = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & 0 \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.13)$$

e sua inversa será:

$$[{}^{i-1}A_i]^{-1} = {}^iA_{i-1} = \begin{bmatrix} \cos\theta_i & \sin\theta_i & 0 & 0 \\ -\cos\alpha_i \sin\theta_i & \cos\alpha_i \cos\theta_i & \sin\alpha_i & -d_i \sin\alpha_i \\ \sin\alpha_i \sin\theta_i & -\sin\alpha_i \cos\theta_i & \cos\alpha_i & -d_i \cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.14)$$

3.8.3 – Matriz de Transformação T

A descrição cinemática completa de uma cadeia articulada pode ser obtida a partir do produto matricial entre as diversas matrizes de transformações homogêneas. Usando a matriz ${}^{i-1}A_i$, podemos relacionar um ponto X_i no link i, e expressar em coordenadas homogêneas, em relação aos sistemas de coordenadas i para i-1, X_{i-1} estabelecido no link i-1 através da relação:

$$X_{i-1} = {}^{i-1}A_i X_i \quad (3.15)$$

em que

$$X_{i-1} = (x_{i-1}, y_{i-1}, z_{i-1}) \text{ e } X_i = (x_i, y_i, z_i)^T$$

Para simplificarmos a notação, a matriz ${}^{i-1}A_i$ será designada A_i . Utilizando-se essa relação de modo recorrente podemos escrever:

$$\begin{aligned} X_{i-2} &= A_{i-2} \cdot X_{i-1} = A_{i-1} \cdot A_{i-2} \cdot X_i \\ X_{i-3} &= A_{i-2} \cdot A_{i-1} \cdot A_{i-3} \cdot X_i \\ &\dots \quad \dots \\ X_0 &= A_1 \cdot A_2 \cdot A_3 \dots A_i \cdot X_i \end{aligned} \quad (3.16)$$

Para um robô com seis graus de liberdade, a transformação de coordenadas do referencial situado na base do robô ao referencial situado no seu elemento terminal é descrito pela matriz de transformação homogênea $T_6 = A_1 \cdot A_2 \cdot A_3 \cdot A_4 \cdot A_5$

. A₆ . A figura 3.14 ilustra as coordenadas cartesianas que expressam a posição do elemento terminal de um robô (P_x , P_y , P_z) e sua orientação espacial especificada através das componentes dos versores de orientação n, s e a.

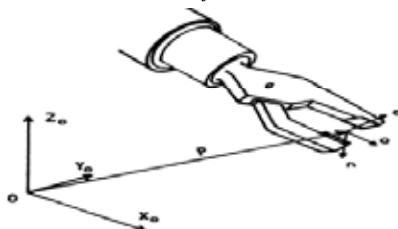
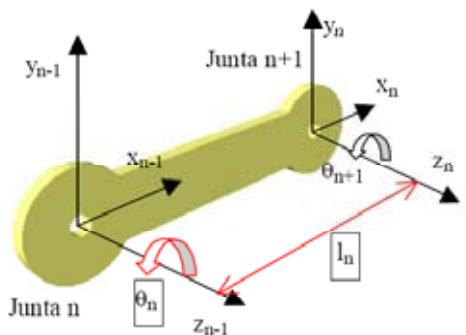


Figura 3.14: Configuração do elemento terminal de um robô.

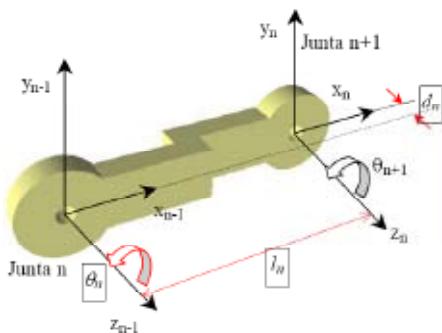
3.9 – Exemplos de obtenção dos Parâmetros de DH

Caso 1: Link com juntas rotacionais paralelas



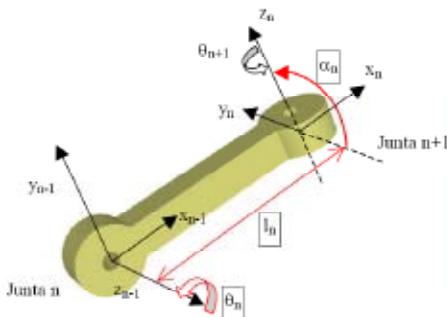
Parâmetro	Valor
l_n	$\neq 0$
d_n	$= 0$
θ_n	Variável
α_n	$= 0$

Caso 2: Link com juntas rotacionais paralelas com desalinhamento



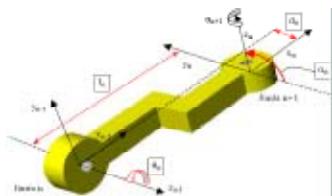
Parâmetro	Valor
l_n	$\neq 0$
d_n	$\neq 0$
θ_n	Variável
α_n	$= 0$

Caso 3: Link com juntas rotacionais ortogonais



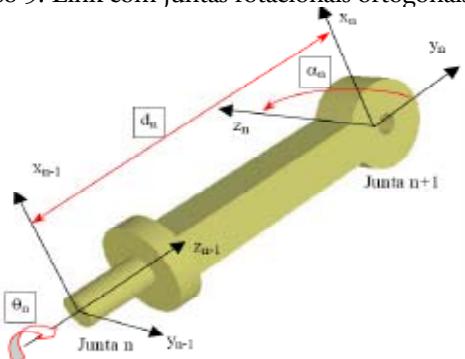
Parâmetro	Valor
l_n	$\neq 0$
d_n	$= 0$
θ_n	Variável
α_n	$\neq 0 (-90^\circ)$

Caso 4: Link com juntas rotacionais ortogonais com desalinhamento



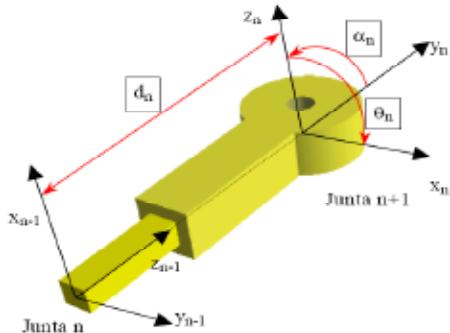
Parâmetro	Valor
l_n	$\neq 0$
d_n	$\neq 0$
θ_n	Variável
α_n	$\neq 0$ (-90°)

Caso 5: Link com juntas rotacionais ortogonais (2º tipo)



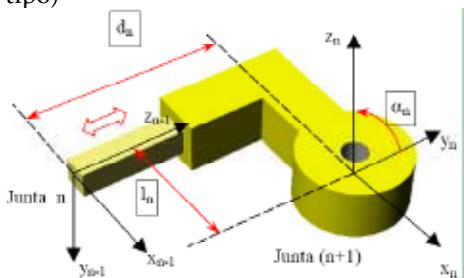
Parâmetro	Valor
l_n	$= 0$
d_n	$\neq 0$
θ_n	Variável
α_n	$\neq 0$ ($+90^\circ$)

Caso 6: Link com juntas prismáticas e rotacionais ortogonais



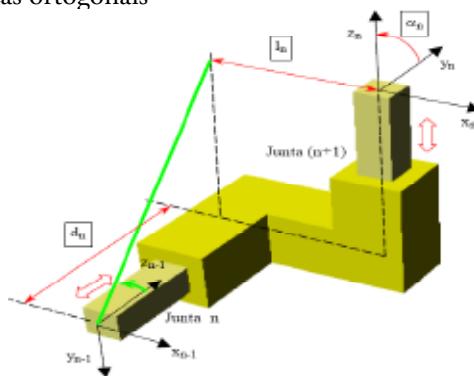
Parâmetro	Valor
l_n	= 0
d_n	$\neq 0$ (variável)
θ_n	$\neq 0$ (+90°)
α_n	$\neq 0$ (+90°)

Caso 7: Link com juntas prismáticas e rotacionais ortogonais (2º tipo)



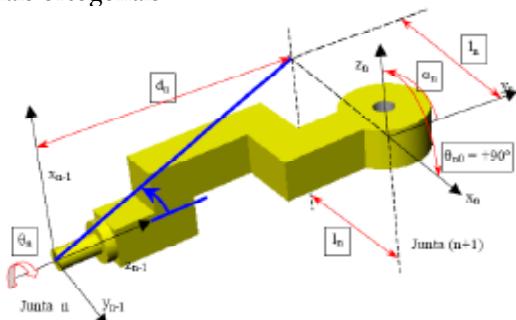
Parâmetro	Valor
l_n	$\neq 0$
d_n	variável
θ_n	= 0
α_n	$\neq 0$ (+90°)

Caso 8: Link com geometria mais elaborada e juntas prismáticas ortogonais



Parâmetro	Valor
l_n	$\neq 0$
d_n	variável
θ_n	$= 0$
α_n	$\neq 0$ ($+90^\circ$)

Caso 9: Link com geometria mais elaborada e juntas rotacionais ortogonais



Parâmetro	Valor
l_n	$\neq 0$
d_n	$\neq 0$
θ_n	$90^\circ + \text{variável}$
α_n	$\neq 0$ ($+90^\circ$)

3.10 – Modelagem Através do Método de Vetores Locais

A convenção de Denavit-Hartenberg é bastante utilizada pelos fabricantes de robôs industriais no fornecimento de parâmetros de um robô, de modo a sistematizar a entrada de dados de um modelo. Entretanto, para modelagem e simulação gráfica é necessário o conhecimento dos diversos pontos do robô, permitindo, assim, a construção de sólidos primitivos (paralelepípedos, cilindros e outros) que representam o robô e ambiente de atuação, tornando possível a execução de testes de colisão com o ambiente.

3.10.1 – Metodologia

O método consiste em determinar vetores para cada ponto de interesse de visualização e, a partir da multiplicação destes vetores por matrizes de transformação, obter as coordenadas (posição e orientação) de cada um destes pontos, como mostra a figura 3.15.

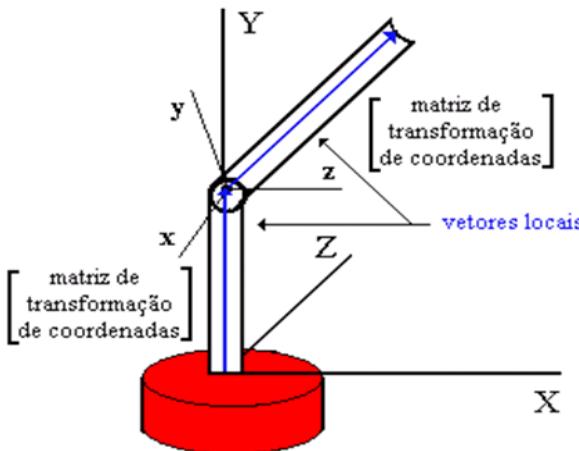


Figura 3.15: Descrição do método de vetores locais.

Portanto, para o software de programação *off-line* é

necessário que se modele o robô através de vetores locais de translação e rotação. O modelo obtido desta forma, para o robô Manutec, é apresentado a seguir. Inicialmente são determinados os vetores de rotação e as matrizes de rotação; para isso utiliza-se a figura 3.16.

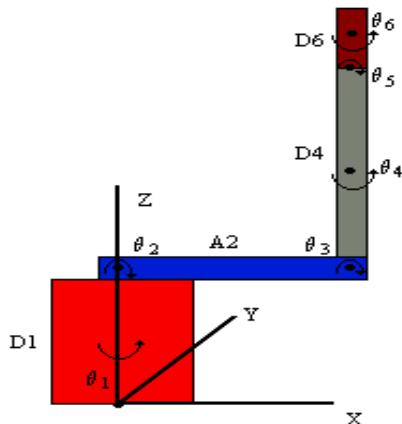


Figura 3.16: Representação do robô.

Os vetores de rotação são dados por:

$$\begin{aligned}
 \vec{V}_{r1} &= [0, 0, 1] && \text{-- rotação } R_1 \\
 \vec{V}_{r2} &= [0, 1, 0] && \text{-- rotação } R_2 \\
 \vec{V}_{r3} &= [0, 1, 0] && \text{-- rotação } R_3 \\
 \vec{V}_4 &= [0, 1, 0] && \text{-- rotação } R_4 \\
 \vec{V}_5 &= [0, 1, 0] && \text{-- rotação } R_5 \\
 \vec{V}_6 &= [0, 1, 0] && \text{-- rotação } R_6
 \end{aligned} \tag{3.17}$$

As matrizes de rotação são:

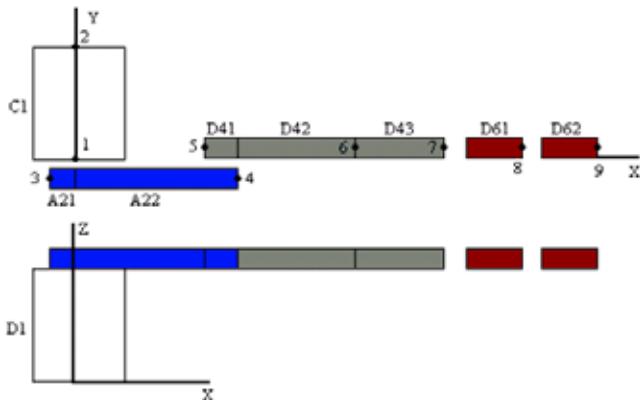
$$\begin{matrix} R_1(z, \theta_1) & R_2(z, \theta_2) & R_3(z, \theta_3) & R_4(z, \theta_4) \\ R_5(z, \theta_5) & R_6(z, \theta_6) \end{matrix}$$

em que

$$R_i(z, \theta_i) = \begin{bmatrix} c_i & -s_i & 0 \\ s_i & c_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad R_i(y, \theta_i) = \begin{bmatrix} c_i & 0 & s_i \\ 0 & 1 & 0 \\ -s_i & 0 & c_i \end{bmatrix} \quad (3.18)$$

e $c_i = \cos(\theta_i)$, $s_i = \sin(\theta_i)$ para $i = 1, 2, \dots, 6$.

A figura 3.16 mostra os pontos de visualização necessários para implementação do programa de visualização gráfica dos pontos de interesse, a partir das dimensões indicadas.



Parâmetro	dimensão (mm)
C1	370
A21	120
A22	500
A2	500
D1	665
D41	280
D42	120
D43	610
D4	730
D41	280
D61	50
D62	100

Figura 3.16: Pontos de Visualização do robô Manutec da Siemens.

A figura 3.17 mostra os vetores de translação (\vec{V}_i), especificados em relação ao referencial XYZ.

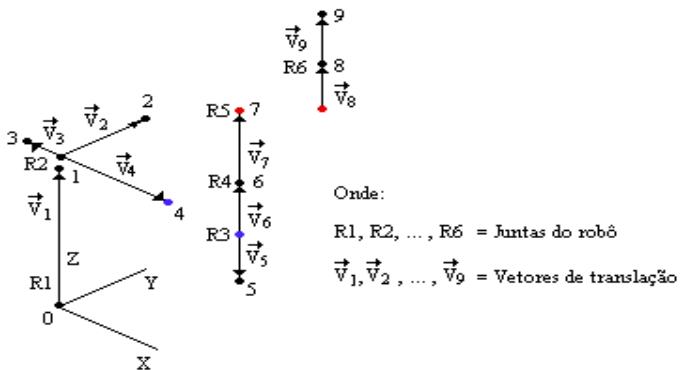


Figura 3.17: Vetores de translação.

em que

$$\vec{V}_1 = [0, 0, d_1] - R_1 \text{ entre os pontos de visualização } O_0 \text{ e } O_1 \text{ inicia-se na junta } R_1$$

$\vec{V}_2 = [0, c_1, 0] - R_2$ entre os pontos de visualização
O₁ e O₂ inicia-se na junta R₂

$\vec{V}_3 = [-a_{21}, 0, 0] - R_3$ entre os pontos de visualização
O₁ e O₃ inicia-se na junta R₂

$\vec{V}_4 = [a_{22}, 0, 0] - R_4$ entre os pontos de visualização
O₁ e O₄ inicia-se na junta R₂

$\vec{V}_5 = [0, 0, -d_{41}] - R_5$ entre os pontos de visualização
O₄ e O₅ inicia-se na junta R₃

$\vec{V}_6 = [0, 0, d_{42}] - R_6$ entre os pontos de visualização
O₄ e O₆ inicia-se na junta R₃

$\vec{V}_7 = [0, 0, d_{43}] - R_7$ entre os pontos de visualização
O₆ e O₇ inicia-se na junta R₄

$\vec{V}_8 = [0, 0, d_{61}] - R_8$ entre os pontos de visualização
O₇ e O₈ inicia-se na junta R₅

$\vec{V}_9 = [0, 0, d_{62}] - R_9$ entre os pontos de visualização
O₈ e O₉ inicia-se na junta R₆

Com o objetivo de obter a visualização do robô em vários ângulos pelo software de programação, define-se:

- M_V: Matriz de orientação com as variáveis do software que definem o ângulo de visualização gráfica.

No modelo completo poderão ser inseridos no modelo uma matriz e um vetor de correção de erros identificados representados por:

- R_{t_a}: Matriz de correção de erros de orientação identificados,
- \vec{V}_{t_a} : Vetor de correção de erros de posição identificados.

O objetivo da inserção da matriz de correção de erros de orientação e do vetor de correção de erros de posição, identificados, é de permitir posicionar o meio ambiente gerado no computador (programação “off-line”) da mesma forma que ele estará posicionado quando da colocação do robô no meio ambiente. Isto permite um alto grau de precisão de posicionamento e orientação. Campos (1993) implementou um método para a identificação desses parâmetros.

Neste estudo não se preocupou em inserir valores medidos para estes parâmetros, pois não está dentro do escopo deste trabalho. Portanto, para não interferir com o modelo, a matriz de correção de erros de orientação e o vetor de correção de erros de posição identificados são os seguintes:

$$R_{t_a} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.19)$$

$$\vec{V}_{t_a} = [0 \ 0 \ 0]$$

Definindo-se então as matrizes de transformação $M_T(i)$ associadas a cada elo, tem-se:

$$M_T(1) = M_V * R_{t_a}$$

$$M_T(2) = M_T(1) * R_1$$

$$M_T(3) = M_T(2) * R_2 = M_T(1) * R_1 * R_2$$

$$M_T(4) = M_T(3) * R_3 = M_T(1) * R_1 * R_2 * R_3$$

$$M_T(5) = M_T(4) * R_4 = M_T(1) * R_1 * R_2 * R_3 * R_4 \quad (3.20)$$

$$M_T(6) = M_T(5) * R_5 = M_T(1) * R_1 * R_2 * R_3 * R_4 * R_5$$

$$M_T(7) = M_T(6) * R_6 = M_T(1) * R_1 * R_2 * R_3 * R_4 * R_5 * R_6$$

Finalmente, determina-se a posição dos pontos de interesse, em relação ao referencial XYZ, as equações são apresentadas a seguir:

$$\begin{aligned}
 \vec{O}(0) &= M_T(1) * \vec{V}_{t-a} \\
 \vec{O}(1) &= \vec{O}(0) + M_T(2) * \vec{V}_1 \\
 \vec{O}(2) &= \vec{O}(1) + M_T(2) * \vec{V}_2 \\
 \vec{O}(3) &= \vec{O}(1) + M_T(3) * \vec{V}_3 \\
 \vec{O}(4) &= \vec{O}(1) + M_T(3) * \vec{V}_4 \\
 \vec{O}(5) &= \vec{O}(4) + M_T(4) * \vec{V}_5 \\
 \vec{O}(6) &= \vec{O}(4) + M_T(4) * \vec{V}_6 \\
 \vec{O}(7) &= \vec{O}(6) + M_T(5) * \vec{V}_7 \\
 \vec{O}(8) &= \vec{O}(7) + M_T(6) * \vec{V}_8 \\
 \vec{O}(9) &= \vec{O}(8) + M_T(7) * \vec{V}_9
 \end{aligned} \tag{3.21}$$

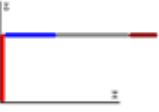
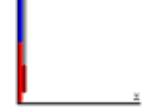
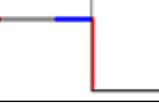
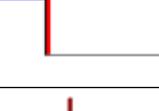
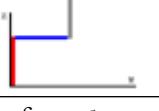
Simulação	Posição angular						Configuração do robô
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	
1	0	90	90	0	0	0	
2	0	0	90	0	0	0	
3	0	90	0	0	0	0	
4	0	-90	-90	0	0	0	
5	0	-180	-90	0	0	0	
6	0	-180	-90	0	0	0	
7	0	-180	0	0	0	0	
8	0	0	0	0	0	0	

Tabela 3.2: Valores angulares das juntas para cada configuração, para comparar o modelo geométrico obtido através dos dois métodos.

3.10.2 – Comparação do modelo cinemático com o obtido através de DH.

A partir da montagem da matriz,

$$C_{4 \times 4} = \begin{bmatrix} M_T(7)_{3 \times 3} & \vec{O}(9)_{3 \times 1} \\ \vec{0}_{1 \times 3} & 1 \end{bmatrix} \quad (3.22)$$

pode-se confirmar se os valores obtidos para a posição e orientação do elemento terminal do robô, são os mesmos em ambos os métodos. A matriz gerada pela equação 3.22 tem de ser igual a da equação obtida através dos parâmetros de DH (equação 3.16).

A partir dos valores angulares das juntas apresentados na tabela 3.2 foram realizadas as simulações para comparar o modelo geométrico obtido através dos dois métodos. Os valores obtidos para a posição e orientação foram os mesmos nos dois métodos (tabela 3.3).

Simulação	Matriz obtida
1	$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & -665 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
2	$\begin{bmatrix} 0 & 0 & 1 & 1330 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 665 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

3	$\begin{bmatrix} 0 & 0 & 1 & 830 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 165 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
4	$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 335 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
5	$\begin{bmatrix} 0 & 0 & -1 & -1330 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 665 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
6	$\begin{bmatrix} 0 & 0 & 1 & 330 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 665 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
7	$\begin{bmatrix} -1 & 0 & 0 & -500 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & -165 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
8	$\begin{bmatrix} 1 & 0 & 0 & 500 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1495 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Tabela 3.3: Valores obtidos para a posição e orientação para cada simulação.

3.11 – Obtenção da Matriz de Orientação

A orientação de um sistema de coordenadas (em relação à ferramenta terminal, TCP) pode ser descrito através de uma matriz de rotação que descreve a direção dos eixos do sistema de coordenados associados em relação ao sistema de referência do robô. A rotação dos eixos associados ao sistema de coordenadas (x , y , z) são vetores que expressam as coordenadas associadas ao sistema na seguinte forma:

$$\begin{aligned} \mathbf{x} &= (\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z) \\ \mathbf{y} &= (\mathbf{s}_x, \mathbf{s}_y, \mathbf{s}_z) \\ \mathbf{z} &= (\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z) \end{aligned} \quad (3.23)$$

O significado de cada componente x do vetor no sistema de coordenadas de referência será \mathbf{n}_x , a componente y será \mathbf{s}_x , etc. Estes três vetores podem ser dispostos em cada coluna de uma matriz, designada de matriz de rotação.

$$\begin{bmatrix} \mathbf{n}_x & \mathbf{s}_x & \mathbf{a}_x \\ \mathbf{n}_y & \mathbf{s}_y & \mathbf{a}_y \\ \mathbf{n}_z & \mathbf{s}_z & \mathbf{a}_z \end{bmatrix} \quad (3.24)$$

O uso dos vetores \vec{n} , \vec{s} e \vec{a} como vetores de direção, para descrever a orientação do elemento terminal do robô, além de ser de difícil compreensão, apresenta muitos termos para serem processados matematicamente.

Consequentemente, torna-se interessante a descrição da matriz de orientação espacial, composta pelos componentes dos vetores \vec{n} , \vec{s} e \vec{a} através de parâmetros constantes: três ângulos (ψ , θ e ϕ) ou quaternions (q_1 , q_2 , q_3 e q_4). Isto implicará que o posicionamento do elemento terminal do robô será descrito por um vetor de seis ou sete dimensões. Esta represen-

tação torna mais simples a utilização da posição e orientação do robô para a geração de uma trajetória.

Normalmente, em aplicações industriais são utilizados ângulos Euler ou RPY (Roll, Pitch e Yaw) para descrição da orientação de um corpo rígido no espaço, e mais recentemente se tem adotado a utilização de quartenions de orientação que apresentam inúmeras vantagens em relação aos ângulos de orientação.

3.11.1 - Ângulos de Euler

Os ângulos de Euler são obtidos a partir de três rotações elementares ψ , θ , ϕ em torno dos eixos Z, Y, Z (figura 3.18). Estas transformações devem ser biunívocas.

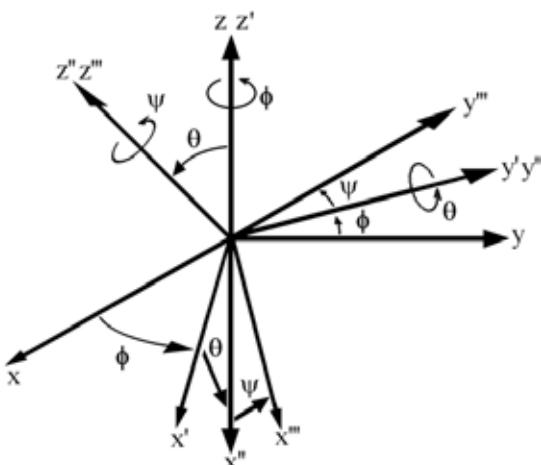


Figura 3.18: Ângulos de Euler.

Para que isso ocorra, a definição dos valores dos ângulos ψ , θ , ϕ deve ser realizada a partir da utilização da função ATAN2, definida na tabela 3.4.

Função ATAN2			
$\theta = \left[\begin{array}{c} x \\ y \end{array} \right] =$	$0 \leq \theta \leq 90$, com +x, +y	
	$90 \leq \theta \leq 180$, com -x, +y	
	$-180 \leq \theta \leq -90$, com -x, -y	
	$-90 \leq \theta \leq 0$, com +x, -y	

Tabela 3.4: Definição da função ATAN2.

$$\text{EULER}(\psi, \theta, \phi) = \text{ROT}(z, \psi) \cdot \text{ROT}(y, \theta) \cdot \text{ROT}(z, \phi)$$

$$\text{EULER}(\psi, \theta, \phi) = \begin{bmatrix} C\psi C\theta C\phi - S\psi S\phi & -C\psi C\theta S\psi - S\psi C\phi & S\theta C\psi \\ S\psi S\theta C\phi - C\psi S\phi & -S\psi C\theta S\phi + C\psi C\phi & S\psi S\theta \\ -S\theta C\phi & S\theta S\phi & C\theta \end{bmatrix} \quad (3.25)$$

em que

$$\begin{aligned} \psi &= \text{ATAN2}\left[\frac{a_z}{-a_y}\right] \\ \theta &= \text{ATAN2}\left[\frac{S\psi a_x - C\psi a_y}{a_z}\right] \\ \psi &= \text{ATAN2}\left[\frac{-C\psi a_x - S\psi S_y}{C\psi n_x + S\psi n_y}\right] \end{aligned}$$

A partir dos ângulos de Euler (ψ, θ, ϕ) podemos obter a matriz de orientação, conforme mostra a figura 3.19, cuja solução inversa (obtenção dos três ângulos de Euler a partir do conhecimento da matriz de orientação) deverá ser biunívoca.

Figura 3.19: Obtenção da Matriz de orientação a partir dos três ângulos de Euler.

$$\begin{matrix}
 & \begin{bmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{bmatrix} \\
 \uparrow & \psi = \text{ATAN2}\left[\frac{a_x}{-a_y}\right] \\
 \uparrow & \theta = \text{ATAN2}\left[\frac{s_y a_x - c_y a_y}{a_z}\right] \\
 \uparrow & \text{EULER}(\psi, \theta, \phi) = \begin{bmatrix} c_y c_\theta c_\phi - s_y c_\theta s_\phi & -c_y c_\theta s_\phi - s_y c_\phi & s_\theta c_\psi \\ s_y c_\theta c_\phi - c_y s_\theta s_\phi & -s_y c_\theta s_\phi + c_y c_\phi & s_\theta s_\psi \\ -s_y s_\theta & s_y s_\theta c_\phi & c_\theta \\ s_y s_\theta c_\phi + c_y s_\theta s_\phi & s_y s_\theta s_\phi + c_y c_\theta s_\phi & c_\theta s_\psi \\ -s_y s_\theta c_\phi & s_y s_\theta s_\phi & c_\theta s_\psi \end{bmatrix}
 \end{matrix}$$

3.11.2 - Ângulos de RPY

Os ângulos de Roll Pitch e Yaw (RPY) são obtidos a partir de três rotações elementares ψ , θ , ϕ em torno dos eixos X, Y, Z (3.20). Estes ângulos são muito utilizados em aplicações aeronáuticas. Estes são os três graus de liberdade possíveis associados ao movimento do punho, respectivamente os ângulos de rotação do punho ao redor dos eixos X, Y e Z (figura 3.21).

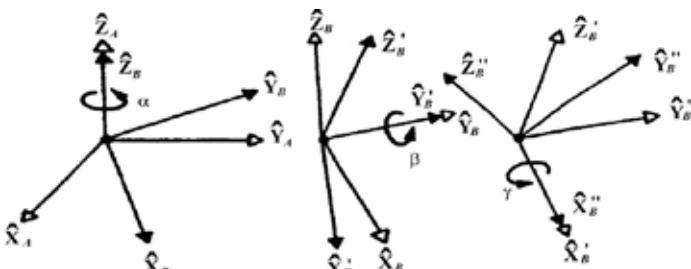


Figura 3.20: Ângulos de Roll, Pitch, Yaw – Rotações Elementares.

$$RPY(\phi, \theta, \psi) = \text{ROT}(z, \phi) \cdot \text{ROT}(y, \theta) \cdot \text{ROT}(z, \psi)$$

$$RPY(\phi, \theta, \psi) = \begin{bmatrix} C\phi C\theta & -C\phi S\theta S\psi - S\phi C\psi & C\phi S\theta C\psi + S\phi S\psi \\ S\phi C\theta & -S\phi S\theta S\psi + C\phi C\psi & S\phi S\theta C\psi - C\phi S\psi \\ -S\theta & C\theta S\psi & C\theta C\psi \end{bmatrix} \quad (3.26)$$

onde,

$$\phi = \text{ATAN2}\left[\frac{n_y}{n_z}\right]$$

$$\theta = \text{ATAN2}\left[\frac{-n_z}{C\phi n_x + S\phi n_y}\right]$$

$$\psi = \text{ATAN2}\left[\frac{S\phi n_x - C\phi n_y}{-S\phi n_x + C\phi n_y}\right]$$

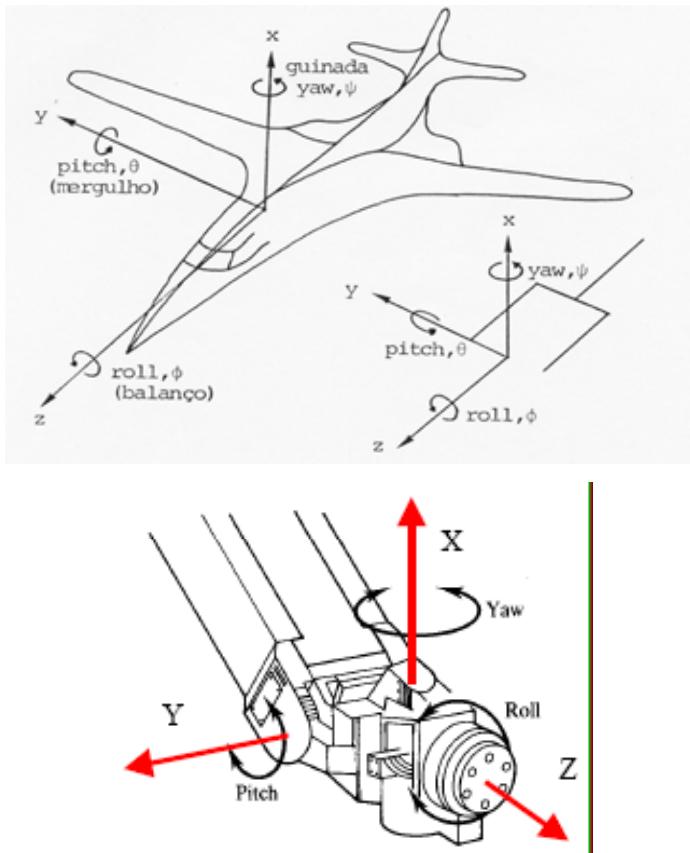


Figura 3.21: Aplicações utilizando os ângulos de rotação Roll, Picth e Yaw.

A partir dos ângulos RPY (ψ , θ , ϕ) podemos obter a matriz de orientação, conforme mostra a figura 3.21, cuja solução inversa (obtenção dos três ângulos de RPY a partir do conhecimento da matriz de orientação) deverá ser biunívoca.

Figura 3.22: Obtenção da Matriz de orientação
a partir dos três ângulos RPY.

$$\begin{aligned}
 (\Psi, \theta, \phi) &\xrightarrow{\text{CALCUL}} \text{RPY}(\phi, \theta, \psi) = \begin{bmatrix} c\phi c\theta & c\phi s\theta s\psi - s\phi c\psi & c\phi s\theta c\psi + s\phi s\psi \\ s\phi c\theta & s\phi s\psi s\theta s\psi + c\phi c\psi & s\phi s\theta c\psi - c\phi s\psi \\ -s\theta & c\theta s\psi & c\theta c\psi \end{bmatrix} \xrightarrow{\text{S ANO}}
 \begin{bmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{bmatrix} \\
 &\quad \uparrow \quad \uparrow \quad \uparrow \\
 \phi &= \text{ATAN} 2 \left[\frac{n_y}{n_x} \right] \quad \theta = \text{ATAN} 2 \left[\frac{-n_z}{c\phi n_x + s\phi n_y} \right] \quad \psi = \text{ATAN} 2 \left[\frac{s\phi a_x - c\phi a_y}{-s\phi s_x + c\phi s_y} \right] \\
 &\quad \uparrow \quad \uparrow \quad \uparrow \\
 &\quad \text{CALCULO} \quad \text{DE MATRIZ DE ORIENTAÇÃO} \quad \text{DE MATRIZ DE ORIENTAÇÃO}
 \end{aligned}$$

3.11.3 – Exemplo

Muitos robôs, entre eles o Manutec da Siemens, utilizam os ângulos Roll, Pitch e Yaw (RPY) para expressarem a orientação do elemento terminal em relação às coordenadas da base. Para isto basta efetuar três rotações sucessivas em relação aos eixos X, Y e Z, obtendo-se uma matriz correspondente a essa transformação de coordenadas.

Começando com os dois sistemas de coordenadas A e B coincidentes, inicia-se fazendo-se uma rotação no sistema de coordenadas B sobre \hat{X}_A de um ângulo ψ , depois sobre \hat{Y}_A de um ângulo θ e então sobre \hat{Z}_A de um ângulo ϕ , figura 3.23.

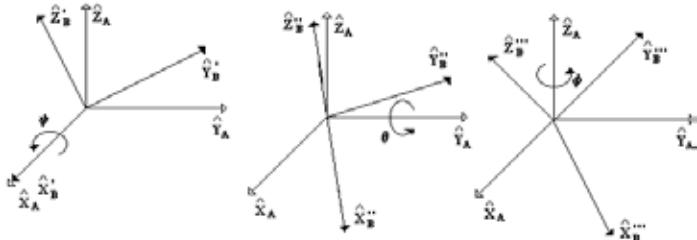


Figura 3.23: Rotações dos eixos do sistema de coordenadas B.

Portanto, a matriz de orientação é dada por:

$$\text{RPY}(\psi, \theta, \phi) = \text{ROT}(z, \phi) * \text{ROT}(y, \theta) * \text{ROT}(z, \psi)$$

$$= \begin{bmatrix} c\phi \, c\theta & c\phi \, s\theta \, s\psi - s\phi \, c\psi & c\phi \, s\theta \, c\psi + s\phi \, s\psi \\ s\phi \, c\theta & s\phi \, s\theta \, s\psi + c\phi \, c\psi & s\phi \, s\theta \, c\psi - c\phi \, s\psi \\ -s\theta & c\theta \, s\psi & c\theta \, c\psi \end{bmatrix} \quad (3.27)$$

Para se obter os três ângulos a partir da matriz de orientação, o problema inverso, analisa-se os seus termos. Os casos possíveis, a partir desta observação, serão descritos a seguir. A matriz de orientação pode ser escrita da seguinte forma:

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \quad (3.28)$$

- caso 1

$$\begin{aligned} \text{se } m_{31} = -1 \Rightarrow \theta = 90^\circ \\ \text{se } m_{31} = 1 \Rightarrow \theta = -90^\circ \end{aligned}$$

- casos 2 a 4

$$\text{se } m_{21} > 0$$

caso 2 $\Rightarrow \phi = -90^\circ, \psi = -90^\circ \text{ e } \theta = \text{atan2}(-m_{31}, -m_{21});$

caso 3 $\Rightarrow \phi = -90^\circ, \psi = 90^\circ \text{ e } \theta = \text{atan2}(-m_{31}, -m_{21});$

caso 4 $\Rightarrow \phi = -90^\circ, \psi = \text{atan2}(-m_{13}, m_{12}) \text{ e } \theta = \text{atan2}(-m_{31}, -m_{21}).$

$$- \quad \text{se } m_{21} > 0$$

caso 2 $\Rightarrow \phi = 90^\circ, \psi = 90^\circ \text{ e } \theta = \text{atan2}(-m_{31}, m_{21});$

caso 3 $\Rightarrow \phi = 90^\circ, \psi = -90^\circ \text{ e } \theta = \text{atan2}(-m_{31}, m_{21});$

caso 4 $\Rightarrow \phi = 90^\circ, \psi = \text{atan2}(m_{13}, -m_{12}) \text{ e } \theta = \text{atan2}(-m_{31}, m_{21}).$

- caso 5

se $\text{abs}(m_{33}) > 0 \text{ e } \text{abs}(m_{32}) > 0 \Rightarrow$

$\phi = \text{atan2}(m_{21}, -m_{11}), \psi = \text{atan2}(m_{32}, -m_{33}) \text{ e } \theta = \text{atan2}(-m_{31}, \text{sen}(\psi)).$

se $m_{33} = 0 \text{ e } m_{32} > 0 \Rightarrow \phi = \text{atan2}(m_{13}, -m_{23}), \psi = 90^\circ \text{ e } \theta = \text{atan2}(-m_{31}, m_{32}).$

se $m_{33} = 0 \text{ e } m_{32} < 0 \Rightarrow \phi = \text{atan2}(-m_{13}, m_{23}), \psi = -90^\circ \text{ e } \theta = \text{atan2}(-m_{31}, -m_{32}).$

- se $m_{32} = 0 \Rightarrow \phi = \text{atan2}(-m_{12}, m_{22}), \psi = 0^\circ$ e $\theta = \text{atan2}(-m_{31}, m_{33})$.

- se $m_{32} \neq 0 \Rightarrow \phi = \text{atan2}(-m_{12}, m_{22}), \psi = 0^\circ$ e $\theta = \text{atan2}(-m_{31}, m_{33})$.

3.11.4 – Quartenions

Outra forma bastante utilizada para a resolução do problema de orientação e cinemática inversa de robôs é a utilização de *quaternions*, que permite escrever de forma concisa a matriz de orientação. Eles são calculados a partir do conhecimento dos elementos da matriz de orientação.

A partir da utilização da álgebra de *quaternions*, a matriz de orientação da ferramenta de um manipulador (3.24), expressa em relação ao sistema de coordenadas da base, é expressa a partir de quatro parâmetros (q_1, q_2, q_3 e q_4).

$$\begin{aligned}
 q_1 &= \frac{1}{2} \sqrt{n_x + s_y + a_z + 1} \\
 q_2 &= \frac{1}{2} \sqrt{n_x - s_y - a_z + 1} && \text{com sinal de } q_2 = \text{sinal}(s_z - a_y) \\
 q_3 &= \frac{1}{2} \sqrt{s_y - n_x - a_z + 1} && \text{com sinal de } q_3 = \text{sinal}(a_x - n_z) \\
 q_4 &= \frac{1}{2} \sqrt{a_z - n_x - s_y + 1} && \text{com sinal de } q_4 = \text{sinal}(n_y - s_x)
 \end{aligned} \tag{3.29}$$

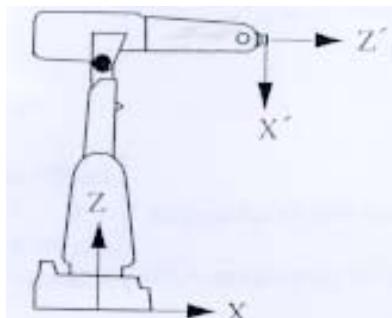
Ao mesmo tempo a orientação deverá ser normalizada, ou seja:

$$q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1 \tag{3.30}$$

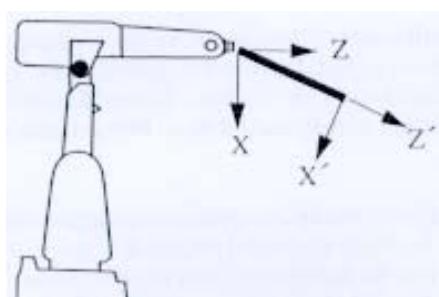
A principal vantagem dessa representação é que a utilização de quatro parâmetros permitirá a obtenção de soluções únicas, implicando assim em um número menor de manipulações computacionais.

3.11.5 - Exemplo de Aplicação

A figura 3.24 apresenta a ferramenta de um robô industrial orientada (referencial X' Y' Z') em relação ao sistema de referência fixo a base (X, Y, Z). Para as duas configurações abordadas, apresentaremos a seguir o cálculo da matriz de orientação da ferramenta, ângulos RPY e *quaternions* (q_1, q_2, q_3, q_4):



a) Rotação de 90° em torno de Y



b) Nova Rotação de 30° em torno de Y

Figura 3.24: Cálculo da Matriz de Orientação,

ângulos RPY e *quaternions*.

Solução:

Caso (a): Os eixos de referência podem ser descritos da seguinte forma:

$$X' = -Z = (0, 0, -1)$$

$$Y' = Y = (0, 1, 0)$$

$$Z' = X = (1, 0, 0)$$

Matriz de orientação:

$$T = \begin{bmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$$

Ângulos RPY: $(0, 90^\circ, 0)$ – rotação de 90° do eixo Y no sentido horário.

Quaternions:

$$q_1 = \frac{1}{2} \sqrt{0+1+0+1} = \frac{1}{2} \sqrt{2} = 0,707$$

$$q_2 = \frac{1}{2} \sqrt{0-1-0+1} = 0$$

$$q_3 = \frac{1}{2} \sqrt{1-0-0+1} = \frac{1}{2} \sqrt{2} = 0,707$$

$$q_4 = \frac{1}{2} \sqrt{0-0-1+1} = 0$$

com sinal $q_3 = \text{sinal } (1 + 1) = +$

Caso (b): Os eixos de referência podem ser descritos da seguinte forma:

$$X' = (\cos 30^\circ, 0, -\sin 30^\circ)$$

$$Y' = (0, 1, 0)$$

$$Z' = (\sin 30^\circ, 0, \cos 30^\circ)$$

Matriz de orientação:

$$T = \begin{bmatrix} \cos 30^\circ & 0 & \sin 30^\circ \\ 0 & 1 & 0 \\ -\sin 30^\circ & 0 & \cos 30^\circ \end{bmatrix}$$

Ângulos RPY: $(0, 30^\circ, 0)$ – rotação de 30° do eixo Y no sentido horário.

Quaternions:

$$q_1 = \frac{1}{2} \sqrt{\cos 30^\circ + 1 + \cos 30^\circ + 1} = 0,9659$$

$$q_2 = \frac{1}{2} \sqrt{\cos 30^\circ - 1 - \cos 30^\circ + 1} = 0$$

$$q_3 = \frac{1}{2} \sqrt{1 - \cos 30^\circ - \cos 30^\circ + 1} = 0,2588$$

$$q_4 = \frac{1}{2} \sqrt{0 - 0 - 1 + 1} = 0$$

com sinal $q_3 = \text{sinal } (\sin 30^\circ + \sin 30^\circ) = +$

3.12 - Referências Bibliográficas

CLOSE, C. M., FREDERICK, D.K.: “Modeling and Analysis of Dynamic Systems”, Houghton Mifflin Company, 1989.

CRUZ, J.M.: “Projeto e Desenvolvimento de um Sistema de Geração Automática de Trajetória para Manipuladores”, Dissertação de Mestrado, Unicamp, 1993.

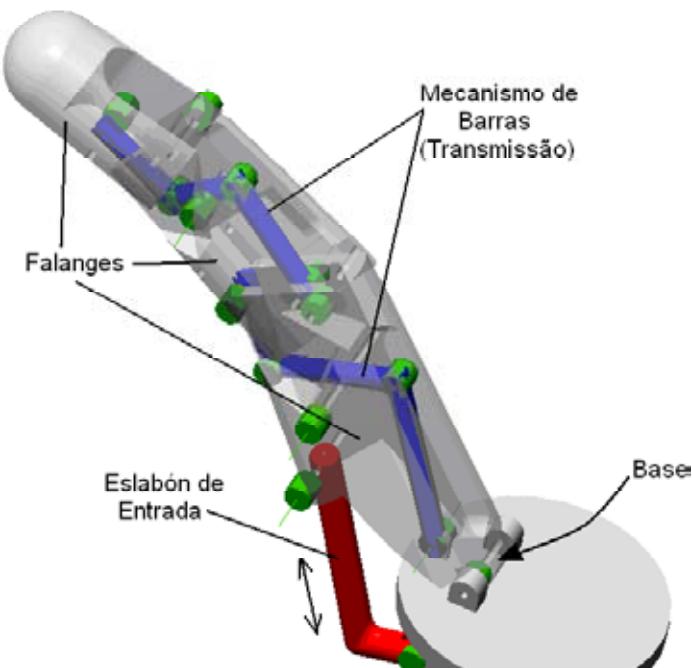
DENAVIT, J., HARTENBERG, R.: “A kinematic notation for lower-pair mechanisms based on matrices”, ASME J. on Applied Mechanics, 1955, pp. 215-221.

PAUL, P.: “Robot Manipulators: Mathematics, Programming and Control”, The Mit Press, 1981.

RAPID: “Reference Manual, Características Gerais”, *ABB Flexible Automation, 1999*

CAPÍTULO 4

Modelagem Cinemática Exemplos Práticos



CAPÍTULO 4

Modelagem Cinemática Exemplos Práticos

Neste capítulo são apresentados alguns exemplos práticos para modelagem cinemática direta de manipuladores utilizando os conceitos apresentados nos capítulos anteriores. Para melhor compreensão do leitor foram escolhidos cinco robôs industriais muito utilizados em instalações produtivas.

4.1 – Robô Stanford

Em 1969, Victor Scheinman, no Laboratório de Inteligência Artificial (IA) da Universidade de Stanford, nos Estados Unidos, desenvolveu um braço robótico articulado de seis graus de liberdade (5 juntas de rotação e 1 de translação), totalmente elétrico, projetado de modo a permitir um solução utilizando a anatomia de um braço.

A partir de um sistema de visão, este robô conseguia localizar objetos e se deslocar até os mesmos conseguia localizar objetos e se deslocar até os mesmos. Isto permitiu que o robô fosse capaz de seguir precisamente caminhos arbitrários no espaço e aumentou as possibilidades de utilizar robôs em aplicações mais sofisticadas tais como montagem e soldagem.

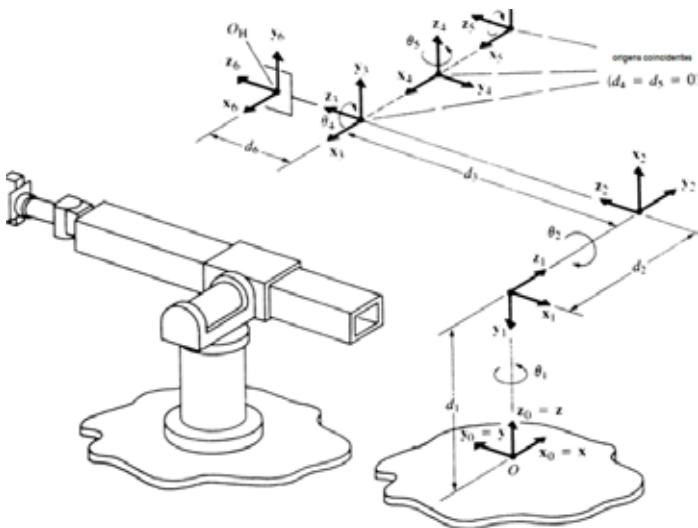


Figura 4.1: Robô Stanford.

4.1.1 - Parâmetros de Denavit-Hartenberg (DH)

Junta i	θ (graus)	α (graus)	a (mm)	d (mm)
1	-90°	-90°	0	d_1
2	-90°	90°	0	d_2
3	-90°	0	0	d_3
4	0	-90°	0	0
5	0	90°	0	0
6	0	0	0	d_6

4.1.2 - Matrizes de Transformação Homogênea (MTH)

A partir da equação 3.11, definida anteriormente, temos as seguintes matrizes de transformação de coordenadas para o robô StandfordTM:

$${}^0 A_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^1 A_2 = \begin{bmatrix} 0 & 0 & -1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^2 A_3 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^3 A_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^4 A_5 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^5 A_6 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.1.3 – Matriz posição-orientação final 0T6

$${}^0 T_6 = {}^0 A_1 \cdot {}^1 A_2 \cdot {}^2 A_3 \cdot {}^3 A_4 \cdot {}^4 A_5 \cdot {}^5 A_6 = \begin{bmatrix} 1 & 0 & 0 & d_2 \\ 0 & 0 & 1 & d_3 + d_6 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.2 – Robô Puma 560TM

Trata-se de um robô articulado com seis graus de liberdade (6 juntas de rotação) totalmente elétrico, projetado de modo a permitir um solução utilizando a anatomia de um braço. Foi também desenvolvido por Scheinman no Laboratório de Inteligência Artificial (IA) do MIT AI Lab, nos Estados Unidos, sendo vendido posteriormente a UNIMATION, a qual o desenvolveu com o auxílio da General Motors e posteriormente o comercializou como a Máquina Programável Universal para Montagem (PUMA).

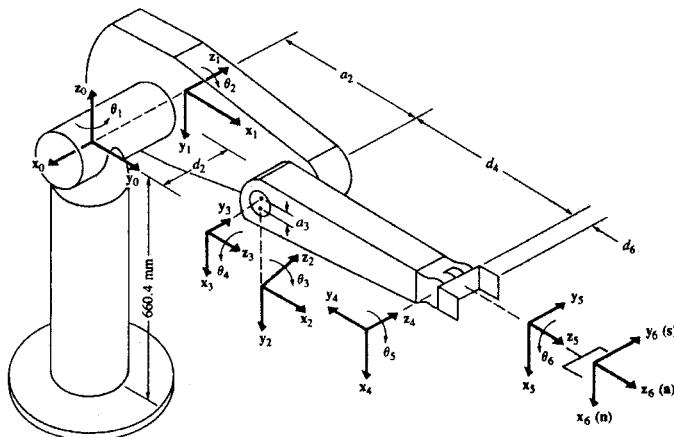


Figura 4.2: Robô Puma 560TM.

4.2.1 - Parâmetros de Denavit Hartenberg (DH)

Junta i	θ (graus)	d (mm)	α (graus)	a (mm)	range (graus)
1	90°	0	-90°	0	-160/+160
2	0	149.09	0	431.8	-225/+ 45
3	90°	0	90°	-20.32	- 45/+225
4	-90°	433.07	-90°	0	-110/+170
5	90°	0	90°	0	-100/+100
6	0	56.25	0	0	-266/+266

4.2.2 - Matrizes de Transformação Homogênea (MTH)

$$\begin{aligned} {}^0A_1 &= \begin{bmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} & {}^1A_2 &= \begin{bmatrix} c_2 & -s_2 & 0 & a_2c_2 \\ s_2 & c_2 & 0 & a_2s_2 \\ 0 & 0 & 1 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^2A_3 &= \begin{bmatrix} c_3 & 0 & s_3 & a_3c_3 \\ s_3 & 0 & -c_3 & a_3s_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} & {}^3A_4 &= \begin{bmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^4A_5 &= \begin{bmatrix} c_5 & 0 & s_5 & 0 \\ s_5 & 0 & -c_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} & {}^5A_6 &= \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

(*) C_i e S_i denotam $\cos \theta_i$ e $\sin \theta_i$, respectivamente.

4.2.3 – Cálculo do Vetor de Posição e da Matriz de Orientação

4.2.3.1 - Matriz posição-orientação final 0TN

a) Cálculo do posicionamento (três primeiros graus de liberdade)

$${}^0T_3 = {}^0A_1 \cdot {}^1A_2 \cdot {}^2A_3 = \begin{bmatrix} c_1c_3 & -s_1 & c_1s_3 & a_2c_1c_2 + a_3c_1c_3 & -d_2s_1 \\ s_1c_3 & c_1 & s_1s_3 & a_2s_1c_2 + a_3s_1c_3 & +d_2c_1 \\ -s_3 & 0 & c_3 & -a_2s_2 - a_3s_3 & \\ 0 & 0 & 0 & & 1 \end{bmatrix}$$

b) Cálculo da orientação (três últimos graus de liberdade)

$${}^3T_6 = {}^3A_4 \cdot {}^4A_5 \cdot {}^5A_6 = \begin{bmatrix} c_4c_5c_6 - s_4s_6 & -c_4c_5s_6 - s_4c_6 & c_4s_5 & d_6c_4s_5 \\ s_4c_5c_6 + c_4s_6 & -s_4c_5s_6 + c_4c_6 & s_4s_5 & d_6s_4s_5 \\ -s_5c_6 & s_5s_6 & c_5 & d_6c_5 + d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(*) C_{ij} e S_{ij} denotam $\cos(\theta_i + \theta_j)$ e $\sin(\theta_i + \theta_j)$, respectivamente.

4.2.3.2 - Orientação final (elemento terminal)

$${}^0T_6 = {}^0T_3 \cdot {}^3T_6 = {}^0A_1 \cdot {}^1A_2 \cdot {}^2A_3 \cdot {}^3A_4 \cdot {}^4A_5 \cdot {}^5A_6 = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

com

$$n_x = c_1 [c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6] - s_1(s_4c_5c_6 + c_4s_6)$$

$$n_y = s_1 [c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6] - c_1(s_4c_5c_6 + c_4s_6)$$

$$n_z = -s_{23}[c_4c_5c_6 - s_4s_6] - c_{23}s_5c_6$$

$$s_x = c_1[-c_{23}(c_4c_5s_6 + s_4c_6) - s_{23}s_5s_6] - s_1(-s_4c_5s_6 + c_4c_6)$$

$$s_y = s_1[-c_{23}(c_4c_5s_6 + s_4c_6) - s_{23}s_5s_6] + c_1(-s_4c_5s_6 + c_4c_6)$$

$$s_z = s_{23}[c_4c_5s_6 - s_4c_6] - c_{23}s_5s_6$$

$$a_x = c_1(c_{23}c_4s_5 + s_{23}c_5) - s_1s_4s_5$$

$$a_y = s_1(c_{23}c_4s_5 + s_{23}c_5) - c_1s_4s_5$$

$$a_z = -s_{23}c_4s_5 + c_{23}c_5$$

4.2.3.3 - Posição final (elemento terminal)

$$p_x = c_1[d_6(c_{23}c_4s_5 + s_{23}c_5) + s_{23}d_4 + a_3c_{23} + a_2c_2] - s_1(d_6s_4s_5 + d_2)$$

$$p_y = s_1[d_6(c_{23}c_4s_5 + s_{23}c_5) + s_{23}d_4 + a_3c_{23} + a_2c_2] + c_1(d_6s_4s_5 + d_2)$$

$$p_z = d_6(c_{23}c_5 - s_{23}c_4s_5) + c_{23}d_4 - a_3s_{23} - a_2s_2$$

4.2.3.4 - Verificação do modelo

Considere os seguintes parâmetros para verificação do modelo $\theta_1 = 90^\circ$, $\theta_2 = 0$, $\theta_3 = 90^\circ$, $\theta_4 = 0$, $\theta_5 = 0$ e $\theta_6 = 0$, a matriz posição-orientação final 0T_6 será a seguinte:

$${}^0T_6 = \begin{bmatrix} 0 & -1 & 0 & -149.09 \\ 0 & 0 & 1 & 921.12 \\ -1 & 0 & 0 & 20.32 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

que corresponde a configuração do robô representada anteriormente.

4.3 – Robô Manutec r3TM

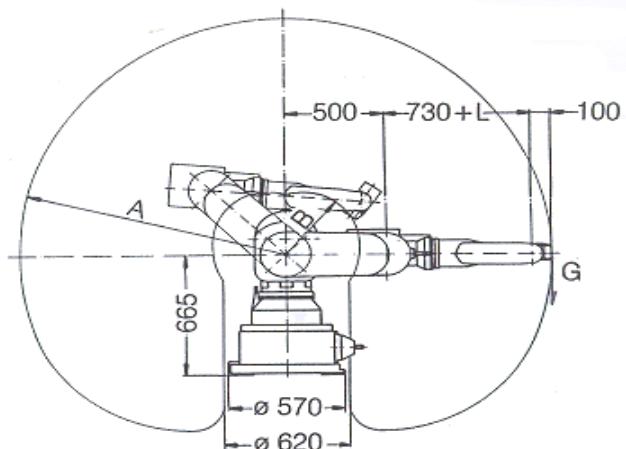
Este robô articulado, desenvolvido pela Siemens, possui seis juntas de rotação totalmente elétrico. Os três primeiros graus de liberdade servem para o seu posicionamento, enquanto os três outros graus são destinados a orientação da ferramenta.

menta de trabalho.



Figura 4.3: Robô Manutec r3TM.

A figura 4.4 apresenta o volume de trabalho (*workspace*) para o robô MANUTECTM. A seguir são apresentados os parâmetros de Denavit-Hartenberg relacionados com os diferentes graus de liberdade deste robô e matrizes de passagem obtidas a partir destes parâmetros.



L	A	B	G [kg]
0 mm	R 1330	R 380	15
100	R 1430	R 470	12

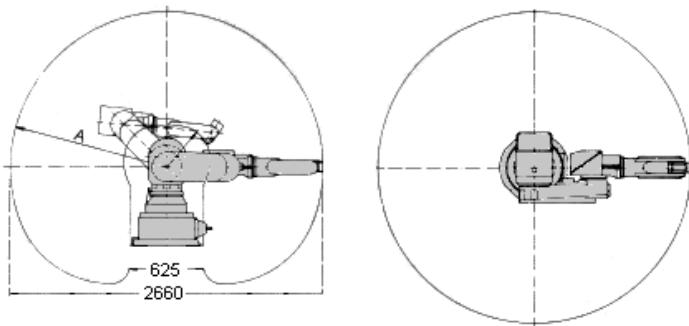
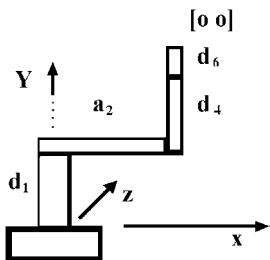


Figura 4.4: Dimensões e Volume de Trabalho do Robô Manutec r3.

4.3.1 - Parâmetros de Denavit-Hartenberg (DH)

Junta	θ (graus)	d (mm)	α (graus)	a (mm)	range (graus)
1	θ_1	665.0	-90.0	0.0	± 165
2	θ_2	0.0	0.0	500.0	-20 / +220
3	θ_3	0.0	90.0	0.0	-225 / +45
4	θ_4	730.0	-90.0	0.0	± 190
5	θ_5	0.0	90.0	0.0	± 120
6	θ_6	100.0	0.0	0.0	± 265

Representação



4.3.2 - Matrizes de Transformação Homogênea (MTH)

$${}^0 A_1 = \begin{bmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^1 A_2 = \begin{bmatrix} c_2 & -s_2 & 0 & a_2 c_2 \\ s_2 & c_2 & 0 & a_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^2 A_3 = \begin{bmatrix} c_3 & 0 & s_3 & 0 \\ s_3 & 0 & -c_3 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^3 A_4 = \begin{bmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^4A_5 = \begin{bmatrix} c_5 & 0 & s_5 & 0 \\ s_5 & 0 & -c_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^5A_6 = \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Onde: $C_i = \cos(\theta_i)$ e $S_i = \sin(\theta_i)$ para $i = 1, 2, \dots, 6$

4.3.3 – Cálculo do Vetor de Posição e da Matriz de Orientação

As equações obtidas para o robô são apresentadas abaixo:

$$T_6(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) = A_{0,1} * A_{1,2} * A_{2,3} * A_{3,4} * A_{4,5} * A_{5,6}$$

$$T_6(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) = \begin{bmatrix} \vec{n} & \vec{s} & \vec{a} & \vec{p} \\ 0 & 1 \end{bmatrix}$$

em que $\vec{p} = [p_x, p_y, p_z]$: vetor posição;

$\vec{n} = [n_x, n_y, n_z]$, $\vec{s} = [s_x, s_y, s_z]$ e $\vec{a} = [a_x, a_y, a_z]$ vetores ortonormais que descrevem a orientação.

4.3.4 – Cálculo Matriz Posição-Orientação Final 0T_N

$${}^0T_6 = {}^0T_3 \cdot {}^3T_6 = {}^0A_1 \cdot {}^1A_2 \cdot {}^2A_3 \cdot {}^3A_4 \cdot {}^4A_5 \cdot {}^5A_6 = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

com:

Orientação:

$$n_x = ((c_1c_{23}c_4 - s_1s_4)c_5 - c_1s_{23}s_5)c_6 + (-c_1c_{23}s_4 - s_1c_4)s_6$$

$$n_y = ((s_1c_{23}c_4 - c_1s_4)c_5 - s_1s_{23}s_5)c_6 + (-s_1c_{23}s_4 - c_1c_4)s_6$$

$$n_z = (-s_{23}c_4c_5 - c_{23}s_5)c_6 + s_{23}s_4c_6$$

$$s_x = -((c_1 c_{23} c_4 - s_1 s_4) s_5 - c_1 s_{23} s_5) s_6 + (-c_1 c_{23} s_4 - s_1 c_4) c_6$$

$$s_y = -((s_1 c_{23} c_4 - c_1 s_4) c_5 - s_1 s_{23} s_5) s_6 + (-s_1 c_{23} s_4 - c_1 c_4) c_6$$

$$s_z = (s_{23} c_4 c_5 - c_{23} s_5) s_6 + s_{23} s_4 c_6$$

$$a_x = (c_1 c_{23} c_4 - s_1 s_4) s_5 + c_1 s_{23} c_5$$

$$a_y = (s_1 c_{23} c_4 - c_1 s_4) s_5 + s_1 s_{23} c_5$$

$$a_z = -s_{23} c_4 s_5 + c_{23} c_5$$

Posição final (elemento terminal)

$$p_x = c_1 [d_6 (c_{23} c_4 s_5 + s_{23} c_5) + s_{23} d_4 + a_3 c_{23} + a_2 c_2] - s_1 (d_6 s_4 s_5 + d_2)$$

$$p_y = s_1 [d_6 (c_{23} c_4 s_5 + s_{23} c_5) + s_{23} d_4 + a_3 c_{23} + a_2 c_2] + c_1 (d_6 s_4 s_5 + d_2)$$

$$p_z = d_6 (c_{23} c_5 - s_{23} c_4 s_5) + c_{23} d_4 - a_3 s_{23} - a_2 s_2$$

$$p_x = ((c_4 s_5 d_6) c_{23} + (c_5 d_6 + d_4) s_{23} + a_2 c_2) c_1 - s_4 s_5 d_6 s_1$$

$$p_y = ((c_4 s_5 d_6) c_{23} + (c_5 d_6 + d_4) s_{23} + a_2 c_2) s_1 - s_4 s_5 d_6 c_1$$

$$p_z = -(c_4 s_5 d_6) s_{23} + (c_5 d_6 + d_4) c_{23} - a_2 s_2 + d_1$$

4.4 – Manipulador Submarino Kraft

O manipulador Kraft possui seis juntas rotacionais e foi desenvolvido para executar tarefas gerais em ambientes hostis e submarinos (figura 4.6). Os seus movimentos são comandados a distância através de um controle chamado “*master*” que é um modelo em escala reduzida do manipulador. Suas trajetórias podem ser definidas pelo operador ou por programações pré-definidas. O sistema completo robô tele-operado é denominado sistema *master-slave*.



a) Mestre (*Master*)



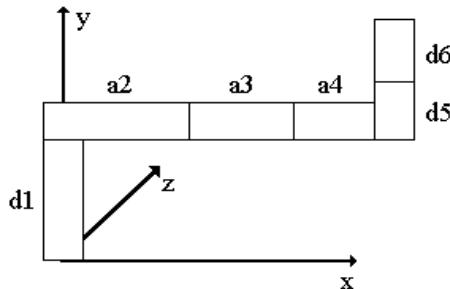
b) Escravo (*Slave*)

Figura 4.5: Manipulador Submarino Kraft™.

4.4.1 - Parâmetros de Denavit-Hartenberg (DH)

Junta i	θ (graus)	α (graus)	a (mm)	d (mm)	range (graus)
1	θ_1	90	0	d_1	-90/+90
2	θ_2	0	a_2	0.0	0/+120
3	θ_3	0	a_3	0.0	0/-130
4	θ_4	-90°	a_4	0.0	-42/+58
5	θ_5	-90°	0	d_5	+34/+134
6	θ_6	0	0	d_6	-90/+90

Representação



Parâmetros:

$$d_1 = 352.43 \text{ mm}, d_5 = 48.06 \text{ mm}, d_6 = 50 \text{ mm} \\ a_2 = 532.65 \text{ mm}, a_3 = 264.32 \text{ mm}, a_4 = 132.16 \text{ mm}$$

4.4.2 - Matrizes de Transformação Homogênea (MTH)

$${}^0 A_1 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^1 A_2 = \begin{bmatrix} c_2 & -s_2 & 0 & a_2 c_2 \\ s_2 & c_2 & 0 & a_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^1 A_2 = \begin{bmatrix} c_2 & -s_2 & 0 & a_2 c_2 \\ s_2 & c_2 & 0 & a_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^3 A_4 = \begin{bmatrix} c_4 & 0 & -s_4 & a_4 c_4 \\ s_4 & 0 & c_4 & a_4 s_4 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^4 A_5 = \begin{bmatrix} c_5 & 0 & s_5 & 0 \\ s_5 & 0 & -c_5 & 0 \\ 0 & 1 & 0 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^5 A_6 = \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(*) C_i e S_i denotam $\cos \theta_i$ e $\sin \theta_i$, respectivamente.

4.4.3 – Cálculo do Vetor de Posição e da Matriz de Orientação

4.4.3.1 - Orientação final

$$\begin{aligned}n_x &= c_1 (c_5 c_6 c_{234} - s_6 s_{234}) - s_1 s_5 c_6, \\n_y &= s_1 (c_5 c_6 c_{234} - s_6 s_{234}) + c_1 s_5 c_6, \\n_z &= c_5 c_6 s_{234} - s_6 c_{234},\end{aligned}$$

$$\begin{aligned}s_x &= -c_1 (c_5 s_6 c_{234} + c_6 s_{234}) + s_1 s_5 s_6, \\s_y &= -s_1 (c_5 s_6 c_{234} + c_6 s_{234}) - c_1 s_5 s_6, \\s_z &= -c_5 s_6 s_{234} + c_6 c_{234},\end{aligned}$$

$$\begin{aligned}a_x &= c_1 s_5 c_{234} + s_1 c_5, \\a_y &= s_1 s_5 c_{234} - c_1 c_5, \\a_z &= s_5 s_{234}.\end{aligned}$$

4.4.3.2 - Posição final

$$\begin{aligned}p_x &= d_6 (c_1 s_5 c_{234} + s_1 c_5) + c_1 (-d_5 s_{234} + a_4 c_{234} + a_3 c_{23} + a_2 c_2), \\p_y &= d_6 (s_1 s_5 c_{234} - c_1 c_5) + s_1 (-d_5 s_{234} + a_4 c_{234} + a_3 c_{23} + a_2 c_2), \\p_z &= d_6 s_5 s_{234} + d_5 c_{234} + a_4 s_{234} + a_3 s_{23} + a_2 s_2 + d_1.\end{aligned}$$

em que

$$\begin{aligned}c_{23} &= c_2 c_3 - s_2 s_3, \quad s_{23} = s_2 c_3 + s_3 c_2, \\c_{234} &= c_{23} c_4 - s_{23} s_4, \quad s_{234} = c_{23} s_4 - s_{23} c_4\end{aligned}$$

4.5 – Robô ABB IRB-1400TM

O robô ABB IRB 1400TM é um robô de seis graus de liberdade, desenvolvido pela empresa sueca Asea Brown Boveri (ABB), que se tornou a primeira empresa no mundo a vender 100.000 robôs, e cuja as aplicações respondem por 30% dos

robôs vendidos no mundo todo. O maior mercado para destes robôs destina-se atualmente para tarefas de soldagem, principalmente na indústria automobilística.

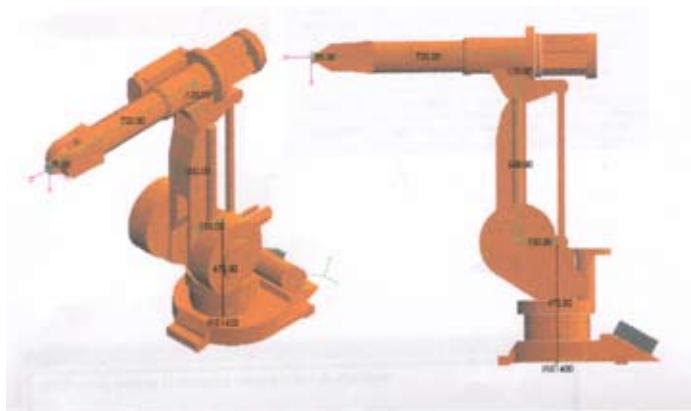


Figura 4.6: Robô ABB IRB1400TM.

4.5.1 - Parâmetros de Denavit-Hartenberg (DH) e *Workspace*

4.5.1.1 - Parâmetros de DH

Junta	θ (graus)	d (mm)	α (graus)	a (mm)
θ_1	-360°	475	-90°	0
θ_2	-90°	0	0°	360
θ_3	180°	0	90°	0
θ_4	0°	720	-90°	0
θ_5	0°	0	90°	0
θ_6	0°	85	0°	0

4.5.1.2 - Limites angulares e velocidades/accelerações mínimas e máximas

Junta	θ_{\min} (graus)	θ_{\max} (graus)	W_{\max} (graus/s)	α_{\max} (graus/s ²)
θ_1	-170°	170°	120	670
θ_2	-70°	70°	120	660
θ_3	-65°	70°	120	1130
θ_4	-150°	150°	280	3290
θ_5	-115°	115°	280	3290
θ_6	-300°	300°	280	3290

4.5.2 - Matrizes de Transformação Homogênea (MTH)

$${}^0 A_1 = \begin{bmatrix} c_1 & 0 & -s_1 & a_1 c_1 \\ s_1 & 0 & c_1 & a_1 s_1 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^1 A_2 = \begin{bmatrix} c_2 & -s_2 & 0 & a_2 c_2 \\ s_2 & c_2 & 0 & a_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^2 A_3 = \begin{bmatrix} c_3 & 0 & s_3 & a_3 c_3 \\ s_3 & 0 & -c_3 & a_3 s_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^3 A_4 = \begin{bmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^4 A_5 = \begin{bmatrix} c_5 & 0 & s_5 & 0 \\ s_5 & 0 & -c_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^5 A_6 = \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(*) c_i e s_i denotam $\cos \theta_i$ e $\sin \theta_i$, respectivamente.

4.5.3 – Cálculo do Vetor de Posição e da Matriz de Orientação

A seguir apresentaremos os procedimentos utilizados para obtenção e implementação computacional da matriz de transformação homogênea obtida, para obtenção do modelo cinemático final para o robô em estudo. A Tabela 3.6 apresenta um resumo das variáveis de entrada utilizada.

4.5.3.1 – Definição de Parâmetros e Variáveis

a) Entradas das juntas

$$C1 = \cos(\text{TET}(1)) \quad S1 = \sin(\text{TET}(1))$$

$$C2 = \cos(\text{TET}(2)) \quad S2 = \sin(\text{TET}(2))$$

$$C3 = \cos(\text{TET}(3)) \quad S3 = \sin(\text{TET}(3))$$

$$C4 = \cos(\text{TET}(4)) \quad S4 = \sin(\text{TET}(4))$$

$$C5 = \cos(\text{TET}(5)) \quad S5 = \sin(\text{TET}(5))$$

$$C6 = \cos(\text{TET}(6)) \quad S6 = \sin(\text{TET}(6))$$

$$C23 = \cos(\text{TET}(2) + \text{TET}(3))$$

$$S23 = \sin(\text{TET}(2) + \text{TET}(3))$$

b) Variáveis Auxiliares

$$\text{VAR1} = C1 * S23$$

$$\text{VAR10} = C4 * \text{VAR3} - S4 * S1$$

$$\text{VAR2} = S1 * S23$$

$$\text{VAR11} = C4 * \text{VAR4} + S4 * C1$$

$$\text{VAR3} = C1 * C23$$

$$\text{VAR12} = -S4 * \text{VAR3} - C4 * S1$$

$$\text{VAR4} = S1 * C23$$

$$\text{VAR13} = -S4 * \text{VAR4} + C4 * C1$$

$$\text{VAR14} = C4 * S23$$

$$\text{VAR15} = S4 * S23$$

c) Parâmetros Geométricos

A1 = 200.0 mm

A2 = 600.0 mm

A3 = -115.0 mm

D1 = 585.0 mm

D4 = -770.0 mm

D6 = -105.0 mm

4.5.3.2 – Vetor Orientação e Quaternions associados

a) Componentes do Vetor Orientação

$$nx = C6 * (C5 * VAR10 - S5 * VAR1) + S6 * VAR12$$

$$ny = C6 * (C5 * VAR11 - S5 * VAR2) + S6 * VAR13$$

$$nz = C6 * (-C5 * VAR14 - S5 * C23) + S6 * VAR15$$

$$sx = -S6 * (C5 * VAR10 - S5 * VAR1) + C6 * VAR12$$

$$sy = -S6 * (C5 * VAR11 - S5 * VAR2) + C6 * VAR13$$

$$sz = -S6 * (-C5 * VAR14 - S5 * C23) + C6 * VAR15$$

$$ax = S5 * VAR10 + C5 * VAR1$$

$$ay = S5 * VAR11 + C5 * VAR2$$

$$az = -S5 * VAR14 + C5 * C23$$

b) Quaternions de Orientação

$$p1, p2, p3, p4 = MATRIX_QUARTERNION (MATTRIZ')$$

$$(nx, sx, ax),$$

$$(ny, sy, ay),$$

$$(nz, sz, az)$$

c) Cálculo da Matriz de Orientação a partir dos Quaternions

MAT = QUATERNION_MATRIX (MATRIX-QUATERNION)

nx = MAT (1 , 1) sx = MAT (1 , 2) ax = MAT (1 , 3)
ny = MAT (2 , 1) sy = MAT (2 , 2) ay = MAT (2 , 3)
nz = MAT (3 , 1) sz = MAT (3 , 2) az = MAT (3 , 3)

4.5.4 – Posição Final

Px = (a3 * VAR3 + C1 * (a2 * C2 + a1) + d4 * VAR1) + d6 * Ax
Py = (a3 * VAR4 + C1 * (a2 * C2 + a1) + d4 * VAR2) + d6 * Ay
Pz = (-a3 * S23 - a2 * S2 + d1 + d4 * C23) + d6 * Az

4.5.5 – Implementação Computacional do Modelo Cinemático

-- Posição Final

px := px - d6 * ax ; py := py - d6 * ay ; pz := pz - d6 * az ;

-- Cálculo de TET1

TET (1) := ATAN2 (BASIS * py , BASIS * px) ;
S1 := SIN (TET (1)) ; C1 := COS (TET (1)) ;

VAR0 := pz - D1 ;
VAR1 := px * px + py * py ;
VAR01 := BASIS * SQRT (VAR1) ;
VAR03 := D4 ;
VAR2 := VAR03 * VAR03 ;
VAR3 := A2 * A2 ;
VAR02 := VAR1 + VAR0 * VAR0 - (VAR2 + VAR3);

-- Cálculo de TET3

```
TET (3) := ATAN2 ( VAR02 , ARM * ( SQRT ( 4 *  
VAR2 * VAR3 - VAR02 * VAR02 ) ) );  
S3 := SIN ( TET (3) ) ; C3 := COS ( TET (3) ) ;
```

```
VAR5 := VAR03 * S3 + A2 ;  
VAR6 := VAR03 * C3 ;
```

-- Cálculo de TET2

```
TET (2) := ATAN2 ( - VAR0 * VAR5 + VAR6 * VAR01,  
VAR0 * VAR6 + VAR5 * VAR01 ) ;  
S2 := SIN ( TET (2) ) ; C2 := COS ( TET (2) ) ;
```

```
VAR7 := COS ( TET (2) + TET (3) ) ;  
VAR8 := SIN ( TET (2) + TET (3) ) ;  
VAR05 := C1 * VAR7 ;  
VAR06 := S1 * VAR7 ;  
VAR07 := C1 * VAR8 ;  
VAR08 := S1 * VAR8 ;
```

-- Cálculo de TET4

```
TET (4) := ATAN2 ( C1 * AY - S1 * AX ,  
VAR05 * AX + VAR06 * AY - VAR8 * AZ ) ;  
S4 := SIN ( TET (4) ) ; C4 := COS ( TET (4) ) ;
```

```
VAR10 := VAR7 * C4 ;  
VAR11 := VAR7 * S4 ;  
VAR12 := VAR8 * C4 ;  
VAR13 := VAR8 * S4 ;
```

-- Cálculo de TET5

```
TET (5) := ATAN2 (
    (C1 * VAR10 - S1 * S4) * Ax + (S1 * VAR10 + C1 *
S4) * Ay - VAR12 * Az ,
    S5 := SIN ( TET (5) ) ; C5 := COS ( TET (5) ) ;

    VAR07 * Ax + VAR08 * Ay + VAR7 * Az ) ;
```

-- Cálculo de TET6

```
TET (6) := ATAN2 (
    (-C1 * VAR11 - S1 * C4) * Nx + (-S1 * VAR11 +
C1 * C4) * Ny + VAR13 * Nz ,
    (-C1 * VAR11 - S1 * C4) * Sx + (-S1 * VAR11 +
C1 * C4) * Sy + VAR13 * Sz ) ;
    S6 := SIN ( TET (6) ) ; C6 := COS ( TET (6) ) ;
```

-- Critério de Solução Indeterminação Cinemática

```
if ABS (TET (5)) < 0.0000000001 then
    if (ABS (TET (4))) > 0.01 AND ( ABS (TET (6))
) > 0.01 then
        TET (4) := TET (4) + PI ;
        TET (6) := TET (6) + PI ;
        TET (5) := -TET (5) ;
    end if ;
end if ;
```

4.6 - Referências Bibliográficas

COUTINHO, L.: “Um Ambiente Integrado de Desenvolvimento de Software a Robótica”, Dissertação de Mestrado, Unicamp, 1993.

CRUZ, J.M.: “Projeto e Desenvolvimento de um Sistema de Geração Automática de Trajetória para Manipuladores”, Dissertação de Mestrado, Unicamp, 1993.

JACOBSEN, T. K.: “Visualização e Geração de Trajetórias de Robôs a Partir da Utilização do Software WORKSPACE”, programa IAESTE (Brasil-Dinamarca), 1991.

KRAFT TELEROBOTICS: “Underwater Manipulator System”, 1985.

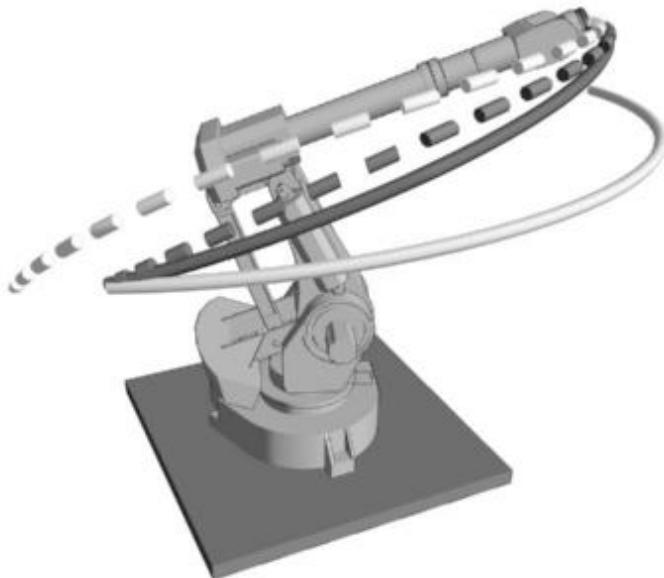
NOGUEIRA, R.: “Controle de Posição e Orientação de um Manipulador através de um Mouse Espacial”. Reinaldo Gonçalves Nogueira, Dissertação de Mestrado, Unicamp, 1995.

PAUL, R.P.: “Robot Manipulators: Mathematics, Programming and Control”, The MIT Press, 1981.

RAPID: “Reference Manual: Características Gerais”, *ABB Flexible Automation, 1999*

CAPÍTULO 5

Modelagem Cinemática Inversa



CAPÍTULO 5

Modelagem Cinemática Inversa

Na maioria das aplicações industriais é necessário que um robô opere de acordo com a posição e orientação do seu elemento terminal, e deste modo é preciso resolver o problema cinemático inverso para determinar o deslocamento de junta necessário para o manipulador atingir o objetivo desejado, necessitando assim da implementação de algoritmos para inversão do modelo geométrico.

Neste capítulo, inicialmente são apresentados de modo suscinto e informativo métodos clássicos para resolução da problema cinemático inverso baseados na configuração geométrica, seguido da metodologia para obtenção e cálculo do Jacobiano, muito utilizado na indústria robótica, para resolução do problema cinemático inverso de manipuladores industriais, sendo apresentado algoritmos numéricos para cálculo do modelo cinemático inverso, para a geração de uma trajetória angular necessária para o movimento das juntas de um manipulador para alcançar um dado objetivo. A apresentação é direcionada a manipuladores robóticos antropomórficos com juntas rotacionais, permitindo que o mesmo possa se deslocar de uma posição e orientação iniciais a uma posição e orientação final desejada. Todos estes procedimentos apresentados poderão ser estendidos para outros tipos de robôs.

No final deste capítulo são apresentados alguns exemplos práticos da utilização desta metodologia em robôs industriais.

5.1 - Introdução

A coordenação de movimentos, que consiste na obtenção de um movimento de referência (angular) para cada junta, para um dado movimento de referência do elemento terminal (cartesiano), é expressa matematicamente pela inversão do modelo geométrico.

A necessidade da obtenção de referências em coordenadas angulares, correspondentes a tarefas definidas no espaço cartesiano, é expressa matematicamente pela inversão do modelo geométrico, isto é:

$$\underline{\theta} = \underline{f}^{-1} (\underline{x}) \quad (5.1)$$

A função \underline{f} é não linear e composta da soma de produtos das coordenadas generalizadas das ligações de translação e de somas e produtos de senos e cossenos das coordenadas generalizadas das ligações de rotação. Por isso, a sua inversão é, em geral, não trivial. Craig, 1986 e Paul, 1981 apresentam para a inversão do modelo geométrico.

Como \underline{f} é não linear não se pode garantir a existência e/ou a unicidade de uma função inversa \underline{f}^{-1} . No caso geral, só se pode determinar o número máximo de prováveis soluções. Os métodos de solução do problema da inversão do modelo geométrico são:

- *Métodos analíticos*
- *Métodos numéricos iterativos.*

5.1.1 - Métodos analíticos

Estes métodos conduzem à obtenção de todas as soluções. Eles não são gerais, isto é, a inversão analítica não é trivial e, além disso, não há garantia de que seja possível fazê-la para um robô qualquer. Os métodos analíticos são adequados para robôs simples, ou seja, aqueles que possuem um grande

número de parâmetros de Denavit-Hartenberg nulos (Ferreira 1991). Estes métodos podem conduzir a soluções múltiplas como pode ser observado na figura 3.13.

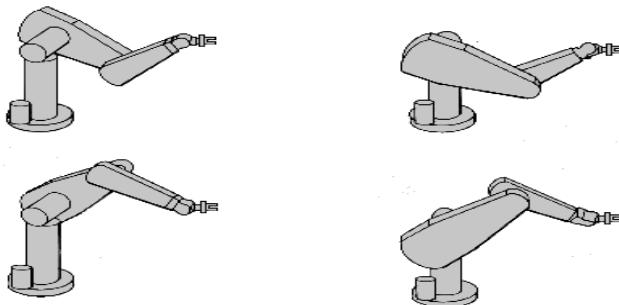


Figura 5.1: Configurações múltiplas para um robô.

5.1.2 - Métodos numéricos iterativos

Estes métodos convergem para uma solução possível entre todas as existentes; são de caráter geral e com o atual desenvolvimento dos microcomputadores a utilização desses métodos em tempo real é viável. Diferentemente das soluções analíticas, as soluções numéricas podem ser combinadas com estratégias de anticolisão com objetos, em que as posições e as dimensões dos mesmos são conhecidas.

O interessante nestes métodos é que os pontos gerados a cada iteração podem ser enviados diretamente para o controlador do robô como um ponto de referência. Existem diversos métodos numéricos iterativos, entre eles o *método recursivo* (Gorla, 1984), figura 5.2, que utiliza ao cálculo do modelo geométrico direto e da matriz Jacobiana inversa para determinar uma configuração $\underline{\theta}^*$ correspondente a uma situação desejada \underline{x}^* .

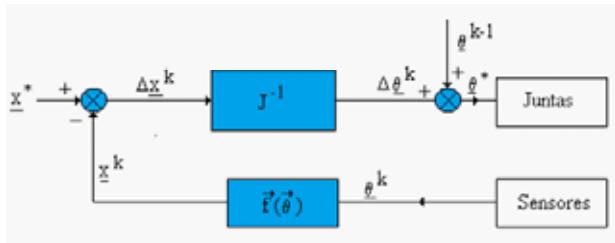


Figura 5.2: Método de Inversão Numérica para cálculo de posicionamento.

5.2 - Descrição Matemática de um Robô com N GL

A transformação de coordenadas de um robô com N graus de liberdade pode ser formulada a partir de uma configuração inicial do robô, na qual a suas variáveis articulares $\underline{\theta}_o$ são conhecidas e a posição completa de seu elemento terminal (ou ferramenta de trabalho) \underline{x}_o será conhecida a partir do modelo do sistema.

A mudança de coordenadas consistirá de um funcional que descreverá a correspondência existente entre a cadeia cinemática para um conjunto de variáveis articulares $\underline{\theta}$ e sua posição \underline{x} correspondente.

$$\underline{x} - \underline{x}_o = F(\underline{\theta} - \underline{\theta}_o) \quad (5.2)$$

em que o vetor F possuí $N \leq 6$ componentes descrevendo a posição e orientação do elemento terminal do robô (no caso de $N = 6$).

No caso da transformação inversa de coordenadas, uma determinada posição \underline{x} do volume de trabalho do robô será atingida pelo robô a partir de uma posição de repouso \underline{x}_o (equação 5.3). Esta equação não apresentará uma solução única, e a mesma poderá ser utilizada para o controle cinemático de mecanismos.

$$(\underline{\theta} - \underline{\theta}_o) = F^{-1}(\underline{x} - \underline{x}_o) \quad (5.3)$$

A transformação direta de coordenadas não apresenta dificuldades na sua resolução, o mesmo não acontece com a transformação inversa que é muito complexa, não apresentando uma solução única. Para eliminarmos as indeterminações que aparecem no problema inverso, utiliza-se geralmente a matriz jacobiana, em que a mesma poderá ser utilizada para o controle cinemático de mecanismos.

5.3 – Procedimento para obtenção do Modelo Cinemático Inverso

Para a obtenção de uma trajetória angular a partir da posição e orientação do elemento terminal, é necessário em primeiro lugar obter o modelo geométrico do robô e, após isso, obter o modelo geométrico inverso.

O modelo geométrico pode ser obtido para um robô utilizando o método de Denavit-Hartenberg ou o método de vetores locais, e a partir do modelo cinemático direto é gerado o modelo geométrico inverso e a matriz Jacobiana.

A orientação do elemento terminal de um robô é conseguida a partir da matriz de orientação que pode ser obtida a partir dos três ângulos: roll, pitch e yaw (que descrevem a orientação do robô em relação às coordenadas da base) e da obtenção destes três ângulos a partir da matriz orientação (problema inverso).

5.3.1 – Modelagem Cinemática Direta

Explicitando a condição de cinemática direta, faremos as condições da transformação homogênea de um sistema robô com “n” juntas, definindo a posição e orientação do TCP:

$$\begin{aligned} T(q) &= \begin{bmatrix} n(q) & s(q) & a(q) & p(q) \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= {}^0_1T(q_1), {}^1_2T(q_2), \dots, {}^{n-2}_{n-1}T(q_{n-1}), {}^{n-1}_nT(q_n) \end{aligned} \quad (5.4)$$

Os vetores \mathbf{n} , \mathbf{s} , \mathbf{a} são ortonormais e descrevem a orientação do TCP no sistema de referência da base (mundo). O vetor \mathbf{p} descreve a translação do TCP do sistema de referência da base a sua posição atual. O reconhecimento da coordenada do robô em relação ao seu referencial zero possui uma solução única. \mathbf{T} descreve uma transformação homogênea 4×4 e é calculado como o produto de ${}^{i-1}_iT(\mathbf{q}_i)$.

Ele descreve uma transformação homogênea para cada link i até o seu anterior link $i-1$ e são somente dependentes do valor da junta \mathbf{q}_i deste link. São chamados parâmetros de Denavit Hartenberg (D-H), usados para descrever uma transformação de coordenadas de um referencial associado a uma determinada junta até a consecutiva junta. Estas transformações são chamadas matrizes de transformação de coordenadas.

5.3.2 – Modelagem Cinemática Inversa

Para resolver a cinemática inversa da equação 5.4, temos que obter a solução de valores de \mathbf{q} para obter $\mathbf{T}(\mathbf{q})$, fornecendo a possibilidade de obtermos 16 valores numéricos, em que quatro dos quais são soluções triviais.

Resolvendo a equação (9.1) para os “N” ângulos da junta q_1, \dots, q_n entre as nove equações obtidas a partir da matriz de rotação partindo de \mathbf{T} , somente três equações são independentes. Essas somas com três equações a partir da posição do vetor parte de \mathbf{T} fornecem seis equações com “N” desconhecidos. Estas equações são **não lineares**, transcendentais de difícil solução e geralmente com singularidades. Métodos diferenciais podem ser usados dentro de tais casos: o jacobiano de um robô especifica um reconhecimento a partir das velocidades internas das juntas espaciais até a velocidade no espaço cartesiano.

$$d\mathbf{x} = J(\mathbf{q}) d\mathbf{q} \quad (5.5)$$

A natureza deste reconhecimento muda com os valores das juntas de cada robô. A ideia básica para resolver a cinemática inversa é até interativamente determinar o erro cartesiano entre posição cartesiana desejada e a atual (incluindo a orientação) pelo significado de cinemática direta e até interpretar este erro como uma velocidade cartesiana.

A equação (5.5) é resolvida pela \mathbf{dq} pela pseudo-inversão da matriz jacobiana para determinar a correspondência da velocidade das juntas. Isto é repetido de forma que a posição e orientação cartesiana sejam bastante precisas.

O passo crucial é o cálculo da pseudo-inversa da matriz jacobiana. Os métodos de solução do problema da inversão do modelo geométrico são geralmente analíticos e numéricos interativos. Os analíticos conduzem à obtenção de todas as soluções, estes métodos não são triviais e, além disso, não há garantia de que seja possível fazê-la. Normalmente utiliza-se em robôs que possuem um grande número de parâmetros de “D-H” nulos.

5.3.3 – Matriz Jacobiana

Dado uma configuração inicial θ^0 e X^0 de um robô, as coordenadas X do elemento terminal são descritas pela equação 5.1. Considerando uma trajetória composta de pequenos deslocamentos $\delta\mathbf{x}$ associados aos pequenos deslocamentos das variáveis articulares $\delta\theta$ podemos escrever:

$$\delta\theta = J^{-1} \delta\mathbf{x} \quad (5.6)$$

em que

n: número de graus de liberdade do robô (coordenadas articulares)

m: número de graus de liberdade consideradas no espaço de trabalho (coordenadas cartesianas)

A matriz Jacobiana $J(\theta)$ será definida como

$$[J(\theta)_{i,j}] = [\partial F_i / \partial \theta_j] \quad (5.7)$$

que poderá ser construída a partir das relações cinemáticas que descrevem a arquitetura de um manipulador:

$$\begin{aligned} X_1 &= F_1(\theta_1, \theta_2, \dots, \theta_n) \\ X_2 &= F_2(\theta_1, \theta_2, \dots, \theta_n) \\ &\dots \quad \dots \dots \dots \\ X_n &= F_n(\theta_1, \theta_2, \dots, \theta_n) \end{aligned} \quad (5.8)$$

Através de derivadas parciais, a matriz Jacobiana $J(\theta)$ será definida como:

$$J(\theta) = \begin{bmatrix} \frac{\partial F_1}{\partial \theta_1} & \frac{\partial F_1}{\partial \theta_2} & \dots & \dots & \frac{\partial F_1}{\partial \theta_n} \\ \frac{\partial F_2}{\partial \theta_1} & \frac{\partial F_2}{\partial \theta_2} & \dots & \dots & \frac{\partial F_2}{\partial \theta_n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \frac{\partial F_n}{\partial \theta_1} & \frac{\partial F_n}{\partial \theta_2} & \dots & \dots & \frac{\partial F_n}{\partial \theta_n} \end{bmatrix} \quad (5.9)$$

5.3.4 – Exemplos

Nesta seção retornaremos aos exemplos apresentados no capítulo 3, referente ao modelo cinemático de manipuladores com 1 GL e 2 GL. A obtenção da matriz Jacobiana, referente a esses dois manipuladores, são apresentadas nos exemplos descritos a seguir.

5.3.4.1 – Manipulador 1 GL

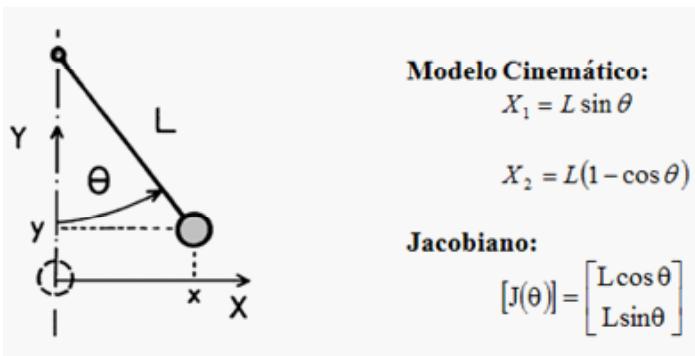


Figura 5.3: Pêndulo simples com 1GL.

5.3.4.2 – Manipulador 2 GL

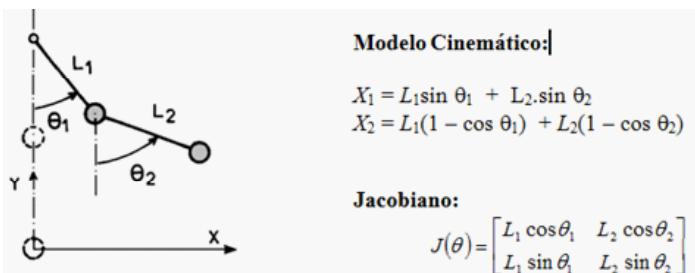


Figura 5.4: Pêndulo duplo com 2GL.

No exemplo apresentado na figura 5.3, para um manipulador com 1GL, a matriz Jacobiana não será completa (2×1), enquanto que para um manipulador 2 GL (figura 5.4) a matriz Jacobiana é quadrada (2×2). Para um manipulador com “n” graus de liberdade, a matriz Jacobiana obtida terá ($m \times n$)

Para um robô industrial, as coordenadas de seu elemento terminal serão descritas através de um vetor posição \underline{X} (x, y, z) e sua orientação definida a partir de três ângulos (ψ, θ, ϕ). Isto representará um conjunto de seis graus de liberdade que deverão ser controlados a partir das “n” variáveis articulares do robô.

5.4 – Inversão da Matriz Jacobiana

No método recursivo, figura 5.2, existe a necessidade da inversão da matriz Jacobiana para se poder obter o conjunto de ângulos (referências angulares) para um determinado trabalho, definido no espaço cartesiano. A matriz Jacobiana inversa pode ser obtida através de dois métodos:

- Inversão simbólica: se deixarmos a matriz Jacobiana em sua forma original, forma simbólica, pode-se encontrar a inversa, usando a álgebra. Mas a complexidade do Jacobiano torna este procedimento muito difícil ou até mesmo impossível.
- Inversão numérica: para cada instante, a configuração do robô define um conjunto de variáveis de juntas, desse modo, então, a matriz Jacobiana, em cada instante, é uma matriz numérica. A literatura em análise numérica apresenta diversas técnicas para a inversão de matrizes. Tais técnicas não devem apenas serem rápidas, mas, também, retornarem respostas precisas.

Na utilização de métodos recursivos, para a implementação de um programa de geração de trajetórias, existe a necessidade da utilização de um método numérico para a inversão da matriz Jacobiana

Existem diversos métodos para a inversão da matriz Jacobiana. Entre eles podemos citar os métodos de Gauss (Kreyszig, 1983), utilizado para o cálculo da matriz inversa de J , e Greville (Gorla, 1984), utilizado para o cálculo da matriz pseudoinversa de J . Estes métodos foram estudados e implementados em Sá (1996).

Estes métodos são ditos métodos diretos e, em princípio, produzirão uma solução exata (desprezando os erros de arredondamento), se houver, em um número finito de operações. Entretanto, as soluções podem perder o significado, caso o sistema linear seja mal-condicionado.

Os métodos numéricos interativos convergem para uma solução possível, são de caráter geral e com o atual desenvolvimento dos microcomputadores a utilização destes métodos em tempo real é viável (Romano et al., 2002). Isto pode ser implementado, por exemplo, através do método numérico recursivo que usa o modelo cinemático inverso e a matriz jacobiana, utilizando o método de eliminação de Gauss (Dorf et al., 1972).

O método de Gauss possui o inconveniente de que em muitos casos a solução de um sistema linear existe, mesmo quando a inversa da matriz não existe, e também devido ao fato do alto número de operações aritméticas necessárias. Já o método de Greville, pseudoinversa (Nashed, 1976), não apresenta esses inconvenientes, motivo pelo qual ele será utilizado.

Pode-se notar que o modelo geométrico do robô é de grande importância, pois através dele pode-se calcular a matriz Jacobiana do robô e, além disso, ele é utilizado diretamente na malha de controle para a geração de uma trajetória ponto a ponto em tempo real.

A seguir será apresentado um algoritmo de geração de trajetórias para um robô, utilizando o método recursivo para a inversão do modelo cinemático obtido através de Denavit-Hartenberg. Também será apresentado um algoritmo para a geração de trajetórias através de interpolação e filtragem (de pontos de passagem) uma vez que em muitos casos é necessária a geração de trajetórias a partir de determinados pontos de passagem angulares obtidos diretamente das juntas do robô.

5.4.1 – Cálculo da Matriz Jacobiana

A matriz Jacobiana, utilizada no método recursivo para o cálculo do modelo cinemático inverso, é uma forma multi-dimensional de derivada e relaciona a velocidade no espaço de juntas à velocidade no espaço cartesiano, isto é:

$$\Delta \underline{x} = J \Delta \underline{\theta} \quad (5.10)$$

Uma técnica para determinar o Jacobiano de um robô de seis graus de liberdade é fazendo uso das matrizes de transformações que definem a geometria do robô (Paul, 1981).

A matriz Jacobiana, como pode ser observado, intervém na solução numérica da inversão do modelo geométrico (método recursivo) e, portanto, nas soluções de controle implementado diretamente no espaço operacional, como pode ser observado na figura 5.2. Ela é uma forma multidimensional da derivada e relaciona a velocidade no espaço de juntas à velocidade no espaço cartesiano.

Uma técnica para determinar o Jacobiano de um robô de seis graus de liberdade é fazendo uso das matrizes de transformações que definem a geometria do robô, que será apresentada posteriormente.

5.4.2 – Controle de Posição de um Manipulador a partir da Inversão da Matriz Jacobiana

O controle de um robô no espaço de tarefas necessita de uma transformação de coordenadas. Esta transformação poderá ser realizada a partir da inversão da matriz Jacobiana:

$$\delta \underline{\theta} = J(\underline{\theta})^{-1} \delta \underline{x} \quad (5.11)$$

em que

$J(\underline{\theta})^{-1}$ representa a matriz Jacobiana inversa (se ela existir)

Esta relação indica a variação do incremento $\delta \underline{\theta}$ das variáveis articulares para um dado deslocamento $\delta \underline{x}$ do elemento terminal do robô. A partir da utilização da equação 5.9, de modo iterativo e recalculando $J(\underline{\theta})$ a cada passo de iteração, uma trajetória $\underline{x}(t)$ poderá ser realizada num determinado tempo, em função da variação dos ângulos das juntas, conforme

mostra o diagrama de blocos apresentado na figura 5.5, referente ao controle de posição de um robô com N GL no espaço operacional.

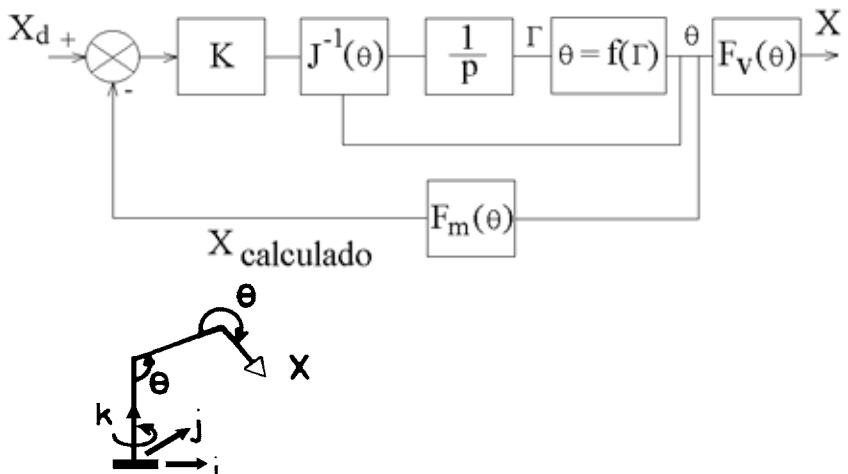


Figura 5.5: Controle de Posição de um robô no seu espaço operacional.

A figura 5.5 mostra a malha de controle de posição de um robô. A partir da comparação da posição atual do robô x (valor calculado a partir das informações de posições dos sensores de juntas e modelo direto) e sua posição de referência x_d , um sinal de erro é amplificado e transformado em termos de coordenadas articulares $\delta\theta$ a partir do cálculo de $J(\theta)^{-1}$. O sinal de erro é integrado e depois utilizado como sinal de entrada para controle das variáveis articulares do robô.

Nesta representação, as relações cinemáticas $F_m(\theta)$ referentes à modelagem da arquitetura de um robô (no caso rígido) são utilizadas na malha de controle. Por outro lado, o bloco $F_v(\theta)$ representa a relação entre os torques necessários para acionamento das juntas e seus valores angulares θ , podendo ser obtidos a partir das variáveis articulares e métodos de correção de parâmetros.

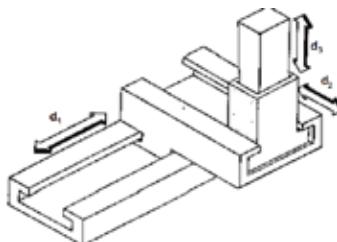
Finalmente, a matriz Jacobiana, utilizada no método recursivo para o cálculo do modelo cinemático inverso, é uma forma multidimensional da derivada e relaciona a velocidade

no espaço de juntas à velocidade no espaço cartesiano. A sua solução deverá ser implementada através da utilização de algoritmos numéricos, que será aproximada por $\Delta \underline{x} = J \cdot \Delta \underline{\theta}$.

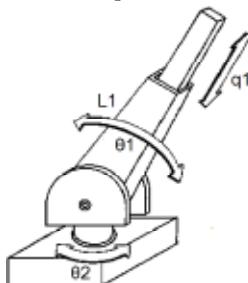
5.5 - Exemplos de Aplicação

Para os dispositivos robóticos apresentados na figura 5.6 pede-se:

- Parâmetros cinemáticos de Denavit-Hartenberg (DH);
- Matrizes de transformação homogênea correspondentes a cada grau de liberdade;
- Modelo cinemático direto;
- Modelo cinemático inverso.



a) Manipulador 3T



b) Manipulador 2R + 1T

Figura 5.6: Dispositivos Robóticos.

5.5.1 - Dispositivo Robótico Cartesiano 3 GL

Os parâmetros cinemáticos de Denavit-Hartenberg

para o dispositivo robótico cartesiano obtidos através da sistemática, apresentada no capítulo 3, são obtidos assim:

- a) Estabelecemos os eixos z dos sistemas de coordenadas de cada elo nos eixos de cada junta da estrutura, ou na direção de deslocamento das juntas prismáticas;
- b) A origem do sistema de coordenadas referencial é colocada no eixo z da junta do primeiro elo;
- c) Estabelecemos a origem do sistema de coordenadas na linha ortogonal aos eixos z dos sistemas de coordenadas de cada par de elos para toda a estrutura;
- d) Definimos o eixo x de cada sistema na linha ortogonal aos eixos z de cada par de elos através da regra da mão direita, do eixo z de maior ordem para o de menor;
- e) Os eixos y são definidos pela regra da mão direita; e
- f) Definimos o sistema de coordenadas da garra.

5.5.1.1 - Parâmetros de Denavit-Hartenberg (DH)

Junta i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	d_1	0
2	90°	0	d_2	0
3	90°	0	d_3	0

5.5.1.2 – Matrizes de Transformação Homogêneas (MTH)

As matrizes de transformação homogênea são obtidas através do modelo de transformação com juntas de translação.

$${}^0 A_1 = \begin{bmatrix} 1 & 0 & 0 & d_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^1 A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^2 A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

em que

${}^0 A_1$: translação em X,

${}^0 A_2$: translação em Y, e

${}^0 A_3$: deslocamento em Z.

5.5.1.3 – Modelo Cinemático Direto

O modelo cinemático direto é obtido através da multiplicação das matrizes de transformação homogênea, ou seja:

$${}^0 T_3 = {}^0 A_1 \cdot {}^1 A_2 \cdot {}^2 A_3 = \begin{bmatrix} 1 & 0 & 0 & d_1 \\ 0 & 0 & 1 & d_2 \\ 0 & 1 & 0 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

A partir desta matriz, podemos resumir o modelo cinemático pelas equações:

$$x = x_o + d_1$$

$$y = y_o + d_2$$

$$z = z_o + d_3$$

Se considerarmos a posição inicial igual a [0, 0, 0], teremos para o robô cartesiano em estudo:

$$x = d_1$$

$$y = d_2$$

$$z = d_3$$

5.5.1.4 – Modelo Cinemático Inverso

a) Matriz Jacobiana

O modelo cinemático inverso pode ser obtido através da inversão da matriz Jacobiana, dada por:

$$J(d_i) = \begin{bmatrix} \frac{\partial F_1}{\partial d_1} & \frac{\partial F_1}{\partial d_2} & \frac{\partial F_1}{\partial d_3} \\ \frac{\partial F_2}{\partial d_1} & \frac{\partial F_2}{\partial d_2} & \frac{\partial F_2}{\partial d_3} \\ \frac{\partial F_3}{\partial d_1} & \frac{\partial F_3}{\partial d_2} & \frac{\partial F_3}{\partial d_3} \end{bmatrix}$$

b) Posição Final

$$\begin{aligned} x &= F_1(d_1, d_2, d_3) = x_o + d_1 \\ y &= F_2(d_1, d_2, d_3) = y_o + d_2 \\ z &= F_3(d_1, d_2, d_3) = z_o + d_3 \end{aligned}$$

c) Cálculo do Jacobiano

$$J = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

d) Modelagem Cinemática Inverso

Procuramos o modelo cinemático inverso, assim procuramos J^{-1} que nos permita definir os deslocamentos em termos $d=f^1(X,Y,Z)$.

Assim temos $\delta d = J^{-1} \delta X$, e o modelo cinemático inverso, portanto, é caracterizado através de :

$$J^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

5.5.2 - Dispositivo Robótico 2R + 1T

Na resolução deste problema, consideraremos somente as duas primeiras rotações, conforme mostra a figura 5.7.

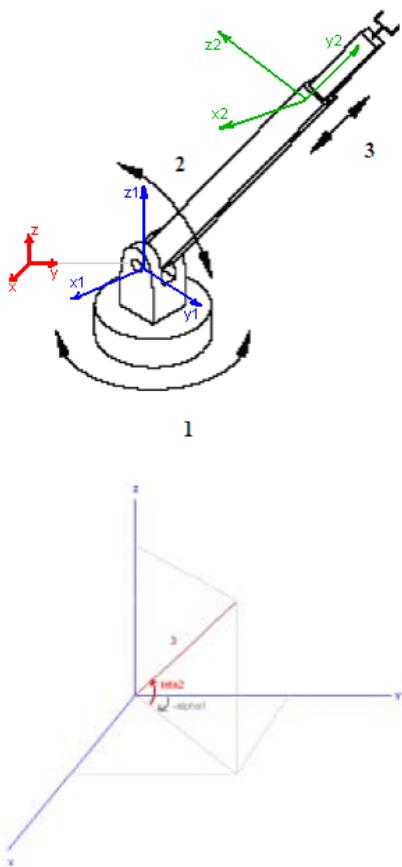


Figura 5.7: Manipulador 2R + 1T.

5.5.2.1 - Parâmetros de Denavit-Hartenberg (DH)

Junta i	a_{i-1}	a_{i-1}	d_i	θ_i
1	α_1	0	0	0
2	0	a_2	0	θ_2

5.5.2.2 – Matrizes de Transformação Homogêneas (MTH)

a) Rotação antiborária em torno de Z:

$${}^0A_1 = \begin{bmatrix} c\alpha_1 & s\alpha_1 & 0 & 0 \\ -s\alpha_1 & c\alpha_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

b) Rotação horária em torno de X_1 :

$${}^1A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\theta_2 & -s\theta_2 & 0 \\ 0 & s\theta_2 & c\theta_2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

5.5.2.3 – Matriz de Transformação Final

$${}^0T_2 = {}^0A_1 \cdot {}^1A_2 = \begin{bmatrix} c\alpha_1 & s\alpha_1 c\theta_2 & -s\alpha_1 s\theta_2 & 0 \\ -\alpha_1 & c\alpha_1 c\theta_2 & -c\alpha_1 s\theta_2 & 0 \\ 0 & s\theta_2 & c\theta_2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$c\alpha_i = \cos(\alpha_i), \quad s\alpha_i = \sin(\alpha_i)$$

$$c\theta_2 = \cos(\theta_2), \quad s\theta_2 = \sin(\theta_2)$$

5.5.2.4 – Implementação em *Matlab*TM

```
clear all  
close all  
clc
```

```
syms cosalpha1 senalpha1 costeta2 senteta2
```

```
T1 = [cosalpha1 senalpha1 0 0  
       -senalpha1 cosalpha1 0 0  
       0 0 1 0  
       0 0 0 1];
```

```
T2 = [1 0 0 0  
       0 costeta2 -senteta2 0  
       0 senteta2 costeta2 0  
       0 0 0 1];
```

```
Tf = T1*T2;
```

5.5.2.5 – Cálculo do Jacobiano

a) Parâmetros utilizados

Parâmetros Geométricos:

$a_1 = 3$	% dimensão do braço
$\alpha_{1i} = 25^\circ$	% posição angular inicial da junta 1(em graus)
$\alpha_{1f} = 12^\circ$	% posição angular final da junta 1(em graus)
$\theta_{2i} = 74^\circ$	% posição angular inicial da junta 2(em graus)
$\theta_{2f} = 10^\circ$	% posição angular final da junta 2(em graus)

b) Implementação em Matlab

```
close all  
clear all  
clc
```

```
alpha1i = 25; % ângulos das juntas  
teta2i = 74; % dados em graus  
comp = 3; % comprimento do braço
```

```
Zi = sin(teta2i*pi/180)*comp % coordenadas cartesianas  
Xi = sin(alpha1i*pi/180)* cos(teta2i*pi/180)*comp % do ponto  
especificado  
Yi = cos(alpha1i*pi/180)* cos(teta2i*pi/180)*comp %
```

```
alpha1f = 12; % ângulos das juntas  
teta2f = 10; % dados em graus
```

```
Zf = sin(teta2f*pi/180)* comp % coordenadas cartesianas  
Xf = sin(alpha1f*pi/180)* cos(teta2f*pi/180)*comp % do ponto  
especificado  
Yf = cos(alpha1f*pi/180)* cos(teta2f*pi/180)*comp %
```

c) Cálculo da Posição

Configuração Inicial:

$$X_i = 0.3495; \quad Y_i = 0.7494; \quad Z_i = 2.8838$$

Configuração Final:

$$X_f = 0.6143; \quad Y_f = 2.8899; \quad Z_f = 0.5209$$

d) Cálculo do Jacobiano

Modelo Cinemático

$$X = \sin(\alpha) * \cos(\theta) * L$$

$$Y = \cos(\alpha) * \cos(\theta) * L$$

$$Z = \sin(\theta) * L$$

A partir de $\underline{\delta\theta} = J^{-1} \underline{\delta X}$

onde $X = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$ e $q = \begin{bmatrix} \alpha \\ \theta \\ L \end{bmatrix}$, obtemos a matriz

Jacobiana:

em que

$$J = \begin{bmatrix} \cos(\alpha_1) * \cos(\theta_2) * L & -\sin(\alpha_1) * \sin(\theta_2) * L & \sin(\alpha_1) * \cos(\theta_2) \\ -\sin(\alpha_1) * \cos(\theta_2) * L & -\cos(\alpha_1) * \sin(\theta_2) * L & \cos(\alpha_1) * \cos(\theta_2) \\ 0 & \cos(\theta_2) * L & \sin(\theta_2) \end{bmatrix}$$

e consequentemente sua inversa:

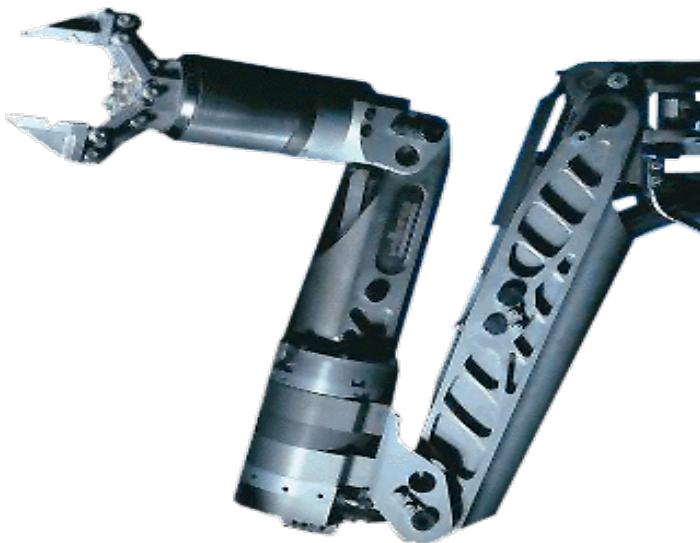
$$J^{-1} = \begin{bmatrix} \cos(\alpha_1) / (\cos(\theta_2) * L) & -\sin(\alpha_1) / (\cos(\theta_2) * L) & 0 \\ -\sin(\alpha_1) * \sin(\theta_2) / L & -\cos(\alpha_1) * \sin(\theta_2) / L & \cos(\theta_2) / L \\ \sin(\alpha_1) * \cos(\theta_2) & \cos(\alpha_1) * \cos(\theta_2) & \sin(\theta_2) \end{bmatrix}$$

5.6 - Referências Bibliográficas

- CLOSE, C. M., FREDERICK, D.K.: “Modeling and Analysis of Dynamic Systems”, Houghton Mifflin Company, 1989.
- CRAIG, J.J. : Introduction to robotics, 2nd ed. New York: Addison Wesley, 1989.
- DORN, W. S., McCRACKEN, D. D.: “Numerical Methods with Fortran IV Cases Studies”, John Wiley & Sons, Inc, 1972.
- KREYSIZIG, E.: “Advanced Engineering Mathematics”, John Wiley & Sons, Inc, 1983.
- PAUL, P.: Robot Manipulators: Mathematics, Programming and Control, The Mit Press, 1981.
- ROMANO, V.F. e outros: Robótica Industrial: Aplicação na Industria de Manufatura e Processos, São Paulo: Edgard Blucher Ltda, 2002 p 20-46
- SÁ, C.E.; ROSARIO, J.M.: “Implementation of Numerical Algorithms for the Resolution of the kinematic Inverse Problem of Robots Manipulators”, ICONE'96 Second International Conference on Non-Linear Dynamics, Chaos, Control and their Applications in Engineering Sciences, São Pedro, 1996.
- SÁ, C. E.: “Implementação de métodos numéricos para a resolução do problema cinemático inverso de manipuladores robóticos com ênfase em controle de posição”, Dissertação de Mestrado, Unicamp, 1996.
- Watt, D. A.. “Ada : Language and Methodology”. Prentice - Hall International (UK) Ltd, 1987.

CAPÍTULO 6

Geração de Trajetórias



CAPÍTULO 6

Geração de Trajetórias

Um manipulador robótico é um dispositivo que tem por função posicionar e orientar um mecanismo existente na sua extremidade. Esse mecanismo tem como objetivo fazer a fixação adequada de ferramentas definidas pelo tipo de tarefa a executar. Assim, duas partes principais podem ser consideradas na estrutura de um manipulador: o braço, constituído no mínimo por três graus de liberdade utilizados para posicionamento do ponto de concentração dos referenciais de orientação, e o punho, normalmente constituído por outros três graus de liberdade rotacionais, com a função de orientação do referencial terminal.

Neste capítulo são apresentados procedimentos para a geração de trajetórias on-line e off-line, utilizando algoritmos numéricos para cálculo do modelo cinemático inverso, para a geração de uma trajetória angular necessária para o movimento das juntas de um manipulador para alcançar um dado objetivo. A apresentação é direcionada a manipuladores robóticos antropomórficos com juntas rotacionais, permitindo que o mesmo possa se deslocar de uma posição e orientação iniciais a uma posição e orientação final desejada. Os procedimentos apresentados poderão ser estendidos para outros tipos de robôs.

6.1 - Introdução

A geração de trajetórias é realizada a partir do modelo geométrico do robô, em que a mesma representa a evolução no tempo da posição, da velocidade e aceleração das juntas do robô, isto é, a geração de incrementos angulares de juntas necessárias para que o robô realize uma determinada tarefa.

As trajetórias podem ser especificadas em coordenadas de juntas ou cartesianas. Em muitas aplicações a programação de tarefas de robôs é realizada no espaço das juntas (*joint space*), não necessitando de um modelo geométrico, e a trajetória angular de mesma natureza dos sinais provenientes do transdutor de posição servirá como sinal de referência para o controlador de cada junta robótica.

A obtenção de referências correspondentes às tarefas definidas no espaço operacional é denominada coordenação de movimentos. A coordenação é expressa matematicamente pela inversão do modelo geométrico. A figura 6.1 apresenta a geração de movimentos de um robô se deslocando de um ponto A para um ponto B (trajetória realizada no espaço das juntas).

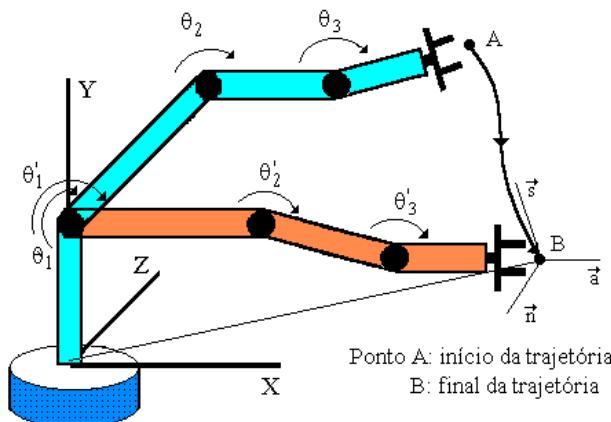


Figura 6.1: Trajetória de movimentação de um manipulador entre duas posições.

Entretanto, na maioria das aplicações, a realização de tarefas está relacionada com o tipo de ferramenta utilizada a partir de um sistema de coordenadas cartesianas fixo à base do robô, designado espaço cartesiano ou espaço operacional (*operational space*). Consequentemente, os movimentos desejados e as leis de controle estão em espaços diferentes.

A geração de trajetórias no espaço cartesiano possui muitas vantagens em relação à programação de tarefas no espaço das juntas, cuja trajetória gerada entre dois pontos pode ser melhor definida, como, por exemplo, quando a ferramenta do robô deve seguir um caminho específico, tal como uma reta ou um arco de círculo. A figura 6.2 apresenta um exemplo de uma trajetória da ferramenta de um robô executando uma tarefa se movimentando em linha reta.

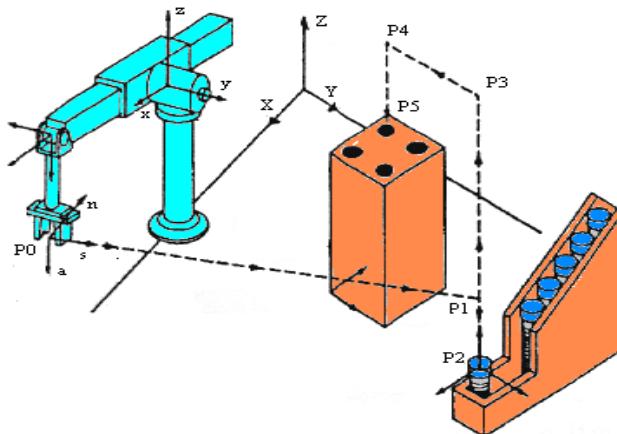


Figura 6.2: Robô executando uma tarefa de movimentação em linha reta.

Para a implementação de um algoritmo de geração de trajetórias no espaço cartesiano, é necessário o conhecimento do modelo geométrico do robô e também de métodos para a inversão do mesmo, para gerar os ângulos das juntas correspondentes a uma determinada configuração espacial. Existem dois métodos para a solução do problema

da inversão do modelo geométrico: os métodos analíticos e os métodos numéricos. Estes métodos serão discutidos no decorrer deste capítulo.

6.2 – Arquitetura de Controle e Geração de Movimentos de um Robô

A programação de tarefas de robôs é realizada a partir da geração de movimentos de mesma natureza dos sinais provenientes do transdutor de posição de referência para cada grau de liberdade do mesmo, e servirá como sinal de referência para o controlador de cada junta robótica, após interpolação.

Na maioria das aplicações, a realização de tarefas está relacionada com o tipo de ferramenta utilizada a partir de um sistema de coordenadas cartesianas fixo à base do robô, espaço cartesiano. Portanto, os movimentos desejados e as leis de controle estão em espaços diferentes, isto pode ser observado na figura 6.3, que apresenta a estrutura completa de controle cinemático de um robô com “n” graus de liberdade.

Nesta figura pode-se observar que a partir da comparação de \underline{x}_d , posição e orientação desejadas, e \underline{x}_{ref} referência, um sinal de erro é gerado e transposto em termos de erros de coordenadas articulares $\Delta\theta_i$, as quais serão utilizadas como sinal de referência para o controle das juntas do robô. A parte envolta por linhas pontilhadas representa o controle de trajetórias em nível de juntas, mostrando deste modo que o sistema suporta a concepção de trajetórias também no espaço das juntas.

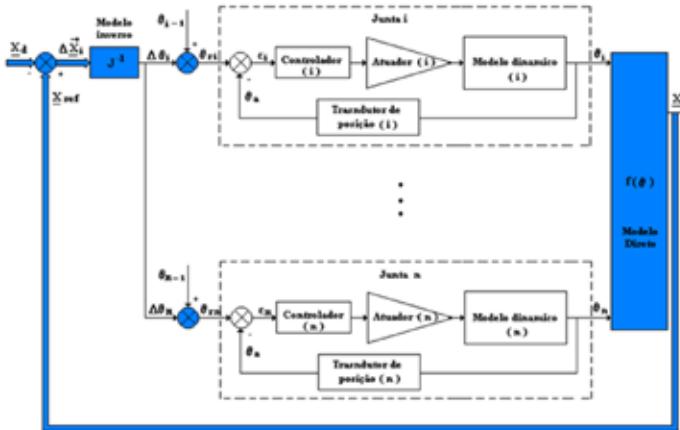


Figura 6.3: Malha de controle de um robô.

6.3 – Controle de Trajetórias

Os métodos de controle de trajetórias utilizados em robôs industriais podem ser divididos segundo duas classificações principais:

Controle Ponto a Ponto (PTP);

Controle por trajetória contínua.

6.3.1 – Controle Ponto-a-Ponto (PTP)

Neste método, o caminho pelo qual o robô precisará passar até um dado ponto final é definido como um conjunto de pontos intermediários. Estes pontos são enviados à memória do sistema de controle pelo usuário como parte do processo de aprendizado do robô. O curso de um ponto intermediário a outro não é pré-determinado e não afeta a implementação da operação principal. Muitos sistemas de controle de robôs industriais presentes no mercado são deste tipo.

O controle Ponto a Ponto é recomendado para robôs planejados para executar tarefas em pontos pré-determinados

(por exemplo, verter misturas em moldes, carregar e descarregar elementos ou pontos de soldagem).

Onde é necessário ultrapassar obstáculos em movimento, o operador deve planejar antecipadamente a introdução de pontos intermediários. Uma modificação mais sofisticada do controle ponto-a-ponto possibilita a introdução de pontos proibidos no controle de programação. O programa irá então ser capaz de assegurar que o robô evitará estes pontos. O robô pode ser ensinado sobre os pontos de seu trajeto de duas maneiras:

- Movimentação manual de cada junta do robô para um ponto desejado, gravando este ponto na memória do robô, e passando para o próximo ponto a ser ensinado - método por aprendizagem (*teach in*).
- Definição das coordenadas de cada ponto desejado, e gravação na memória do robô, sem que este tenha que ser movido fisicamente para que os pontos sejam aprendidos - método de programação *off-line*.

Uma vez aprendidos os pontos do trajeto, programas podem ser escritos direcionando o braço do robô para estes pontos, na ordem desejada, indiferentemente da ordem em que foram ensinados.

O controle ponto-a-ponto é muito mais barato que o controle por procedimento contínuo. No entanto, só é apropriado em operações em que o trajeto entre os pontos definidos não é importante. Para executar caminhos mais complicados, onde é necessário existir precisão do começo ao fim, o controle por trajetória contínua deve ser usado.

6.3.2 – Controle por Trajetória Contínua

Este método de controle é usado nos robôs industriais que executam tarefas em relação a sua ferramenta terminal, sendo importante o controle de trajetórias em nível de ferramenta (por exemplo, pintura spray ou soldagem em arco).

Na maior parte dos robôs industriais disponíveis no mercado se realiza esse tipo de operação através de métodos

numéricos para inversão do modelo cinemático direto, calculando a trajetória em nível de ferramenta em tempo real.

Assim, as coordenadas dos pontos durante o caminho são obtidas a partir da movimentação de forma manual pelo operador, de cada grau de liberdade do robô entre uma determinada posição até a posição final desejada, armazenando pontos importantes durante o percurso em memória.

Durante a realização do movimento, o sistema de controle calcula iterativamente a trajetória seguindo um determinado caminho interpolado (linear, circular ou outros). O número de iterações e a distância entre os pontos dependem da velocidade em que o robô é movido e da taxa em que os pontos são coletados na memória, chamada taxa de amostragem. A taxa de amostragem reflete a quantidade de dados coletados em um período determinado, e é um conceito comum no campo da computação.

Após a etapa de aprendizagem, o robô é retornado ao ponto de partida. É necessário que o robô se movimente através desse caminho desejado quando for chamado para executar a operação.

Como visto acima, este método de controle é muito mais dispendioso que o método ponto-a-ponto. Seu uso é recomendado apenas em tarefas em que o robô deva seguir com extrema precisão todo o percurso desejado.

6.4 – Geração de Trajetórias no Espaço de Juntas

A geração de trajetórias no espaço das juntas, também designado de movimento ponto-a-ponto ou PTP (*point-to-point motion*), consiste na geração de movimentos descritos através da evolução angular das juntas no caso de junta rotacional e linear no caso de junta prismática, obedecendo um perfil de aceleração.

O perfil de aceleração é o modo no qual o corpo se acelera. Obviamente, existem infinitos modos do corpo se acelerar. Entretanto, dois perfis, em particular, estão presentes no sof-

tware e nos interessam diretamente. Para cada um destes perfis existem fórmulas específicas de posição, velocidade e a própria aceleração que foram utilizadas no corpo do programa.

6.4.1 - Aceleração em Degrau

A aceleração em degrau define uma função tipo degrau para a evolução temporal da aceleração, obedecendo três fases distintas (figura 6.4): aceleração constante, velocidade constante e desaceleração constante. As expressões matemáticas correspondentes a cada uma dessas fases é apresentado na tabela 6.1.

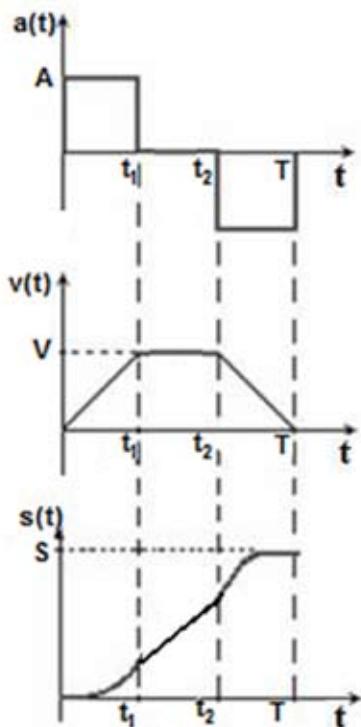
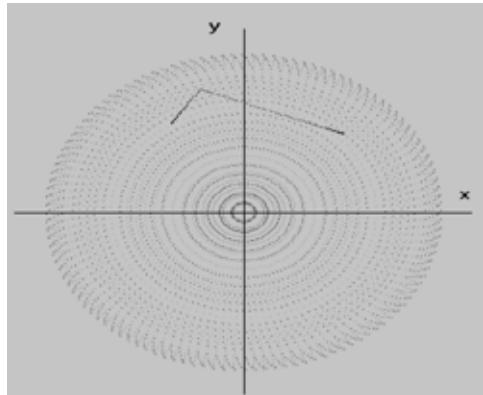


Figura 6.4: Perfil de Aceleração em Degrau.

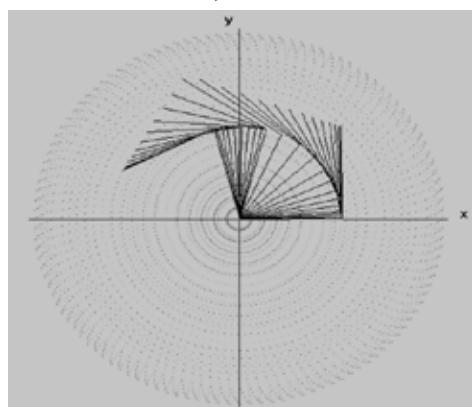
Fase	Tempo	Posição	Velocidade	Aceleração
1 ^a	$0 \leq t < t_1$	$s(t) = \frac{1}{2}At^2$	$v(t) = At$	$a(t) = A$
2 ^a	$t_1 \leq t < t_2$	$s(t) = A(t_1t + \frac{1}{2}t_1^2)$	$v(t) = At_1$	$a(t) = 0$
3 ^a	$t_2 \leq t < T$	$s(t) = A(-\frac{t^2}{2} + Tt + t_1(\frac{1}{2} + t_2))$	$v(t) = A(-t + T)$	$a(t) = -A$

Tabela 6.1: Posição, Velocidade e Aceleração para uma entrada do tipo degrau.

A figura 6.4 apresenta a evolução de uma trajetória no espaço de juntas de um manipulador com 2 GL, apresentando como resultado a evolução da trajetória efetuada pelo ponto final da ferramenta.



a) Final



b) Evolução

Figura 6.5: Trajetória realizada no Espaço de Juntas.

6.4.2 - Aceleração em Polinômio Cúbico

A aceleração em polinômio cúbico define uma função polinomial de terceira ordem para a evolução temporal da aceleração, obedecendo equações matemáticas correspondentes a posição, velocidade e aceleração. A figura 6.6 mostra esse perfil e expressões matemáticas correspondentes a cada uma dessas fases.

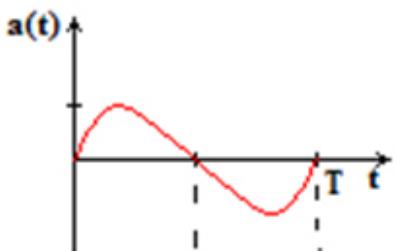


Figura 6.6: Aceleração em Polinômio Cúbico.

Posição	$s(t) = \frac{k}{4} \left(\frac{t^5}{5} - \frac{Tt^4}{2} + \frac{T^2t^3}{3} \right)$
Velocidade	$v(t) = \frac{k}{4} (t^4 - 2Tt^3 + T^2t^2)$
Aceleração	$a(t) = \frac{k}{2} (2t^3 - 3Tt^2 + T^2t)$

$$\text{em que } k = \frac{12 * \sqrt{3}}{T^3} * A$$

Tabela 6.2: Posição, Velocidade e Aceleração para aceleração do tipo polinômio cúbico.

6.5 - Algoritmo Numérico para Geração de Trajetórias

Observando a malha de controle, figura 6.3, para a implementação de um algoritmo de geração de trajetórias, no espaço cartesiano, é necessária a inversão do modelo geométrico, que neste caso pode ser obtido através de Denavit-Hartenberg, pois são necessárias apenas as coordenadas do elemento terminal do robô.

O algoritmo também deverá calcular a matriz Jacobiana do sistema a cada iteração e parar as iterações sempre que o erro máximo permitido para a posição e orientação for alcançado ou quando o número máximo de iterações for alcançado. Este algoritmo permitirá que o elemento terminal do robô siga uma trajetória entre dois pontos bem definida. Este algoritmo permite que a trajetória gerada seja composta por diversos segmentos. Isto permite, por exemplo, que em uma parte da trajetória o robô siga uma linha reta e na outra uma curva qualquer.

O desenvolvimento de um algoritmo numérico (Sá, 1996), para encontrar as posições angulares para um trabalho definido em relação ao seu elemento terminal no espaço cartesiano, contém a solução do problema cinemático inverso através do método numérico recursivo que usa o modelo cinemático e a matriz Jacobiana inversa do manipulador. Para a inversão da matriz Jacobiana podemos utilizar o método de eliminação Gauss (Dorn *et al.*, 1972). Existe quatro critérios para a interrupção das iterações:

- a) Erro máximo permitido:** Este critério utiliza um erro máximo permitido para a posição e para a orientação. O erro de posição (e_{rp}) é obtido através da expressão:

$$e_{rp} = \sum (p_d(i) - p_a(i)) \quad (6.1)$$

em que p_d é a posição final desejada e p_a é a posição atual do elemento terminal do robô.

O erro de orientação (e_p) é obtido utilizando-se o conceito do produto escalar entre dois vetores e é dado por:

$$e_{ro} = \cos^{-1}((\underline{n}_d * \underline{n}_a) / (\|\underline{n}_d\| * \|\underline{n}_a\|)) + \\ \cos^{-1}((\underline{s}_d * \underline{s}_a) / (\|\underline{s}_d\| * \|\underline{s}_a\|)) + \\ \cos^{-1}((\underline{a}_d * \underline{a}_a) / (\|\underline{a}_d\| * \|\underline{a}_a\|)). \quad (6.2)$$

em que os vetores \underline{n} , \underline{s} e \underline{a} são os vetores ortonormais que descrevem a orientação do elemento terminal do robô.

- b) **Número de iterações:** Este critério utiliza um número máximo de iterações, “n”, no caso do sistema não convergir para a posição e orientação desejada.
- c) **Final do limite físico da junta:** Este critério utiliza o máximo percurso para o qual uma junta pode atuar (cada junta possui um limite físico próprio).
- d) **Teste do “Rank” da matriz:** Utilizado apenas para o método de inversão de Gauss. Caso o “rank” da matriz seja menor que o número de linhas da mesma as iterações param, pois o sistema é indeterminado.

Este método apresentará melhores resultados para pequenos deslocamentos a partir do momento que utilizamos uma variável, m , para a divisão do caminho total desejado em pequenos segmentos de trajetos. Neste capítulo serão apresentados resultados de simulação de trajetórias utilizando esses critérios, e estudo da influência do número de divisões do caminho desejado, m .

6.6– Implementação de Algoritmo

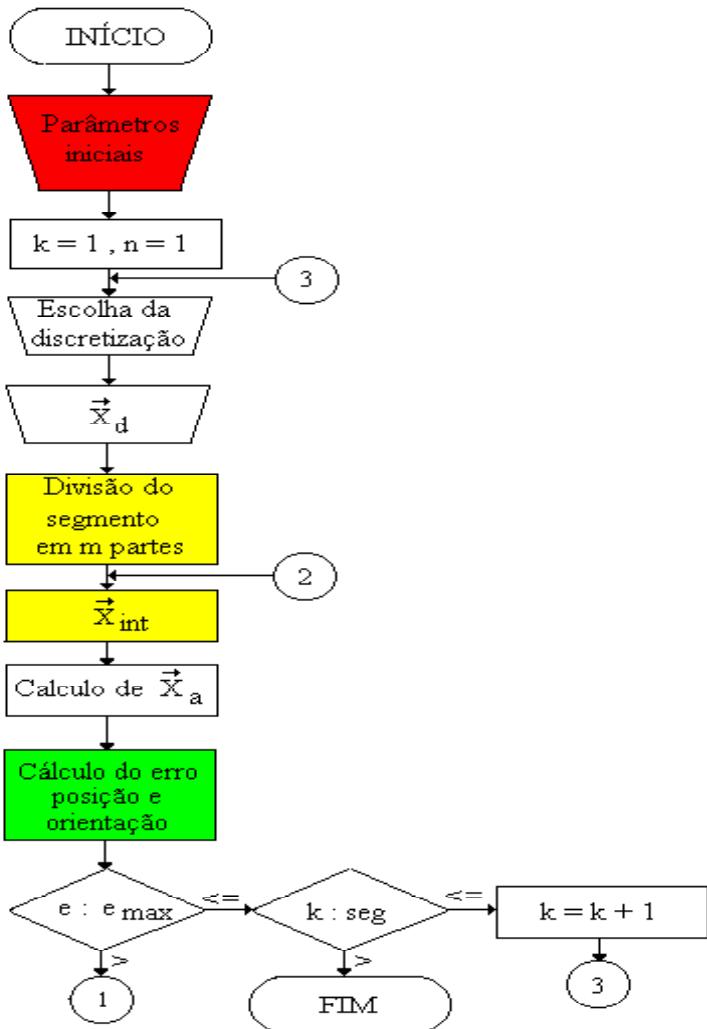
No algoritmo proposto, em cada segmento da trajetória existe a necessidade da escolha da discretização, isto permite que o elemento terminal do robô siga um caminho pré-determinado. A figura 6.7 apresenta sob a forma de diagrama de blocos para a geração de trajetórias.

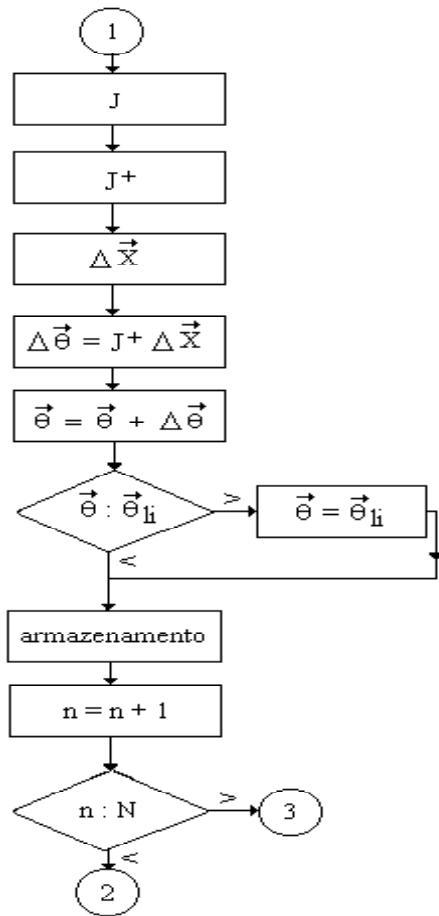


Figura 6.7: Algoritmo simplificado para a geração de trajetórias.

- **Parâmetros iniciais:** Neste bloco é realizado os parâmetros de discretização, sendo escolhido o número de segmentos da tarefa a ser realizada (seg), o número de divisões de cada segmento (m) e determinados os erros máximos de posição e orientação permitidos.
- **Entrada de dados:** A partir do valor da posição (em mm) e orientação (em termos dos ângulos RPY) finais desejadas para cada segmento da tarefa a ser realizada. A posição e orientação finais de cada segmento tornam-se a inicial para o outro segmento. Nesta fase é permitida a alteração do tipo de discretização a ser aplicada ao segmento posterior.

- **Corpo principal do programa:** É realizada a divisão do segmento em m pontos utilizando a discretização esco-lhida. Cada ponto passa a ser uma posição e orientação finais desejadas. É realizado o cálculo do erro de posição e orientação. Neste ponto o algoritmo pode armazenar o conjunto de ângulos para a posição e orientação atual, caso os erros sejam menores que os desejados, ou iniciar o cálculo dos incrementos das juntas. Neste ponto, é calculado J , $J+$, Δx , $\Delta\theta$ e $\theta..$. Verifica-se que cada ângulo de junta está dentro do limite físico permitido para a mesma. Se não estiver, atribui-se a esta junta o valor máximo permitido e, então, armazena-se o conjunto de ângulos obtidos. Este procedimento é realizado até que o valor de m seja alcançado e, após isso, inicia-se o pro-cesso para o próximo segmento, até que o valor dado para *seg* seja atingido.
- **Dados finais:** é obtido o conjunto de ângulos (trajetó-ria gerada) em arquivo ASCII. A figura 6.8 apresenta um algoritmo simplificado para a geração de trajetórias. Neste capítulo serão apresentados resultados de simu-lação e estudo da influência do número de divisões do caminho desejado, m .





Número de segmentos da tarefa (seg)

Número de divisões de cada seg

Erros de orientação e posição máximos

A divisão do segmento em m partes gera os valores para \vec{X}_{int}

O erro de posição e de orientação é obtido a partir de \vec{X}_{int} e \vec{X}_a

onde

N = número máximo de iterações

\vec{X}_d = posição e orientação desejadas

\vec{X}_{int} = posição e orientação intermediária

\vec{X}_a = posição e orientação atuais

e_{max} = erro máximo de orientação e de posição

$\vec{\theta}_{li}$ = ângulo máximo para cada junta

Figura 6.8: Algoritmo para a geração de uma trajetória angular para um robô.

6.7 - Discretização do caminho

Com o objetivo de fazer com que a ferramenta de um robô siga um caminho pré-determinado e bem definido entre a posição e orientação iniciais e a posição e orientação finais desejadas no espaço (por exemplo um círculo ou outro caminho qualquer), o caminho é discretizado em m partes, utilizando equações que gerem o caminho desejado.

A figura 6.9 (a) mostra a discretização entre o ponto inicial da trajetória \vec{x} e o ponto final \vec{x}_d para que o elemento terminal do robô siga uma reta. A figura 6.9 (b) mostra a discretização entre o ponto inicial da trajetória \vec{x} e o ponto final \vec{x}_d para que o elemento terminal do robô siga um semicírculo.

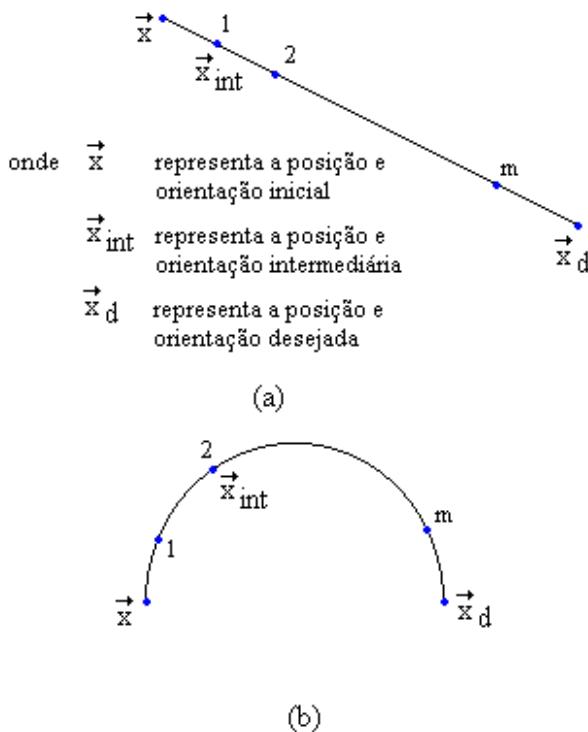


Figura 6.9: Discretização do caminho em m partes.

Deste modo, então, o algoritmo de geração de trajetórias calcula a trajetória da posição inicial até a posição intermediária e, após ela ser atingida, inicia-se o cálculo da trajetória desta posição até a posição intermediária posterior. Isto é realizado até que a posição final desejada seja alcançada. Embora estamos apresentando apenas algoritmos para movimento linear e em semicírculo, outros tipos podem ser implementados utilizando a mesma metodologia apresentada.

A orientação¹ do robô é discretizada de forma linear mesmo quando utiliza-se a discretização em semicírculo, consequentemente, ao discretizarmos um caminho em forma de um semicírculo, estamos discretizando apenas entre a posição inicial e final.

6.7.1 - Discretização Linear

O caminho desejado é discretizado em m partes, de forma linear, fazendo desta forma com que o elemento terminal do robô siga uma linha reta.

Seja (X_1, Y_1, Z_1) o vetor posição inicial do elemento terminal do robô e (X_2, Y_2, Z_2) o seu vetor posição final. Para a discretizar a posição tem-se as seguintes equações:

$$\begin{aligned}x_i &= x_{i-1} + \text{Inc}_1 \\y_i &= y_{i-1} + \text{Inc}_2 \\z_i &= z_{i-1} + \text{Inc}_3\end{aligned}\tag{6.3}$$

1 Para os cálculos internos do programa é utilizada a matriz $[n, s, a]$, uma vez que o modelo geométrico obtido através de Denavit-Hartenberg fornece esta matriz, mas, para melhor compreensão dos resultados finais obtidos apresentados ao usuário, a matriz $[n, s, a]$ será descrita em termos dos ângulos de Euler ou RPY (ψ, θ, ϕ).

em que

$$i = 1, 2, \dots, m;$$

x_i, y_i e z_i = pontos intermediários do vetor posição;

$$Inc_1 = (X_1 - X_2) / m;$$

$$Inc_2 = (Y_1 - Y_2) / m;$$

$$Inc_3 = (Z_1 - Z_2) / m;$$

e m é o número de partes em que o caminho será dividido.

O vetor (x_i, y_i, z_i) é denominado vetor posição intermediário. Seja $[\vec{n}_1, \vec{s}_1, \vec{a}_1]$ a matriz orientação inicial do elemento terminal do robô e $[\vec{n}_2, \vec{s}_2, \vec{a}_2]$ a sua matriz orientação final. Para discretizar a orientação tem-se:

$$\begin{aligned}\vec{n}_i &= \vec{n}_{i-1} + Inc_4 \\ \vec{s}_i &= \vec{s}_{i-1} + Inc_5 \\ \vec{a}_i &= \vec{a}_{i-1} + Inc_6\end{aligned}\tag{6.4}$$

em que

$$i = 1, 2, \dots, m;$$

\vec{n}_i, \vec{s}_i e \vec{a}_i = pontos intermediários do vetor orientação;

$$Inc_4 = (\vec{n}_1 - \vec{n}_2) / m; Inc_5 = (\vec{s}_1 - \vec{s}_2) / m; Inc_6 = (\vec{a}_1 - \vec{a}_2) / m.$$

A matriz $[\vec{n}_1, \vec{s}_1, \vec{a}_1]$ é denominada matriz de orientação intermediária.

6.7.2 - Discretização em semicírculo

O caminho desejado é discretizado em m partes em forma de um semicírculo. A equação de um círculo é dado por:

$$(x - h)^2 + (y - k)^2 = r^2\tag{6.5}$$

Os tipos de discretizações possíveis em semicírculo são:

- no plano X-Y podendo variar o valor da posição z e com $y = f(x)$.
- no plano X-Z podendo variar o valor da posição y e com $z = f(x)$.
- no plano Y-Z podendo variar o valor da posição x e com $y = f(z)$.

Seja (X_1, Y_1, Z_1) o vetor posição inicial do elemento terminal do robô e (X_2, Y_2, Z_2) o seu vetor posição final. Para a discretização em semicírculo, teremos as seguintes equações entre a posição inicial e a posição final:

$$\begin{aligned}y_i &= (r1^2 - (c_1 - h_1)^2)^{1/2} + k_1 \text{ semicírculo no plano x-y} \\z_i &= (r2^2 - (c_2 - h_2)^2)^{1/2} + k_2 \text{ semicírculo no plano x-z} \\z_i &= (r3^2 - (c_3 - h_3)^2)^{1/2} + k_3 \text{ semicírculo no plano y-z}\end{aligned}\quad (6.6)$$

em que

$$i = 1, 2, \dots, m;$$

$$r_1 = \text{abs}(X_2 - X_1) / 2;$$

$$r_2 = \text{abs}(X_2 - X_1) / 2;$$

$$r_3 = \text{abs}(Y_2 - Y_1) / 2;$$

$$h_1 = X_1; \quad h_2 = X_1; \quad h_1 = Y_1; \quad k_1 = Y_1; \quad k_2 = Z_1; \quad k_3 = Z_1;$$

$$c_1 = x_{i-1} + \text{Inc}_1 \text{ onde } \text{Inc}_1 = (X_1 - X_2) / m;$$

$$c_2 = y_{i-1} + \text{Inc}_2 \text{ onde } \text{Inc}_2 = (Y_1 - Y_2) / m;$$

$$c_3 = z_{i-1} + \text{Inc}_3 \text{ onde } \text{Inc}_3 = (Z_1 - Z_2) / m;$$

e m é o número de partes em que o caminho será dividido.

A figura 6.10 mostra como exemplo o semicírculo no plano x-y.

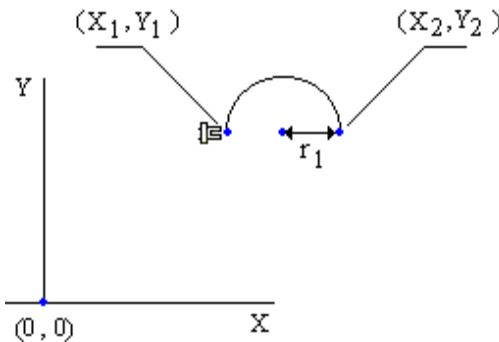


Figura 6.9: Discretização em semicírculo no plano x-y.

Os semicírculos podem apresentar as configurações indicadas na tabela 6.2. Estas configurações são alcançadas adicionando-se nas equações dos semicírculos uma variável que pode ter o valor +1 (direção positiva) ou -1 (direção negativa). As equações tornam-se então

$$\begin{aligned} y_i &= ((r_1^2 - (c_1 - h_1)^2)^{1/2}) \cdot \text{Dir} + k_1 && \text{- semicírculo no plano x-y} \\ z_i &= ((r_2^2 - (c_2 - h_2)^2)^{1/2}) \cdot \text{Dir} + k_2 && \text{- semicírculo no plano x-z} \\ z_i &= ((r_3^2 - (c_3 - h_3)^2)^{1/2}) \cdot \text{Dir} + k_3 && \text{- semicírculo no plano y-z} \end{aligned} \quad (6.7)$$

onde a direção Dir pode assumir o valor ± 1

O vetor (x_i, y_i, z_i) é denominado vetor posição intermediário. A orientação, como anteriormente mencionada, é discretizada de forma linear utilizando, deste modo, as equações 6.3.

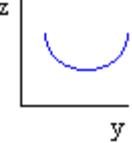
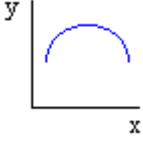
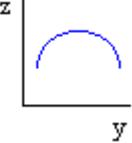
Direção	Plano		
	x-y	x-z	y-z
Positiva (semicírculo abre-se na direção crescente do eixo)			
Negativa (semicírculo abre-se na direção decrescente do eixo)			

Tabela 6.2: Configurações possíveis dos semicírculos.

A figura 6.11 (a) mostra o sentido crescente de cada eixo e a figura 6.11 (b) mostra o sentido decrescente de cada eixo.

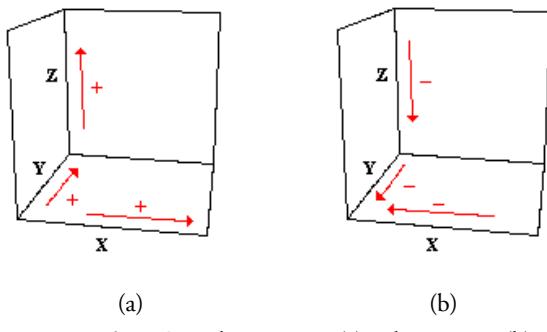


Figura 6.11: Sentido crescente (a) e decrescente (b).

6.8 – Exemplos de Simulação

6.8.1 – Condições utilizadas para Simulação

O algoritmo de geração de trajetórias proposto permitiu a realização de simulações resumidas na tabela 6.3, que contém a posição e orientação finais desejadas para a ferramenta de trabalho do robô. Nestas simulações foram utilizados os seguintes parâmetros:

- Erro máximo de posição permitido: 0.5 mm;
- Erro máximo de orientação permitido: 0.1o;
- Número de divisões para cada segmento: 30 divisões.

A posição inicial do robô é dada por (500.0, 0.0, 1495.0) e a orientação inicial é dada por (0.0, 0.0, 0.0), para todas as simulações, que corresponde a uma posição angular das juntas, em graus, dada por (0.0, 0.0, 0.0, 0.0, 0.0, 0.0). As trajetórias angulares foram obtidas utilizando-se o algoritmo descrito acima.

A tabela 6.3 apresenta a posição angular final das juntas do robô para cada simulação realizada e o número total de pontos para cada trajetória gerada, que são apresentados na forma de gráficos, resultados da simulação referentes a evolução angular das juntas e da trajetória espacial da ferramenta de trabalho do robô.

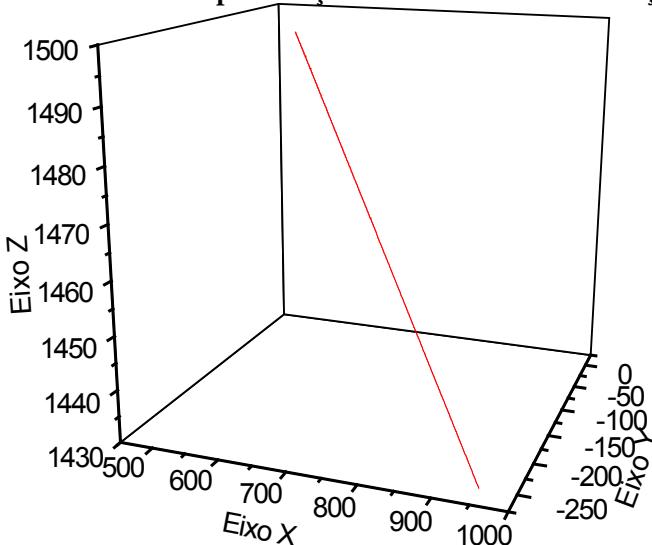
Posição (em mm) e orientação (em graus) finais								
Nº	X	Y	Z	Psi	Teta	Phi	Trajetória ferramenta	Dir.
1	950.0	-250.0	1430.0	0.0	25.0	30.0	linear	
2	850.0	$Y = f(x)$	fixo	10.0	0.0	30.0	semicírculo (x-y)	+
3	850.0	fixo	$z = f(x)$	10.0	0.0	30.0	semicírculo (x-z)	+
4	fixo	-400.0	$z = f(y)$	0.0	25.0	35.0	semicírculo (y-z)	+
5	850.0	$Y = f(x)$	1200.0	10.0	0.0	30.0	semicírculo (x-y)	+
6	850.0	300.0	$z = f(x)$	10.0	0.0	30.0	semicírculo (x-z)	+
7	50.0	-400.0	$z = f(y)$	10.0	0.0	30.0	semicírculo (y-z)	+
8	700.0 500.0	$y = f(x)$ $y = f(x)$	1600.0 1495.0	0.0 0.0	0.0 0.0	0.0 0.0	círculo (x-y)	+
9	600.0	100.0	1495.0	15.0	12.0	20.0	linear	
	600.0	100.0	1700.0	15.0	12.0	20.0		
10	550.0 750.0	0.0 $y=f(x)$	1500.0 1500.0	9.0 9.0	0.0 0.0	3.0 3.0	semicírculo (x-y)	+

Tabela 6.3: Posição e orientação finais desejadas para o elemento terminal para cada simulação.

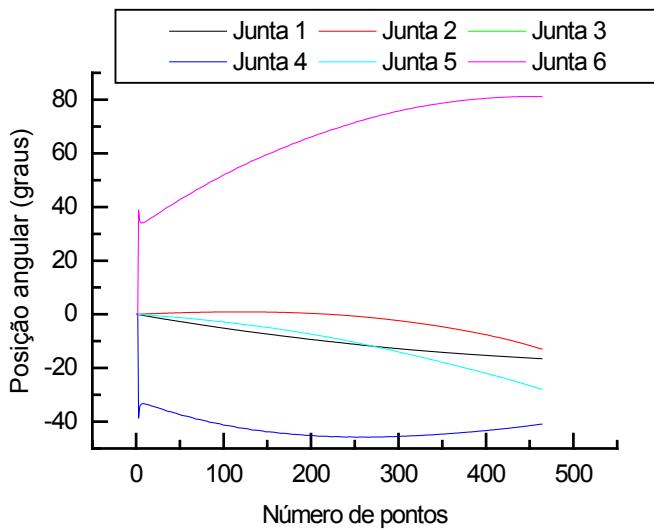
Nº	Posição angular final das juntas (em graus)						Nº ptos
	1	2	3	4	5	6	
1	-16.5	-12.95	52.56	-40.9	-27.92	81.15	464
2	1.03	-10.46	39.08	20.89	-25.21	10.29	559
3	1.02	-10.45	39.08	20.89	-25.21	10.29	540
4	-42.35	-2.41	12.68	99.71	24.73	-24.57	621
5	1.02	22.07	9.11	19.12	-27.62	12.27	577
6	20.52	-15.29	50.04	17.66	-34.35	-5.11	612
7	-83.87	-1.18	-7.71	-12.82	18.47	101.38	747
8	0.0	-16.12	33.63	0.0	-17.51	0.0	329
	0.0	0.0	0.0	0.0	0.0	0.0	328
9	4.56	-1.18	7.78	-60.97	14.96	68.5	130
	4.6	-26.55	37.36	76.34	13.49	84.41	227
10	1.16	-0.89	4.71	-112.01	9.56	113.65	61
	1.19	-5.87	26.06	-155.06	21.77	158.47	320

Tabela 6.4: Posição angular final das juntas do robô e número total de pontos para cada trajetória.

6.8.2 – Apresentação de Resultados da Simulação

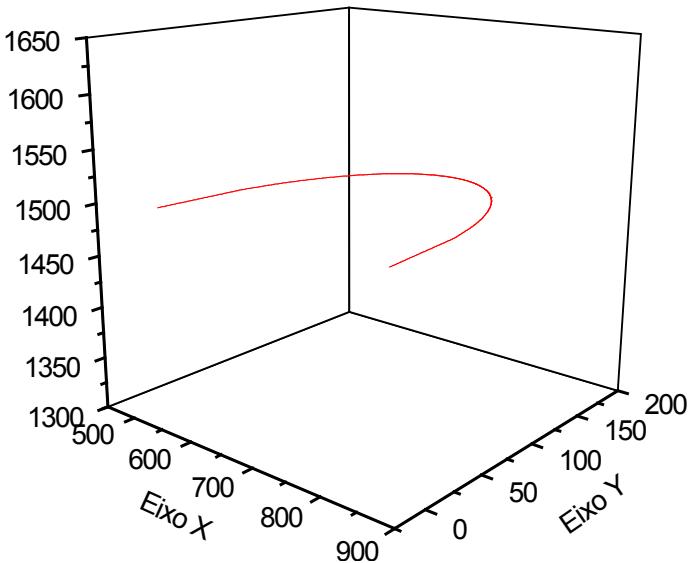


(a) Trajetória espacial seguida pela ferramenta do robô.

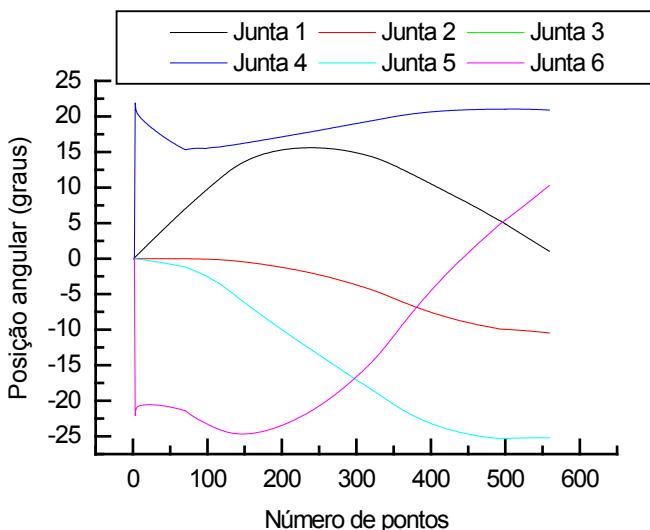


(b) Evolução angular das juntas.

Figura 6.12: Trajetória linear da ferramenta.

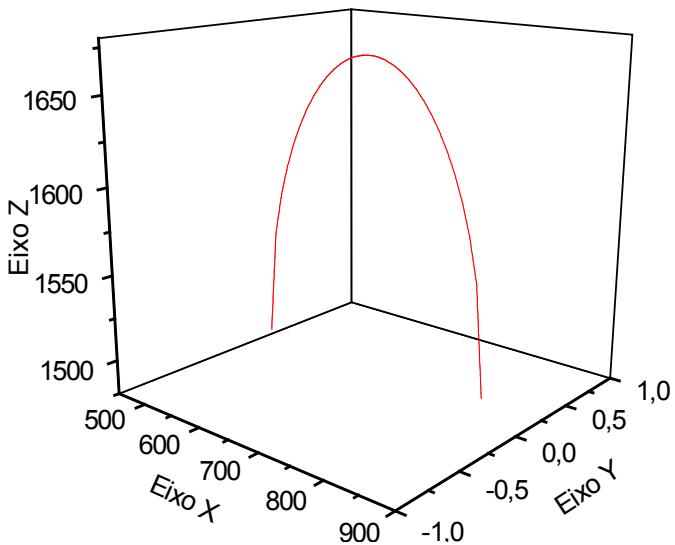


(a) Trajetória espacial seguida pela ferramenta do robô.

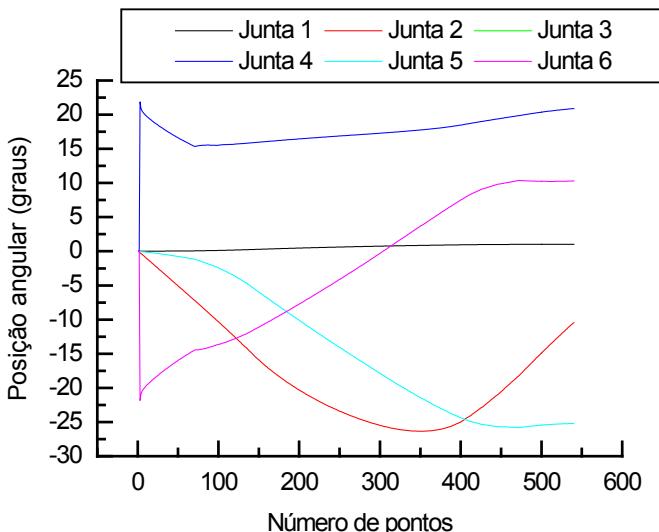


(b) Evolução angular das juntas.

Figura 6.13 : Trajetória da ferramenta realizando um semicírculo (plano x-y) sem variação de z, na direção positiva.

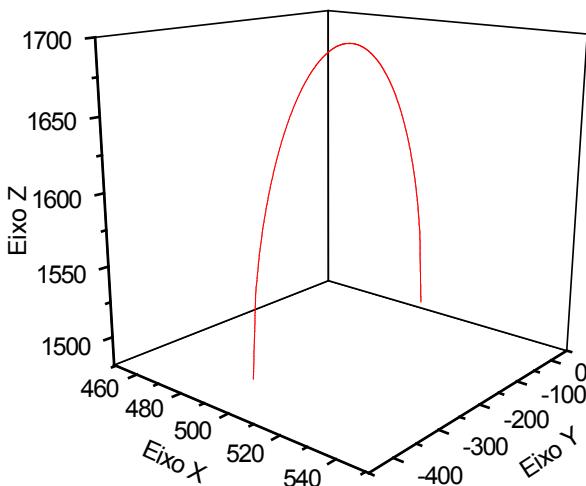


(a) Trajetória da ferramenta do robô realizando um movimento espacial.

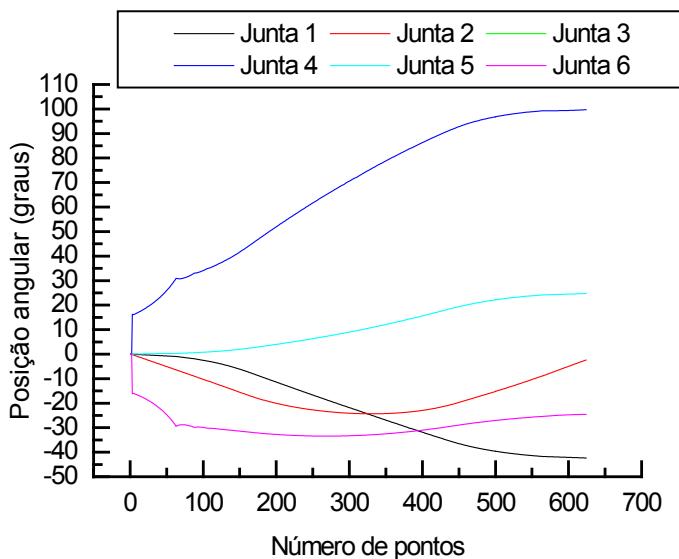


(b) Evolução angular das juntas.

Figura 6.14: Trajetória realizada pela ferramenta do robô em semicírculo (plano x-z), sem variação de y, na direção positiva.

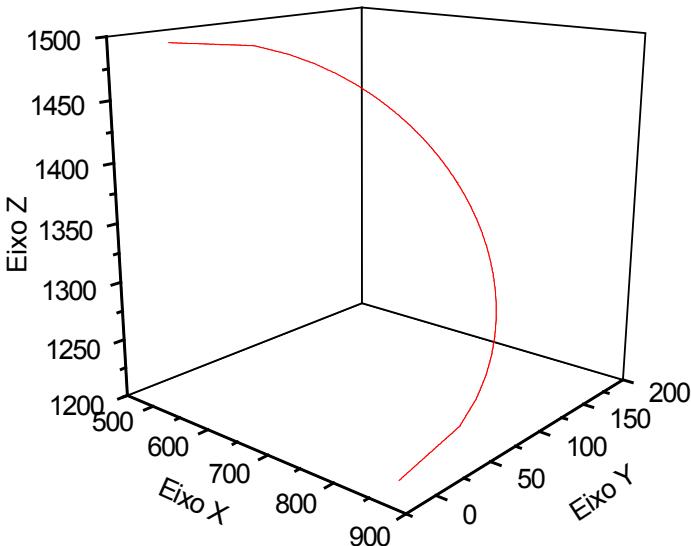


(a) Trajetória espacial seguida pela ferramenta do robô.

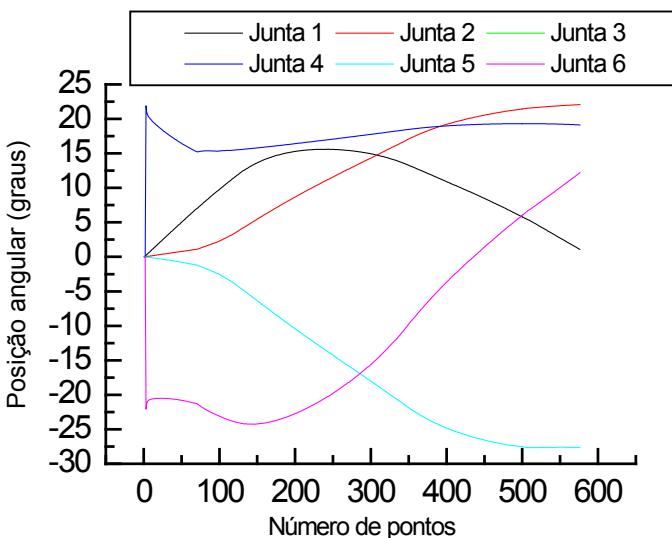


(b) Evolução angular das juntas.

Figura 6.15: Trajetória realizada pela ferramenta do robô em semicírculo (plano y-z), sem variação de x, na direção positiva.

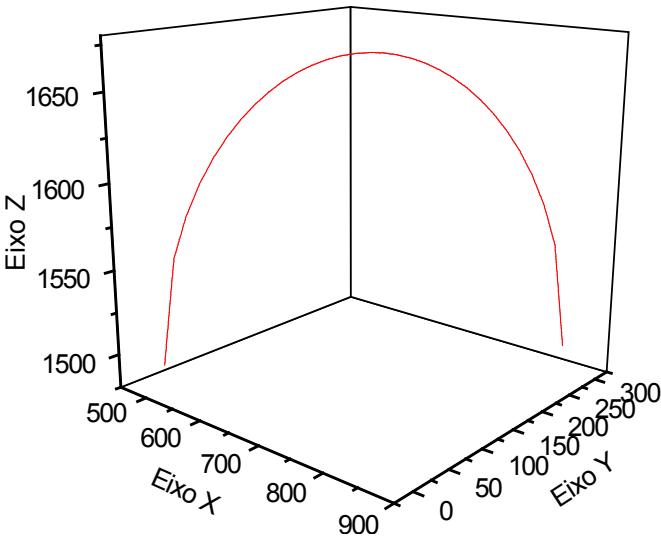


(a) Trajetória espacial seguida pelo ferramenta do robô.

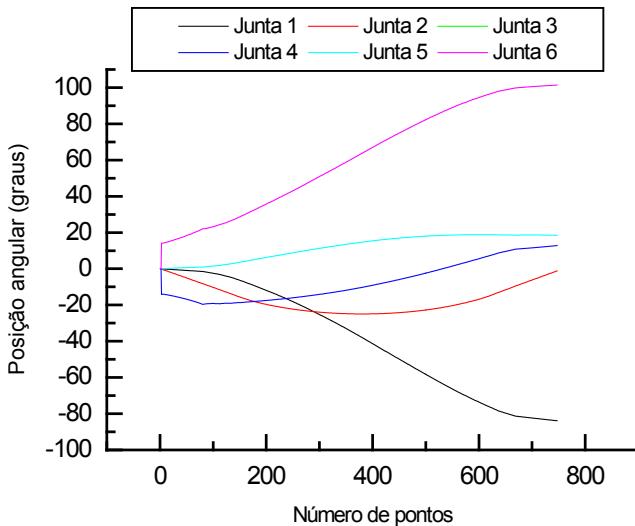


(b) Evolução angular das juntas.

Figura 6.16: Trajetória realizada pela ferramenta do robô em semicírculo (plano x-y), com variação de z, na direção positiva.

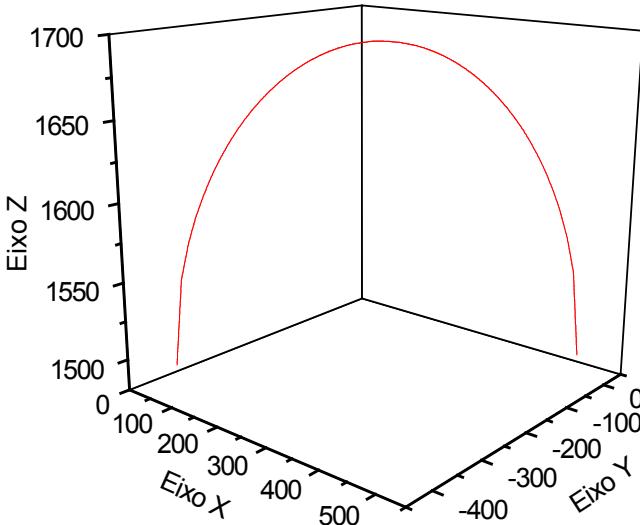


(a) Trajetória espacial seguida pela ferramenta terminal do robô.

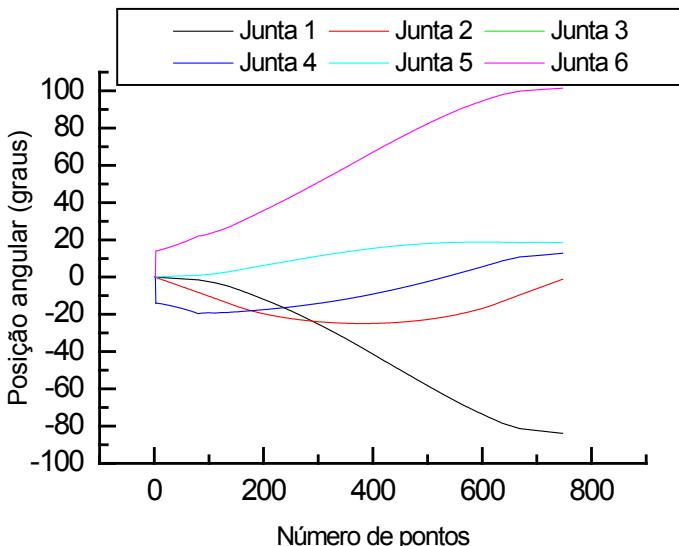


(b) Evolução angular das juntas.

Figura 6.17: Trajetória realizada pela ferramenta do robô em semicírculo (plano x-z), com variação de y, na direção positiva.

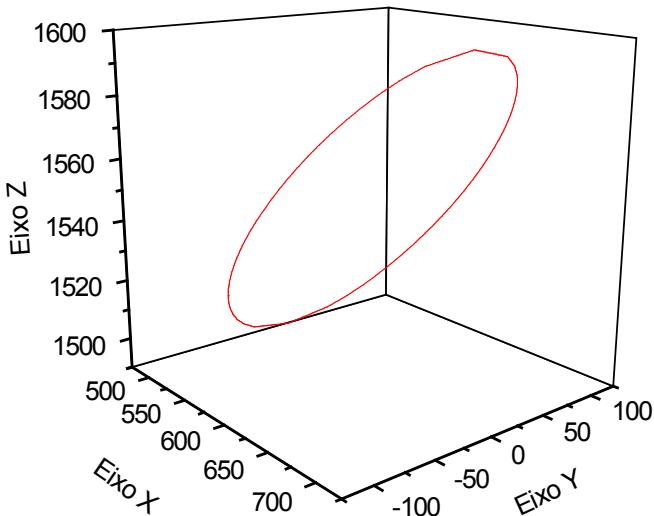


(a) Trajetória espacial seguida pela ferramenta do robô.

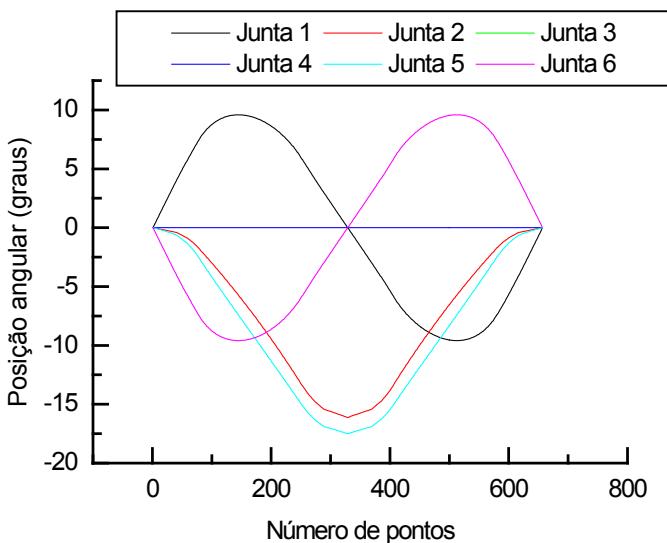


(b) Evolução angular das juntas.

Figura 6.18: Trajetória realizada pela ferramenta do robô em semicírculo (plano y-z), com variação de x, na direção positiva.

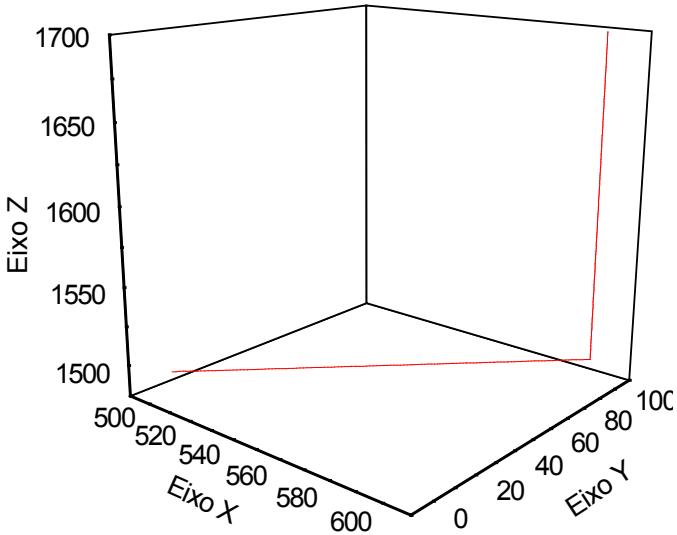


(a) Trajetória espacial seguida pela ferramenta do robô.

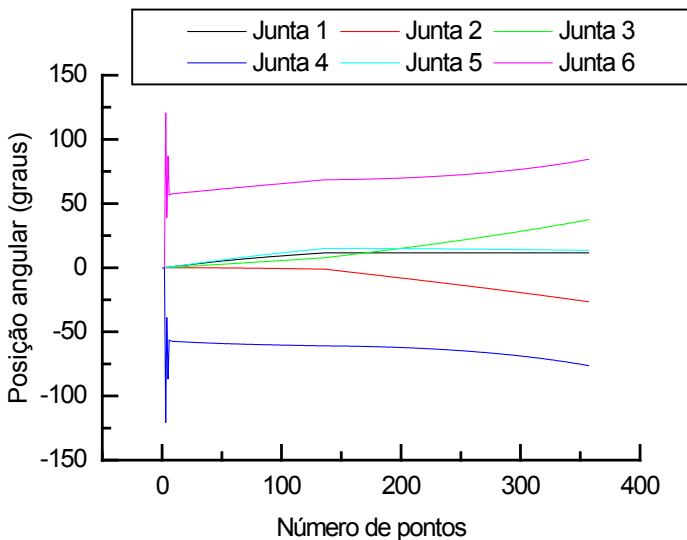


(b) Evolução angular das juntas.

Figura 6.19: Trajetória realizada pela ferramenta do robô em semicírculo (plano x-y), com variação de z (composta de duas partes).

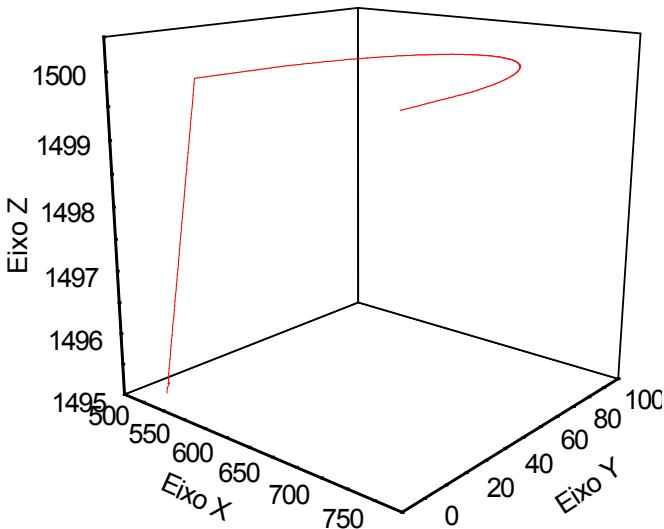


(a) Trajetória espacial seguida pela ferramenta do robô.

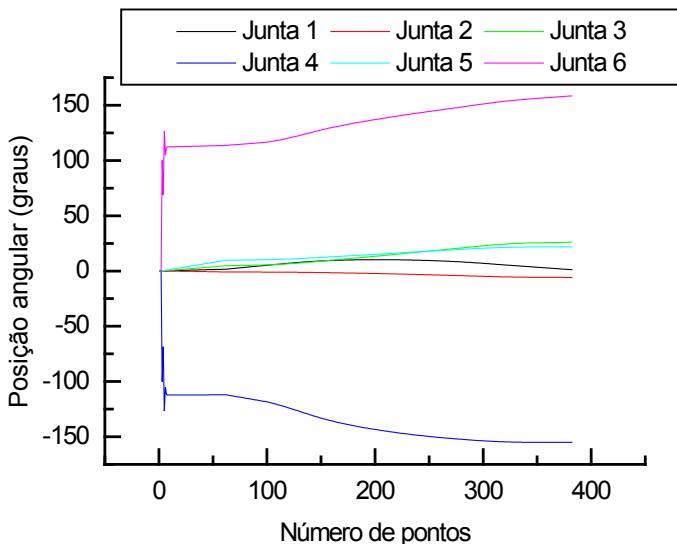


(b) Evolução angular das juntas.

Figura 6.20: Trajetória realizada pela ferramenta do robô (movimento linear composta de duas partes).



(a) Trajetória espacial seguida pelo ferramenta do robô.



(b) Evolução angular das juntas.

Figura 6.21: Trajetória realizada pela ferramenta do robô em duas partes:
movimento linear e um semicírculo no planox-y, sem variação de z.

6.9 - Interpolação e filtragem de pontos de passagem no espaço da juntas

Com o objetivo de gerar uma trajetória a partir de determinados pontos de passagem obtidos pelo operador, no espaço das juntas, torna-se necessário a implementação de algoritmos de interpolação linear.

Ao mesmo tempo torna-se imprescindível a realização de uma filtragem, tendo em vista que a realização somente da interpolação acarretará em acelerações elevadas no início de uma trajetória (Snyder, 1985), o que não é aconselhável para qualquer dispositivo robótico. Assim, para que esta aceleração seja baixa, que resulta em uma velocidade alta da junta, deve-se colocar mais pontos na parte inicial e final da trajetória. Isto é feito através da filtragem da trajetória interpolada, figura 6.22.

Para a realização de uma tarefa, de um modo em geral, o robô executa diversas trajetórias (intervalos) e deste modo a técnica implementada tem de suavizar as extremidades de cada um destes intervalos.

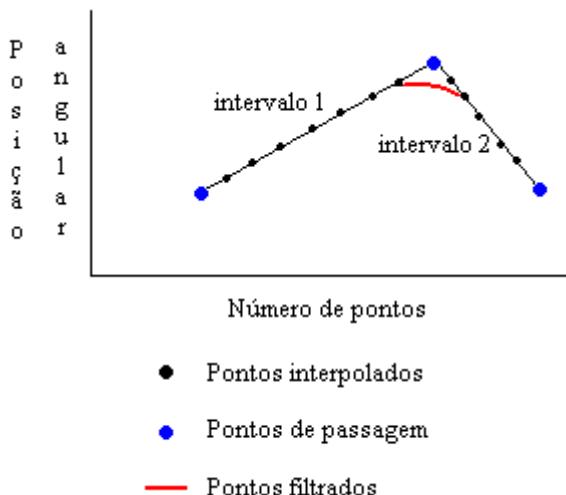


Figura 6.22: Interpolação e filtragem de pontos de passagem.

A configuração angular final obtida, em graus são nos três casos dados por (36.6, 72.56, -99.45, -10.99, -131.38, 60.81).

6.9.1 Interpolação Linear da Trajetória

O principal objetivo da interpolação linear é a criação de uma sequência de pontos de passagem que interligam os pontos da trajetória inicial dada.

6.9.1.1 Algoritmo implementado

O primeiro passo é conhecer os valores (ângulos das juntas) dos pontos de passagem a serem interpolados. Uma vez conhecidas as configurações, pode-se calcular o deslocamento de cada junta para cada intervalo.

$$\Delta_{ij} = \theta_{ij\text{ FINAL}} - \theta_{ij\text{ INICIAL}} \quad (6.8)$$

para $i : 1 \dots$ número de graus de liberdade do robô e $j : 1 \dots$ número total de intervalos.

em que

Δ_{ij} = deslocamento que cada junta terá de realizar;

$\theta_{ij\text{ FINAL}}$ = posição angular final da junta i no intervalo j ;

$\theta_{ij\text{ INICIAL}}$ = posição angular inicial da junta i no intervalo j ;

Após isso se determina qual é o valor do maior deslocamento.

$$\Delta_{\max_j} = \max (\Delta_i) \quad (6.9)$$

O próximo passo é calcular qual será o número total de pontos a serem inseridos em cada intervalo, isto é:

$$n_j = \Delta_{\max_j} / V_{it} \quad (6.10)$$

em que

V_{it} = velocidade máxima utilizada para cada junta varia de 1 a 10, em que 1 representa 10% da velocidade máxima de cada junta, 2 representa 20% e assim por diante.

Os incrementos a serem enviados para cada junta são dados por:

$$\Delta\theta_{ij} = (\theta_{i\text{ FINAL}} - \theta_{i\text{ INICIAL}}) / n_j = \Delta_i / n_j \quad (6.11)$$

em que

$\Delta\theta_{ij}$: incremento angular a ser enviado para a junta i no intervalo j.

O tempo Δt entre cada ponto da trajetória é dado por:

$$\Delta t = \Delta\theta_{ij} / V_{it} = (\Delta_i / n_j) / (\Delta_{\max_j} / n_j) = \Delta_i / \Delta_{\max_j} \quad (6.12)$$

6.9.1.2 - Exemplo

Com o objetivo de mostrar a simplicidade deste método será apresentado a seguir um exemplo, passo-a-passo, no qual serão obtidos os incrementos necessários para a interpolação. A tabela 6.5 mostra os pontos de passagem que sofrerão interpolação.

Ponto	Articulação	
	1 (em graus)	2 (em graus)
1	10.0	10.0
2	20.0	100.0
3	400.0	5.0
4	200.0	500.0

Tabela 10.5: Pontos de passagem a serem interpolados.

- Cálculo dos deslocamentos que cada junta terá de realizar, equação 6.10, e dos deslocamentos angulares máximos, equação 6.11, para cada intervalo (tabela 6.6).

Intervalo	Δ (deslocamento de cada junta)		
	1 (em graus)	2 (em graus)	$\Delta_{max,j}$
1-2	10.0	90.0	90
2-3	380.0	-95.0	380
3-4	-200.0	495.0	495

Tabela 6.6: Cálculos dos deslocamentos angulares máximos.

- Cálculo do número de pontos a serem inseridos em cada intervalo, equação 6.8, e dos incrementos, equação 6.9. O valor de Vit utilizado em ambos os casos foi de 10.0° (tabela 6.7).

Intervalo	n_j	$\Delta\theta$	
		1 (em graus)	2 (em graus)
1-2	9	1.11	10.0
2-3	38	10.0	-2.5
3-4	49	-4.04	10.0

Tabela 6.7: Incrementos das juntas para cada intervalo.

6.9.2 Filtragem da Trajetória Interpolada

Como foi mencionado anteriormente, para que não haja grande variação da velocidade, devido a problemas de dinâmica (inéncias) e mudanças bruscas de direção, deve-se colocar mais pontos nas extremidades de cada intervalo que forma o caminho (angular) a ser seguido pelas juntas. A seguir serão descritos dois tipos de filtragem que podem ser utilizados: triangular e retangular.

6.9.2.1 Filtragem na forma triangular

A figura 6.23 mostra um filtro do tipo janela triangular utilizado para alisar o sinal interpolado. A área da janela triangular é unitária para que a energia seja constante (Lau, 1993).

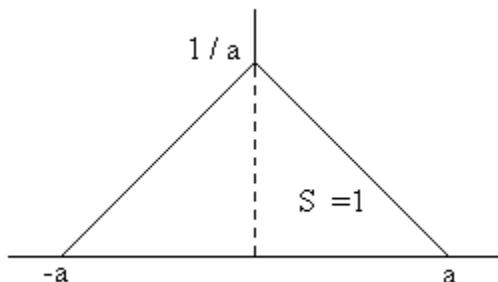


Figura 6.23: Filtro tipo janela triangular.

Utilizando a equação:

$$y = Ax + B \quad (6.13)$$

tem-se:

- para o intervalo entre $-a$ e 0

$$\begin{aligned} \text{para } y = 1/a \text{ tem-se } x = 0 &\Rightarrow B = 1/a \\ y = 0 \text{ tem-se } x = -a &\Rightarrow A = B/a \therefore A = 1/a^2 \end{aligned}$$

deste modo os valores da janela triangular (w), para este intervalo, serão dados por:

$$w(i) = (1/a^2) * (i + a) \quad (6.14)$$

para $i : -a, 1-a, \dots, 0$.

- para o intervalo entre 0 e a

para $y = 1 / a$ tem-se $x = 0 \Rightarrow B = 1 / a$

$$y = 0 \quad \text{tem-se } x = a \Rightarrow A = B / a \quad \therefore A = -1 / a^2$$

deste modo os valores da janela triangular, para este intervalo, serão dados por:

$$w(i) = - (1 / a^2) * (i - a) \quad (6.15)$$

para $i : 0, 1, \dots, a$.

6.9.2.2 Filtragem na forma retangular

A figura 6.24 mostra um filtro do tipo janela retangular.

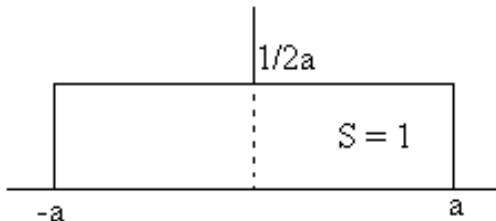


Figura 6.24: Filtro tipo janela retangular.

Os valores da janela retangular (w) para o intervalo de $[-a, a]$ são dados por:

$$w(i) = 1 / 2 * a \quad (6.16)$$

6.9.2.3 Exemplo

Na tabela 6.8 são apresentados os valores obtidos para w, para uma janela triangular, para a igual a 5, utilizando os valores da função w(i) descritos a seguir

Intervalo		[-5 , 0]		[0 , 5]	
w(i) = (1 / 25) * (i + 5)	i	w	i	w	
para o intervalo [-5 , 0]	-5	0.0	0	0.2	
	-4	0.04	1	0.16	
w(i) = -(1 / 25) * (i - 5)	-3	0.08	2	0.12	
	-2	0.12	3	0.08	
para o intervalo [0 , 5]	-1	0.16	4	0.04	
	0	0.2	5	0.0	

Tabela 6.8: Valores do filtro triangular para o intervalo [-5 , 5].

6.9.3 - Equação para a obtenção da trajetória filtrada e interpolada

Após a obtenção do vetor que contém o filtro (triangular ou retangular) e do vetor que contém a trajetória interpolada, é realizada uma convolução entre os dois vetores. Tal convolução para sinais discretos no tempo pode ser definida pela seguinte expressão:

$$Y(n) = \sum x(k) * w(n-k) \quad (6.17)$$

em que

Y = vetor que contém os pontos interpolados e filtrados;

x = vetor que contém os pontos interpolados;

k = número de pontos da trajetória;

w = vetor que contém a janela triangular ou retangular;

n = número de pontos do filtro.

A figura 6.25 apresenta um fluxograma implementado para a interpolação (parte sublinhada a direita) e filtragem em azul de filtragem (parte sublinhada a esquerda).

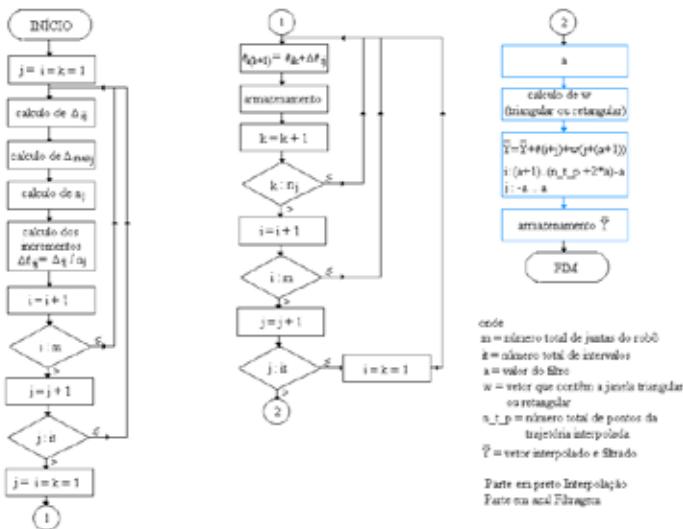


Figura 6.25: Fluxograma para a interpolação e filtragem.

6.9.4 – Exemplo de Aplicação e resultados

Para se observar o efeito da filtragem sobre a trajetória interpolada são apresentadas na tabela 6.9 as posições angulares iniciais para um intervalo correspondente a dois pontos de passagem de uma junta de um robô. Nesta tabela pode-se observar os pontos iniciais da trajetória após a interpolação e após a filtragem, utilizando os filtros triangular e retangular. Observa-se que houve um aumento do número de pontos, o que faz com que a velocidade do robô seja menor, o que é desejado. A figura 6.26 mostra o gráfico dos pontos desta tabela.

Ponto	Trajetória interpolada		
	sem filtragem	filtro triangular	filtro retangular
	Posição angular para a junta 1 (graus)		
1	0.0000	0.0000	0.0000
2	0.0052	0.0002	0.0006
3	0.0103	0.0008	0.0017
4	0.0155	0.0021	0.0034
5	0.0206	0.0041	0.0057
6	0.0258	0.0072	0.0086
7		0.0111	0.0120
8		0.0157	0.0160
9		0.0206	0.0206
10		0.0258	0.0258

Tabela 6.9: Posições angular após a interpolação e após a filtragem utilizando filtro triangular e retangular.

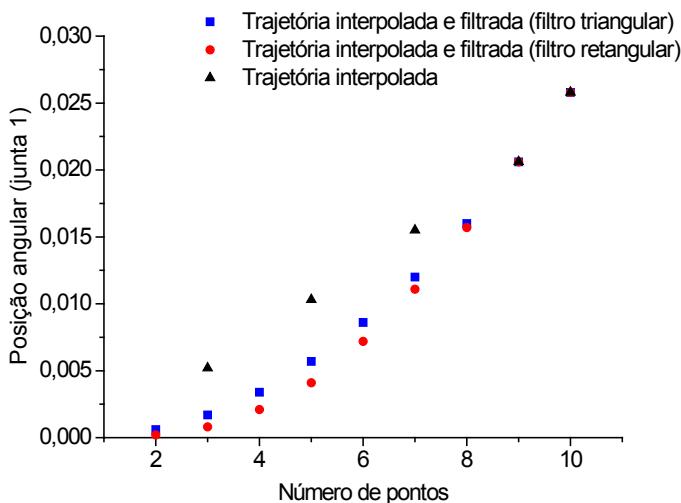


Figura 6.25: Gráfico dos pontos apresentados na tabela 6.8.

O algoritmo de geração de trajetórias apresentou excelentes resultados tanto para a discretização linear como em semicírculo. A partir disto podem-se implementar novos tipos de equações para a discretização, dependendo do caminho espacial que o elemento terminal necessita de seguir.

Pode-se observar através de análise da tabela 6.9 que a posição angular final das juntas para as simulações 2 e 3 são as mesmas, portanto, a posição espacial atingida pelo elemento terminal do robô, em ambas as simulações, são as mesmas (isto não é considerada uma configuração múltipla). A posição espacial final alcançada em ambos os casos é 850,0 e 1495 (em mm). O que difere é o caminho espacial seguido pelo elemento terminal em cada simulação. Os caminhos podem ser observados na figura 6.27, ou nas figuras 6.13(a) e 6.14(a) apresentadas anteriormente.

O algoritmo de interpolação e filtragem também mostrou excelentes resultados, como pode ser observado na simulação realizada. No algoritmo implementado é realizada automaticamente a filtragem após a interpolação, e pela sua simplicidade poderá ser implementado em tempo real, utilizando circuito eletrônico embarcado.

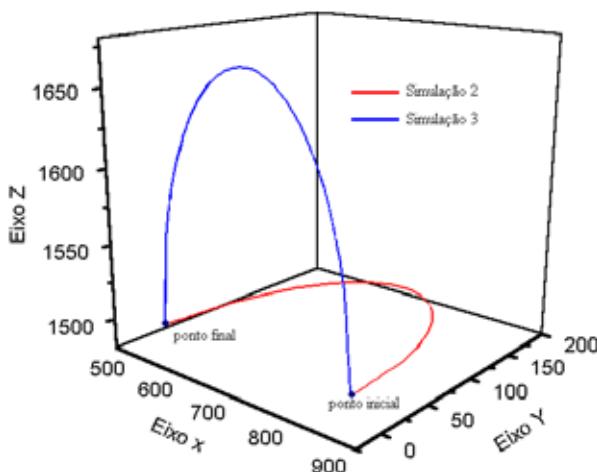


Figura 6.27: Posição espacial final igual obtida através de dois caminhos diferentes.

6.10 - Conclusão

A geração de trajetórias através do uso do modelo cinemático inverso utilizando métodos numéricos apresenta excelentes resultados, permitindo a ainda introduzir novos módulos complementares, tais como módulos de detecção de colisões entre outros, que poderão ser implementados em tempo real, os quais terão como principal característica a fácil utilização e/ou modificação pelo usuário.

Em muitas aplicações existe a necessidade de que a orientação do elemento terminal permaneça constante, enquanto a sua posição varia. Isto pode ser útil para o robô, por exemplo, para realizar uma operação de soldagem ou corte em uma chapa, consequentemente no algoritmo de geração de trajetórias proposto neste capítulo, permite que a orientação do elemento terminal permaneça constante enquanto se varia a sua posição.

O procedimento apresentado neste capítulo permite ainda o estudo da acurácia, repetibilidade e estabilidade do manipulador, ao descrever a trajetória automaticamente, e implementação das eventuais modificações (nos sensores do manipulador e/ou no software desenvolvido).

6.11 - Referências Bibliográficas

Dorn, W. S.: Mccracken, D. D.: Numerical Methods with Fortran IV Cases Studies, John Wiley & Sons, Inc, 1972.

Kraft Telerobotics: Underwater Manipulator System, 1985.

Kreyszig, E.: Advanced Engineering Mathematics, John Wiley & Sons, Inc, 1983.

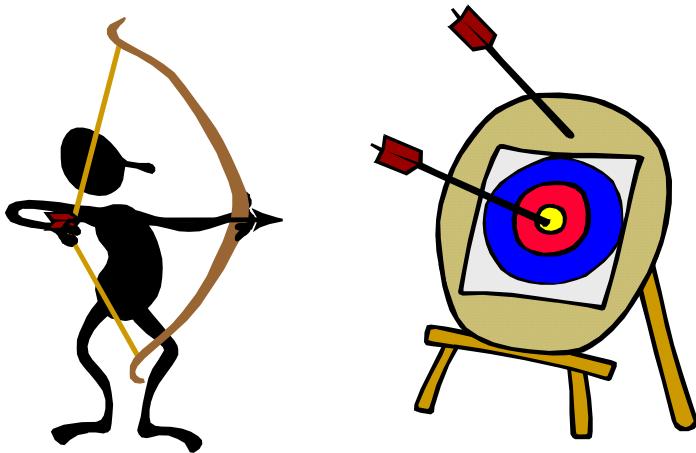
Paul, P.: Robot Manipulators: Mathematics, Programming and Control, The MIT Press, 1981.

Sá, C. E. A.: Implementação de Métodos Numéricos para a Resolução do Problema Cinemático Inverso de Manipuladores Robóticos com ênfase em Controle de Posição, Dissertação de Mestrado, Unicamp, 1996.

Watt, D. A.: Ada: Language and Methodology, Prentice-Hall International (UK) Ltd, 1987.

CAPÍTULO 7

*Análise de Desempenho,
Capacidade e Precisão*



CAPÍTULO 7

Análise de Desempenho, Capacidade e Precisão

Geralmente as principais funções de um robô industrial são a movimentação de materiais, peças, ferramentas ou dispositivos. Para implementação de um robô industrial numa célula de manufatura é imprescindível a verificação da sua capacidade de realizar as funções para as quais foi construído. Desta forma, é necessário a avaliação de seu desempenho em função de normas e critérios, garantindo, assim, a eficiência de sua utilização para a realização de suas tarefas.

Neste capítulo serão apresentadas normas e critérios utilizados na indústria para quantificar a eficiência com que um robô executa uma tarefa. É importante destacar a importância e a utilização da avaliação de desempenho, e por isso descreve-se as características para esta avaliação estabelecidas em normas internacionais, as condições de teste normalizadas e ainda uma orientação para a seleção das características de desempenho a serem testadas para algumas aplicações dos robôs. Adicionalmente, são descritos alguns procedimentos para a realização de testes comparativos entre diferentes robôs, indicando os métodos de medição recomendados pelas normas para a avaliação de desempenho.

7.1 - Introdução

Uma medida de eficiência com que um robô realiza os movimentos inerentes a uma tarefa é a precisão com que os movimentos são realizados. Como os robôs industriais desenvolvem movimentos repetitivos, outra medida importante é sua repetibilidade, ou seja, sua capacidade de repetir o mesmo movimento. Além disso, o comportamento do robô durante seus movimentos, isto é, seu desempenho dinâmico, também é uma informação importante para avaliar sua eficácia na execução de uma tarefa. Assim, pode-se dizer que a avaliação de desempenho de um robô industrial é a verificação da sua precisão, da sua repetibilidade e de seu desempenho dinâmico.

A precisão, a repetibilidade e o desempenho dinâmico determinam se um robô é adequado à execução de uma tarefa. Assim, através da avaliação de desempenho pode-se verificar a adequação de um robô para a realização de uma tarefa industrial. Além disso, comparando desempenhos pode-se selecionar o robô mais apropriado para uma tarefa.

A obtenção de referências correspondentes às tarefas definidas no espaço operacional é denominada coordenação de movimentos, que é expressa matematicamente pela inversão do modelo geométrico que pode ser realizada matematicamente ou numéricamente.

7.2 - Critérios Utilizados na Seleção de Robôs

A avaliação de desempenho pode auxiliar ainda na escolha entre robôs de diferentes fabricantes, sempre que é possível comparar as avaliações realizadas pelos fabricantes. Ao mesmo tempo é um importante critério de sua aceitação em um ambiente industrial, escolhido a partir das exigências da tarefa e das especificações do fabricante, é fundamental que no recebimento do robô todas essas especificações sejam avaliadas.

Além disso, durante o funcionamento do robô podem ocorrer desgastes que alteram suas características. Por isso ava-

liações de desempenho ao longo do tempo podem assegurar que o funcionamento do robô continue adequado para a realização de uma determinada tarefa. E mais, dificuldades verificadas na avaliação de desempenho podem facilitar a manutenção. A Tabela 7.1 apresenta os principais critérios utilizados na seleção de um robô.

Critério	Características
Tipo de Controle	Controle Manual, Sequência Variável, Sequência Fixa, de Repetição, Controle Numérico e Inteligente.
Sistemas de Coordenadas	Juntas, Global e da Ferramenta
Tipos de Movimentos e Métodos de Cálculo	Interpoladores: em nível de Juntas, Linear, Circular
Indicadores de Desempenho	Tempo de Resposta, Estabilidade
Indicadores de Precisão	Resolução Espacial, Precisão Espacial e Repetibilidade
Indicadores de Capacidade	Carga Máxima, Complacência

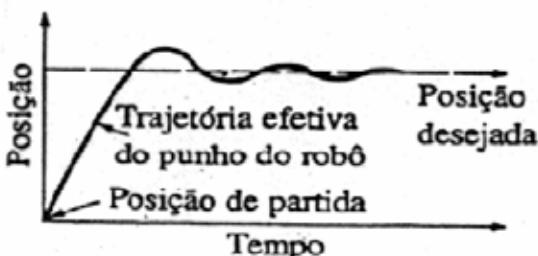
Tabela 7.1: Critérios de Seleção de um Robô.

Algumas considerações:

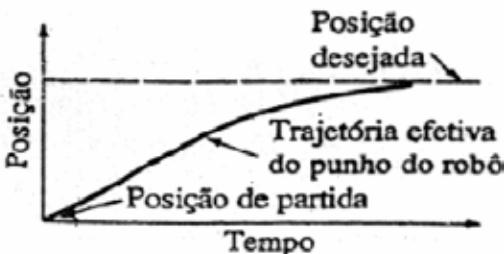
- O tempo de resposta é a capacidade de um robô responder a um comando de movimentação, medido a partir do recebimento do comando até o final do movimento do robô. Ele depende fundamentalmente da velocidade do robô e do sistema de controle.
- Estabilidade é a medida das oscilações que ocorrem no braço de um robô durante o movimento de uma posição a outra. Podemos considerar um sistema com boa estabilidade aquele que durante a sua movimentação

não apresentar pouca ou nenhuma oscilação, enquanto que ele é ruim quando apresentar uma grande quantidade de oscilações.

- Na seleção de um robô podemos ter objetivos conflitantes, como por exemplo o dispositivo apresentar um baixo tempo de resposta e boa estabilidade (figura 7.1), mas utilizáveis segundo a aplicação requerida.



a) Tempo de Reposta



b) Estabilidade

Figura 7.1: Exemplo de Desempenho de Robôs.

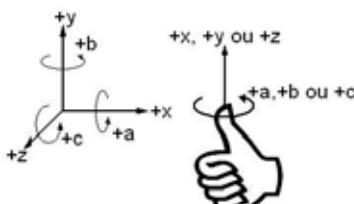
- Normalmente os indicadores de precisão são definidos para a piores situações de um robô, ou seja, para a extremidade ou no caso do braço estiver totalmente estendido.
- No caso do critério de resolução espacial devemos considerar o menor incremento de movimento que um robô possa executar. Este valor é dependente da capacidade de endereçamento da memória do controlador, dada pela sua capacidade

em número de bits endereçáveis, ou seja, é o deslocamento total dividido pelo número de incrementos endereçáveis 2^n .

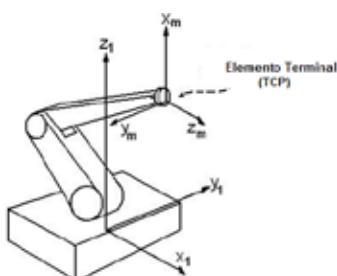
7.3 - Precisão e Repetibilidade

A precisão, a repetibilidade e o desempenho dinâmico para um robô são definidas a partir do elemento terminal, existente na extremidade do robô, onde são fixadas as ferramentas e os dispositivos empregados na execução das tarefas. Ele é designado pelo fabricante como TCP (Tool Center Point).

A posição desta interface geralmente é dada em termos de três coordenadas cartesianas para a sua posição (figura 7.2a) e sua orientação através de três ângulos (figura 7.2b). De acordo com a norma ISO 9787 a representação da orientação da extremidade do robô é feita através dos ângulos a, b e c, respectivamente, em torno dos eixos x, y e z, cujo sentido positivo está indicado na figura 7.2b.



a) Posição



b) Orientação

Figura 7.2: Elemento Terminal de um Robô (Ferramenta).

O conjunto formado pelas três coordenadas cartesianas e pelos três ângulos é definido como postura. A figura 7.2b mostra os sistemas de coordenadas fixados na base do robô (x_1 , y_1 , z_1) e na interface mecânica (x_m , y_m , z_m) de acordo com a norma ISO 9787. Neste caso a postura da interface mecânica é formada pelas três coordenadas cartesianas da origem do sistema (x_m , y_m , z_m) em relação ao sistema da base (x_1 , y_1 , z_1), e pelos três ângulos em torno dos eixos X_1 , Y_1 e Z_1 que determinam a orientação do sistema (x_m , y_m , z_m).

A avaliação de desempenho é uma medida da eficácia com que o robô realiza tarefas com as ferramentas fixadas à sua interface mecânica. Por isso, essa avaliação é feita em relação a um ponto de medição, colocado a uma distância da interface mecânica para levar em conta a dimensão da ferramenta. Nesse ponto de medição é fixado um sistema de coordenadas cuja postura é o foco da avaliação de desempenho do robô.

A postura do ponto de medição é o resultado da combinação das posições de suas juntas. Por isso a precisão, a repetibilidade e o desempenho dinâmico em uma dada postura são, respectivamente, combinações da precisão, da repetibilidade e do desempenho dinâmico de cada uma de suas juntas.

A influência de cada junta nessa combinação varia ao longo do espaço de trabalho do robô. Devido a isso, a precisão, a repetibilidade e o desempenho dinâmico de um robô industrial variam dentro do seu espaço de trabalho. As características de desempenho também variam com a velocidade e com carga aplicada na interface mecânica do robô.

Por isso, para avaliar o desempenho de um robô e compará-lo com o desempenho de outro é preciso conhecer as condições de teste utilizadas na avaliação de cada uma das características.

Existem fabricantes que desenvolveram condições de teste próprias. Há outros que adotam testes definidos nas normas americanas, estabelecidas pelo *American National Standard*

Institute (ANSI). Outros empregam a norma internacional estabelecida pela *International Standard Organization* (ISO).

Os resultados podem variar muito de uma norma para outra tendo em vista que as condições de teste não são as mesmas. Para os mesmos dados de postura, por exemplo, as fórmulas empregadas nas normas americanas (ANSI) dão como resultado uma precisão maior do que as da ISO. Já a repetibilidade calculada segundo a ANSI é sempre menor do que a calculada de acordo com a ISO.

As normas americanas (ANSI) são voltadas principalmente à comparação de desempenho entre robôs de diferentes fabricantes. Compreendem dois volumes: a R15.05-1, que define métodos para avaliar o desempenho estático dos robôs industriais, e a R15.05-2, que estabelece métodos para avaliar o desempenho dinâmico.

No Brasil, a Associação Brasileira de Normas Técnicas (ABNT) adota as normas ISO, por isso apresentam-se aqui mais detalhadamente os testes recomendados por essa instituição. A ISO estabelece que a avaliação de desempenho deve ser realizada de acordo com a norma ISO 9283 (1998, *Second edition*): “*Manipulating Industrial Robots – Performance Criteria and Related Methods*”.

Os testes descritos nesta norma internacional permitem a avaliação de desempenho de robôs individuais e a comparação do desempenho entre robôs diferentes. A tabela 7.2 apresenta os principais critérios de análise de desempenho de um robô industrial (precisão, repetibilidade e desempenho dinâmico) segundo a norma ISO 9283:1998.

Critério	Características de Medida
Precisão	<ul style="list-style-type: none"> • Precisão de postura; • Variação multidirecional na precisão de postura; • Precisão à distância; • Precisão de percurso.
Repetibilidade	<ul style="list-style-type: none"> • Postura • Distância • Percurso
Desempenho Dinâmico	<ul style="list-style-type: none"> • Tempo de estabilização; • Sobrepasso; • Desvios de canto; • Velocidades no percurso; • Tempo mínimo de posicionamento.

Tabela 7.2: Análise de Desempenho de um Robô
(Norma ISO 9283:1998).

É importante destacar que:

- Os desvios na precisão e a repetibilidade ao longo do tempo são medidos pelo deslocamento das características de postura;
- As variações na precisão e na repetibilidade entre robôs do mesmo modelo são caracterizadas pela Intercambiabilidade; e
- A norma ISO 9283 estabelece ainda uma característica para avaliar a flexibilidade do robô denominada de Flexibilidade estática.

As características apresentadas acima podem ser usadas no todo ou em parte para avaliar o desempenho de um robô. A norma não especifica quais delas devem ser empregadas para testar um robô em particular. Apresenta, no entanto, uma orientação para a seleção das características a serem testadas para algumas aplicações típicas, reproduzidas mais adiante neste capítulo. No próximo tópico serão detalhadas essas grandezas.

7.4 – Características de Desempenho

A norma ISO 9283 define características para testar a postura, o percurso, o tempo mínimo de posicionamento e a flexibilidade estática dos robôs industriais.

7.4.1 – Postura

As grandezas de postura quantificam os erros entre uma postura comandada (especificada através da programação do robô) e a respectiva postura atingida (alcançada em resposta à postura comandada com o robô funcionando em modo automático).

As principais fontes de erros podem ser causadas pelo algoritmo de controle, pelas transformações de coordenadas, por diferenças dimensionais entre os componentes do robô e o seu modelo utilizado no sistema de controle, por dificuldades mecânicas como folgas, atrito, histerese, e por influências externas como a temperatura.

Nos robôs industriais a postura comandada pode ser especificada diretamente no robô mediante a gravação das coordenadas das juntas, através de uma caixa de comando ou de outra entrada manual de dados, e indiretamente através de um método de programação fora de linha. A forma pela qual a postura comandada é especificada influencia diretamente os resultados dos testes e, de acordo com a norma ISO 9283, deve estar claramente descrita no relatório dos testes.

Na figura 7.3 são apresentadas a postura comandada e a postura atingida, que devem ser medidas em relação a um sistema de coordenadas paralelo ao sistema da base indicado anteriormente na figura 7.2a. Esta figura apresenta também o ponto de medição, neste caso tomado como o centro da ferramenta acoplada à interface mecânica.

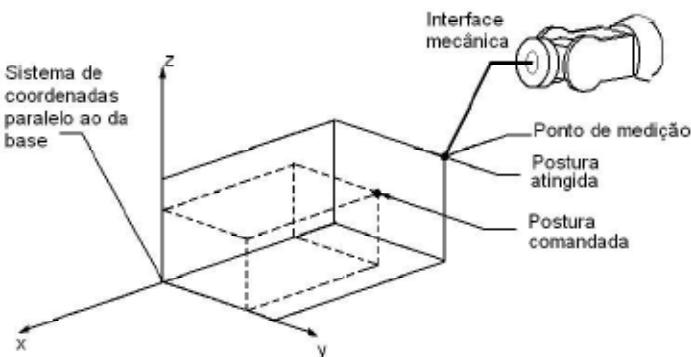


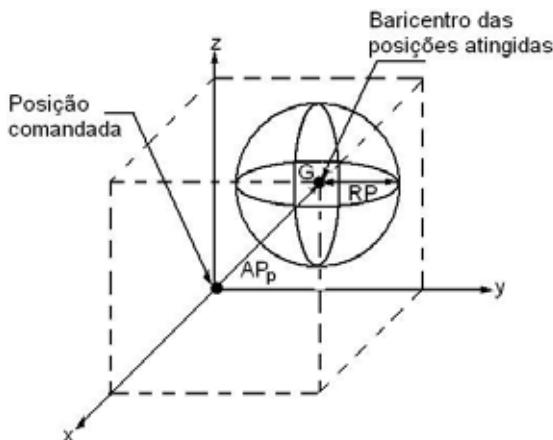
Figura 7.3: Postura comandada e postura atingida.

Segundo a norma ISO, o desempenho é quantificado através da precisão de postura, da repetibilidade de postura, da variação multidirecional na precisão de postura, do deslocamento nas características de postura, da intercambiabilidade, da precisão e repetibilidade à distância, do tempo de estabilização, e do sobrepasso, que são descritos a seguir.

7.4.1.1 – Precisão de Postura (AP)

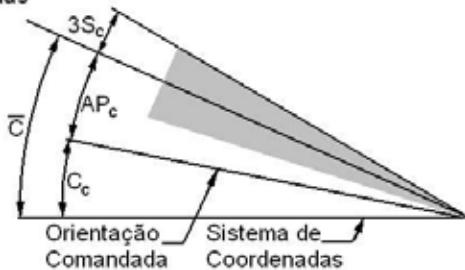
É a diferença entre a postura comandada e a média das posturas atingidas quando a aproximação é realizada pela mesma direção. É dividida em:

- Precisão de posicionamento (AP_p): diferença entre uma posição comandada e o baricentro das posições atingidas (figura 7.4a);
- Precisão de orientação (AP_o): diferença entre a orientação angular comandada e a média das orientações angulares atingidas (figura 7.4b).



a) Posicionamento

Valores médios
das orientações
atingidas



b) Orientação

Figura 7.4: Precisão e repetibilidade.

7.4.1.2 – Repetibilidade de Postura (R_p)

Expressa a proximidade das posturas atingidas após “n” vezes à mesma postura comandada na mesma direção. É quantificada através do:

- Raio da esfera centrada no baricentro do grupo de pontos atingidos (RP_p), calculado a partir da média dos raios de cada ponto atingido e do seu desvio padrão (figura 7.4a);

- Faixa de três desvios padrão das medidas angulares em torno dos valores médios dos ângulos de orientação (figura 7.4b).

7.4.1.3 – Variação Multidirecional na Precisão de Postura (vA_p)

Expressa a diferença entre as médias das posturas atingidas visitando a mesma postura comandada “n” vezes a partir de três direções ortogonais (figura 7.4). É quantificada por:

- vAP_p – a distância máxima entre os baricentros do conjunto de pontos atingidos ao final de cada percurso;
- vAP_a , vAP_b , vAP_c – o máximo desvio entre o valor médio dos ângulos atingidos ao final dos diferentes percursos.

7.4.1.4 – Deslocamento nas Características de Postura (dA_p)

Compreende o deslocamento na precisão e na repetibilidade. O deslocamento na precisão de postura (dA_p) é a variação na precisão de postura no decurso de um tempo específico. Da mesma forma, o deslocamento na repetibilidade de postura (dR_p) é a variação na repetibilidade de postura ao longo de um determinado tempo.

A tabela 7.3 resume as principais características da grandeza postura em função do tipo de aplicação.

Características	Aplicações							
	Soldagem Ponto	Transporte de materiais	Montagem	Inspeção	Deshaste, Polimento e Corte	Pintura	Soldagem a arco	Selamento
Precisão e Repetibilidade de postura	X	X	X	X		X		
Variação multidirecional na precisão de postura		X	X	X				
Deslocamento na precisão e repetibilidade de postura	X	X	X	X			X	
Precisão e Repetibilidade à distância	X	X	X	X				
Tempo de estabilização	X	X	X	X				
Sobrepasso	X	X	X	X			X	

Precisão e Repetibilidade de percurso		X	X	X	X	X
Desvios de canto		X	X	X	X	X
Precisão e Repetibilidade Flutuação na velocidade de percurso				X	X	X
Tempo mínimo de posicionamento	X	X	X			
Flexibilidade estática	X	X	X	X		

Tabela 7.3: Principais Características da grandeza Postura em função de aplicações.

7.4.2 – Resolução Espacial

Resolução Espacial é a capacidade de um robô posicionar seu punho em um ponto objetivo no interior de seu volume de trabalho (figura 7.5). A pior situação é quando este ponto está na metade da distância entre dois pontos endereçáveis.

Precisão espacial é equivalente a metade da resolução de controle.

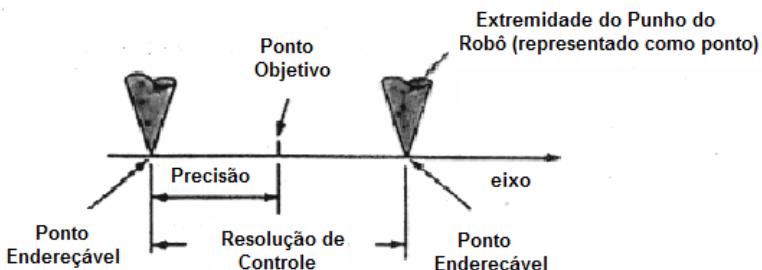


Figura 7.5: Resolução Espacial e Precisão.

7.4.3 – Repetibilidade

Repetibilidade é a medida da dispersão das posições atingidas pelo punho em uma série de tentativas do robô. Deveremos observar que:

- A distribuição do erro de repetibilidade é aproximadamente normal, para robôs com 5 ou 6 GL;
- No espaço 3D, os erros de repetibilidade circundarão um ponto P, formando uma distribuição cujo limite externo é aproximadamente esférico (figura 7.6);
- O tamanho da esfera é maior nas regiões onde o volume de trabalho é mais distante da base do robô;
- Os fabricantes especificam a repetibilidade como $+/ - R$, onde R é o raio da esfera

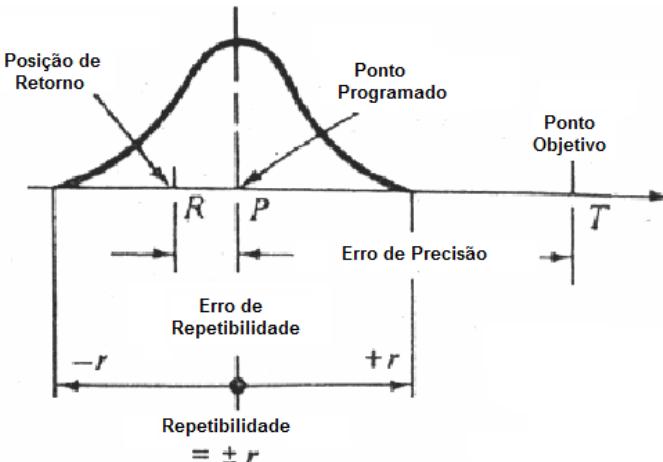


Figura 7.6: Repetibilidade.

7.4.4 – Complacência

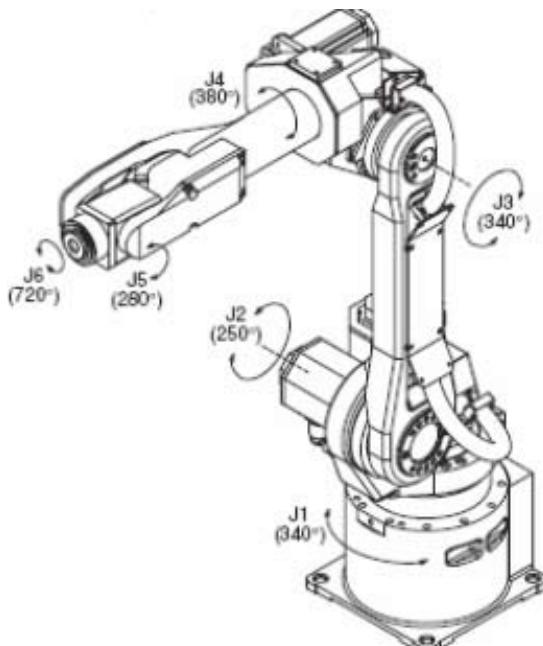
Complacência pode ser definido como a capacidade de um robô de responder com grandes ou pequenos deslocamentos do punho quando submetido a uma força ou torque.

Um manipulador rígido possui baixa complacência, enquanto um manipulador flexível possui uma alta complacência, mas ao mesmo tempo apresenta baixa precisão de movimento.

7.5 – Exemplo de Especificação de Robôs Industriais

A seguir apresentaremos as especificações técnicas do robô industrial FANUC ArcMate 100iBETM (figuras 7.7 e 7.8), Robô ABB IRB 1400TM (figura 7.9) e, finalmente, a tabela 7.4 que resume as características de alguns robôs industriais.

7.5.1 – Robô industrial FANUC ArcMate 100iBE



Items		
Axes		6
Payload (kg)		6
Reach (mm)		1373
Repeatability (mm)		±0.06
Interference radius (mm)		273
Motion range (degrees)	J1 J2 J3 J4 J5 J6	340 250 340 380 280 720
Motion speed (degrees/s)	J1 J2 J3 J4 J5 J6	150 160 170 400 400 520
Wrist moment (kgf·cm)	J4 J5 J6	160 100 60
Wrist inertia (kgf·cm·s ²)	J4 J5 J6	6.4 2.2 0.61
Mechanical brakes		All axes
Mechanical weight (kg)		138
Mounting method ⁽¹⁾		Upright, inverted, wall and angle mount
Installation environment:		
Temperature °C		0 to 45
Humidity		Normally: 75% or less Short term (within a month): 95% or less No condensation
Vibration (m/s ²)		4.9 or less
Payload at axis 3 (kg)		12

Figura 7.7: Especificações Técnicas do Robô Industrial
FANUC ArcMate 100

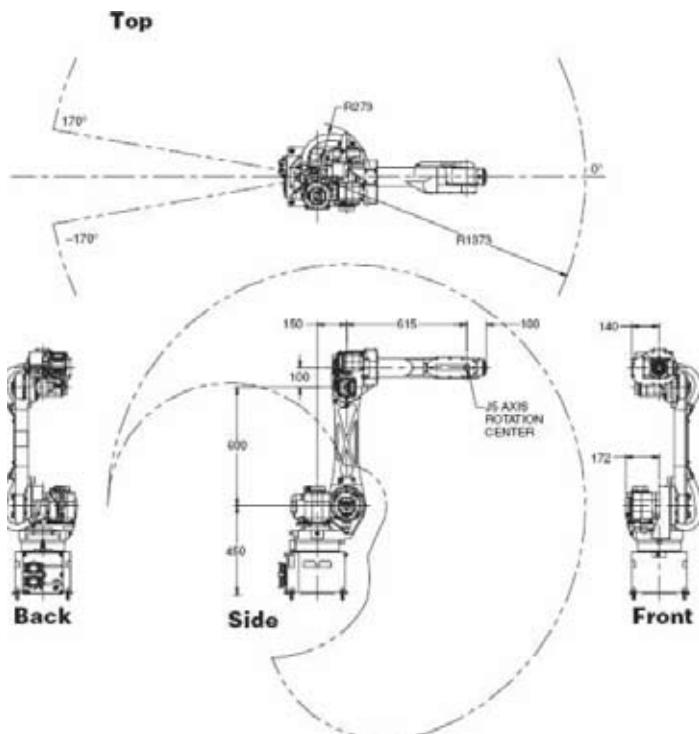


Figura 7.8: Volume de Trabalho do Robô Industrial
FANUC ArcMate 100iBE.

7.5.2 – Robô industrial FANUC ArcMate 100iBE



Junta	θ (graus)	d (mm)	α (graus)	a (mm)
01	-360°	475	-90°	0
02	-90°	0	0°	360
03	180°	0	90°	0
04	0°	720	-90°	0
05	0°	0	90°	0
06	0°	85	0°	0

Junta	θ_{min} (graus)	θ_{max} (graus)	W_{max} (graus/s)	α_{max} (graus/s²)
01	-170°	170°	120	670
02	-70°	70°	120	660
03	-65°	70°	120	1130
04	-150°	150°	280	3290
05	-115°	115°	280	3290
06	-300°	300°	280	3290

Figura 7.9: Especificações Técnicas do Robô Industrial IRB 1400.

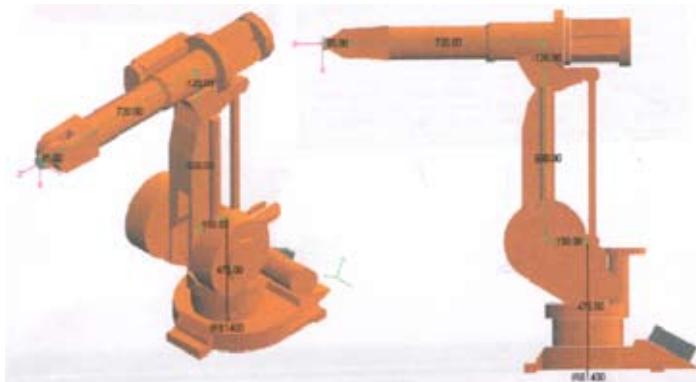


Figura 7.10: Dimensões do Robô Industrial IRB 1400.

7.5.3 – Características de outros Robôs industriais

MODELOS			
CARACTERÍSTICAS	PUMA 760	IRB 60/2	7540 IBM
Capacidade de Carga	10 Kg	80 Kg	25 Kg
Coordenadas	Angulares	Angulares	Cartesianas
Graus de Liberdade	5	5	4
Acionamento	Servomotores Elétricos	Servomotores Elétricos	Servomotores Motores de Passo Pistão Pneumático
Repetibilidade	+/- 0,2 mm	+/- 0,4 mm	+/- 0,05 mm
Peso	222 Kg	400 Kg	276 Kg
Alcance Máximo	1.250 mm	2.288 mm	100 mm (eixo z)
Controlador	L51-11	Multimicroprocessadores adaptado a um computador IBM PC	Baseado em microprocessador 6602
Capacidade de Memória	16 K	470 posições	5.103 bytes
Línguagem de Programação	VAL	----	AML/ENTRY

Velocidade Máxima	1 m/s	1,35 m/s	1,35 m/s
Alimentação e Consumo	220-440 V AC 6,3 Kw	220 V AC 4 Kw	220 V AC 1,6 KvA
Aplicações	Montagem, Soldagem	Mecanização, Alimentação, Manipulação, Soldagem, etc.	Montagem, Paletização, Polimento, etc.
Fabricante	UNIMA- TION	ASSEA	IBM

Tabela 7.6: Especificação de Robôs Industriais.

7.6 - Referências Bibliográficas

ISO 9787:1990: Manipulating Industrial Robots: Coordinate systems and motions – ISO Publications, França, 1990, 33 pp.

ISO 9283 Second edition 1998 - Manipulating Industrial Robots: Performance criteria and related test methods - ISO Publications, França, 1998, 33 pp.

ISO/TR 13309 First edition 1995 -Manipulating industrial robots – Informative guide on test equipment and metrology methods of operation for robot performance evaluation in accordance with ISO 9283 - ISO Publications, França, 1995, 33 pp.

ROMANO, V.F. e outros: Robótica Industrial: Aplicação na Indústria de Manufatura e Processos, São Paulo: Edgard Blucher Ltda, 2002 p 20-46

CAPÍTULO 8

Programação de Robôs Industriais



CAPÍTULO 8

Programação de Robôs Industriais

Um robô industrial pode ser definido como um sistema mecânico articulado que tem como objetivo principal executar operações pré-definidas. Isto é realizado através de um Supervisor de Controle que deverá especificar o que o manipulador deverá fazer para que o mesmo possa realizar tarefas especificadas. Normalmente ele é constituído de seis graus de liberdade, e o posicionamento de sua ferramenta de trabalho é especificado através do controle de modo apropriado das suas variáveis articulares.

Neste capítulo é apresentada a estrutura completa de supervisão e controle de um robô industrial, enfatizando as formas de programação do mesmo a partir do teach-pendant e através de softwares de programação off-line de robô. Este capítulo se destina aos utilizadores de robôs industriais, com ênfase na sua programação e utilização. A apresentação inicial é realizada de forma genérica, e depois particularizada em cima de exemplos de aplicações.

8.1 – Introdução

A utilização de robôs no processo produtivo exige flexibilidade de utilização e procedimentos sistemáticos para alteração de programação sem interromper o ciclo produtivo. Uma das principais vantagens na utilização de robôs é sua flexibilidade efetiva de programação e capacidade de realizar tarefas, sua habilidade para se adaptar para novas linhas de automatizadas produções e sua grande faixa de movimentos. (E. Trostmann, 1988).

Com o intuito de melhor utilizarmos os recursos desses dispositivos num processo automatizado, tornase quase estreitamente necessário a formação de profissionais na área de Robótica, através de procedimentos metodológicos que levem a programação de tarefas a serem executadas através de robôs, utilizando processos de aprendizagem conhecidos como programação ***online*** e programação ***off-line***.

Atualmente a programação offline vem sendo utilizada cada vez mais como ferramenta de concepção de sistemas automatizados e programação de robôs, aumentando a flexibilidade e habilidade de utilização dos mesmos, com uma variedade ilimitada de cenários e movimentos. Programação offline deve ser considerada como os processos por meio dos quais são realizadas as programações de robôs em ambientes de operação complexas, sem a necessidade dos dispositivos automatizados e do próprio robô.

As informações relativas às coordenadas dos pontos são geradas, como também os dados de funções e ciclos lógicos. Desenvolvimentos na tecnologia de robô, tanto ao nível de hardware quanto software, estão fazendo da programação offline um método mais possível, acarretando na melhoria do controle dos robôs, precisões de posicionamento e adoção de tecnologias de sensores.

8.2 – Estrutura de Controle de um Robô Industrial

A trajetória de um robô é definida através de um conjunto de ângulos associados ao movimento angular de cada grau de liberdade do robô, que, utilizando um algoritmo de interpolação, servirá como sinal de referência para o controlador de posição de cada junta robótica que realizará uma comparação com os sinais provenientes dos transdutores de posição das juntas.

Diversas aplicações industriais exigem que o robô trabalhe de acordo com a posição e orientação do seu elemento terminal em relação ao sistema de coordenadas de trabalho, como, por exemplo, um robô trabalhando em conjunto com uma máquina de comando numérico (CNC), numa célula automatizada com outros robôs, ou ainda quando o mesmo é dotado de um sistema de visão.

Neste último caso, a interpretação das imagens se efetuará em relação ao sistema de coordenadas de trabalho (em duas ou três dimensões), e as informações extraídas das mesmas serão transmitidas ao Sistema de Supervisão após tratamento apropriado.

O Supervisor de Controle é responsável pela geração dos sinais de referência individuais ao longo do tempo, para cada junta do robô. Através de uma malha de controle de posição independente para cada junta, estes sinais são comparados com os valores atuais (obtidos através dos sensores de posição articulares), que faz com que a configuração de um robô seja controlada a partir de um valor desejado, independente do movimento realizado e da carga transportada pelo robô.

8.3- Programação de Tarefas em Robôs Industriais

8.3.1 Considerações Iniciais

Os robôs industriais são equipamentos multifuncionais reprogramáveis com grande flexibilidade de operação. Atual-

mente, a programação de tarefas é realizada através de uma “caixa de aprendizagem” que é utilizada para conduzir o robô através das posições críticas do ciclo de operação. Este tipo de programação de tarefas possui alguns inconvenientes, entre eles não utilizar o robô no período de programação e não permitir um controle mais preciso sobre a trajetória da garra ou ferramenta.

A programação de tarefas off-line não apresenta os inconvenientes acima por ser realizada em computadores, necessitando apenas de um modelo matemático. Este modelo contém informações sobre a cinemática e dinâmica do robô.

Normalmente a programação de tarefas de robôs é realizada no espaço das juntas, não necessitando de um modelo geométrico, e a trajetória angular de mesma natureza dos sinais provenientes do transdutor de posição servirá como sinal de referência para o controlador de cada junta robótica (figura 8.1). Entretanto, a realização de algumas tarefas em relação a um sistema de referência colocado na ferramenta (espaço cartesiano) exige o conhecimento completo do modelo geométrico, e torna necessária a utilização de uma transformação de coordenadas, tendo em vista que o sinal de referência correspondente à trajetória, necessária para o controle das juntas, deverá ser angular.

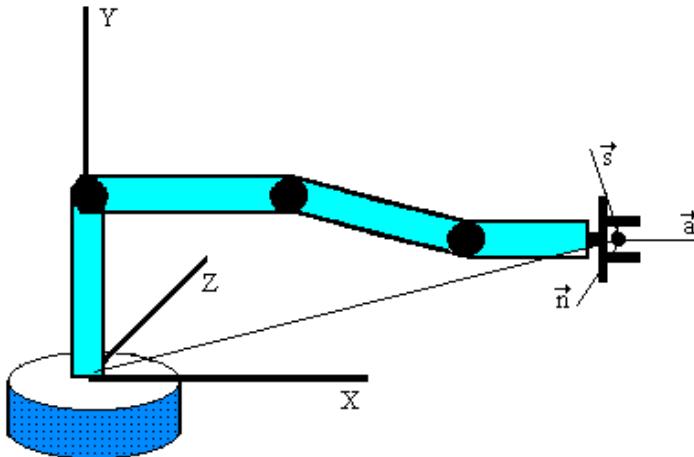


Figura 8.1: Espaço de coordenadas de um Robô.

Num robô industrial, os diferentes graus de liberdade podem ser associados a diversos sistemas de coordenadas, cada um associado a um grau de liberdade e servindo para descrever o movimento de cada grau de liberdade associado.

O modelo geométrico é aquele que expressa a posição e orientação da garra em relação a um sistema de coordenadas fixo à base do robô, em função de suas coordenadas generalizadas (angulares no caso de juntas rotacionais). Esta relação pode ser expressa matematicamente a partir de uma matriz que relaciona o sistema de coordenadas da base com o sistema de coordenadas do último elemento, designada matriz de passagem homogênea do robô.

A tarefa de um robô é especificada em termos de coordenadas cartesianas, \underline{x} . Estas consistem na posição, descrita por um vetor posição \underline{p} , e um vetor orientação, descritos por um vetor unitário de aproximação \underline{a} e um vetor unitário de deslizamento \underline{s} . Todos eles são definidos em relação ao sistema de coordenadas da base. Por conveniência, um vetor unitário normal é definido como $\underline{n} = \underline{s} \times \underline{a}$, onde \times denota o produto vetorial, figura 8.2.

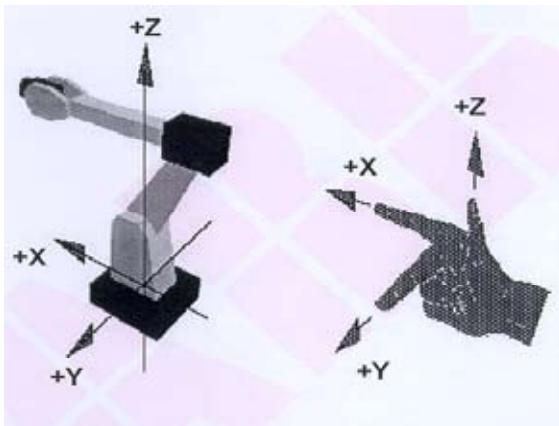
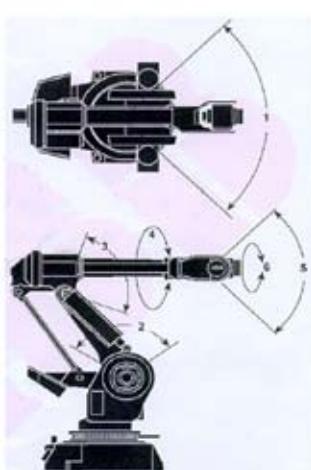


Figura 8.2: Vetores de posição e de orientação de um robô.

Durante o processo de programação de um robô industrial, torna-se importante o conhecimento por parte do operador dos eixos de movimentação do mesmo, conforme é apresentado na figura 8.3.



Eixos Inferiores:

Eixo 1: Giro da Base.

Eixo 2: Para frente e para trás (braço superior).

Eixo 3: Para cima e para baixo (braço superior).

Eixos Superiores:

Eixo 4: Giro do braço superior.

Eixo 5: Quebra do punho.

Eixo 6: Giro da Flange ou Giro da ferramenta.

Figura 8.3: Definição dos eixos de movimentação de um robô ABB-TM

1400TM

8.3.2 Programação de Robôs Industriais

Os métodos de programação mais frequentemente utilizados em robôs industriais são:

- **Aprendizagem ponto-a-ponto:** o robô é manipulado através de um processo de aprendizagem de tarefas até as posições desejadas. Ela pode ser realizada através da:
 - a) **Movimentação angular das juntas:** Neste método são gravados pontos de referência fornecidos pelos transdutores de posição localizados em cada junta e a partir desses pontos são geradas trajetórias (angulares) através da utilização de algoritmos de interpolação. A simplicidade deste método não exige o conhecimento do modelo geométrico do robô (figura 8.4);

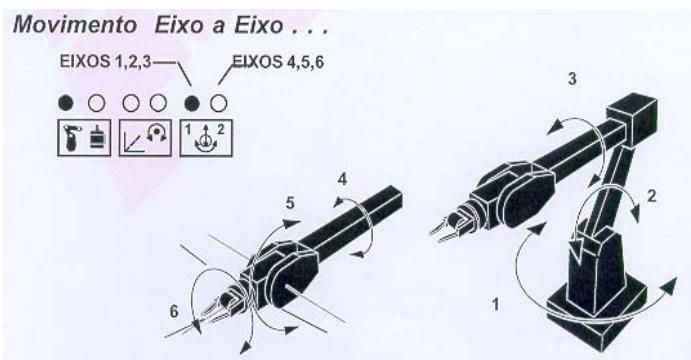


Figura 8.4: Movimento angular das juntas.

- b) **Movimento na direção Cartesiana:** Idêntico ao procedimento descrito anteriormente, mas a obtenção dos pontos de referência é realizada utilizando o modelo cinemático do manipulador, possibilitando assim a operação do robô seguindo as direções X, Y e Z (figura 8.5);



Figura 8.5: Movimento em Coordenadas Retangulares.

c) **Movimento de Reorientação da Ferramenta:** Idêntico ao procedimento descrito anteriormente, mas a obtenção dos pontos de referência é realizada utilizando a orientação da ferramenta em torno da rotação dos eixos X, Y, Z, correspondendo as três rotações Roll, Pitch, Yaw, como será mostrado no próximo capítulo (figura 8.6);

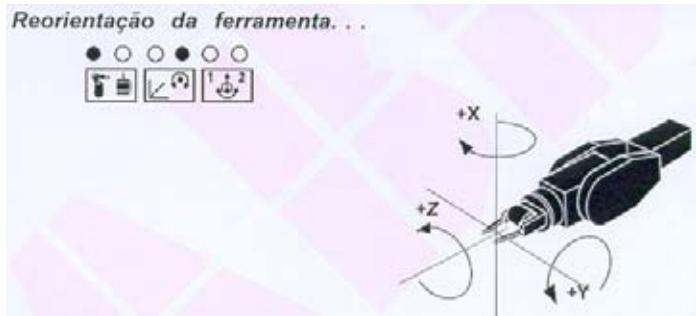


Figura 8.6: Reorientação da Ferramenta.

- **Programação off-line:** A partir de um software para visualização gráfica do modelo geométrico de robôs, devem ser obtidos pontos de passagem correspondentes à trajetória do robô (expressos em coordenadas angulares). Esses pontos de passagem poderão ser obtidos a

partir do movimento angular de cada junta, ou a partir do modelo geométrico. A partir de um conjunto de pontos correspondentes à trajetória a ser realizada pelo robô, torna-se possível a implementação de algoritmos off-line para interpolação e filtragem, levando-se em consideração aspectos dinâmicos e testes de colisão.

- **Programação on-line:** A partir do conhecimento do modelo geométrico e das características da trajetória desejada (posição final, velocidade e forma do caminho), pode-se implementar algoritmos para modelagem cinemática inversa e controle de posição.

Os métodos descritos anteriormente estão associados com o tipo de aplicação requerida. Isto permitirá:

- a realização de tarefas mais precisas e complexas;
- a necessidade da utilização de posições determinadas analiticamente, ou informações provenientes de sensores de percepção externa;
- ao uso de manipuladores em ambientes adversos à presença do homem;
- a necessidade do aumento do tempo útil de trabalho, pois durante a programação da tarefa pelo método de aprendizagem o manipulador é utilizado, diminuindo o seu tempo útil.

8.3.3 - Painel de Açãoamento e Controle

O painel de açãoamento e controle é uma interface homem-máquina que possui diversos nomes na literatura, de acordo com o fabricante de robô, tais como “Teach-in-Box” e “Teach Pendant”, etc. A figura 8.7 mostra o “Teach Pendant” e a unidade de controle de um robô industrial, onde serão armazenados e gravados todos os dados de programa, e a tabela 8.1 mostra a listagem dos principais comandos utilizados por um robô industrial.



a) Teach-Pendant



b) Controlador

Figura 8.7: Painel de Acionamento e Controle de um Robô Industrial.



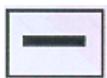
Ajuste de contraste



Robô/eixo externo



Jogging



Funções/Menu



Program



0

Teclado numérico



I/Os



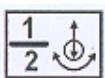
Backspace



Miscellaneous



TABULAÇÃO

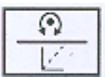


Selecionamento de eixos

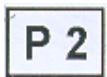


P 1

Tecla Programável



Reorientação/Cartesiano

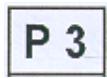


P 2

Tecla programável



Parada suave/ Stop

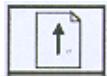


P 3

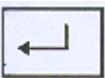
Tecla programável



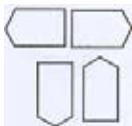
Atalho para velocidade



PgUp e PgDn



Enter



Cursos

Tabela 8.1: Lista de Comandos - Unidade de Programação de um Robô (Teach-Pendant).

8.3.4 – Elementos de Supervisão e Controle de um Robô Industrial

8.3.4.1 - Unidade de Armazenamento

O armazenamento de parâmetros, programas e informações de um robô industrial são realizados em RAM-DISK ou disquete. Os principais tipos de informações geralmente armazenadas são na forma de programas, lista de mensagens de erro, parâmetros utilizados e módulos de programação. A figura 8.8 mostra uma tela típica de programação do robô ABBTM.

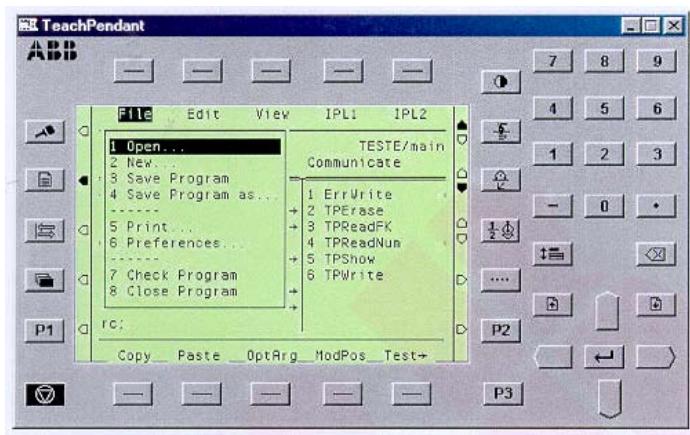


Figura 8.8: Tela Típica de Programação – Robô ABBTM.

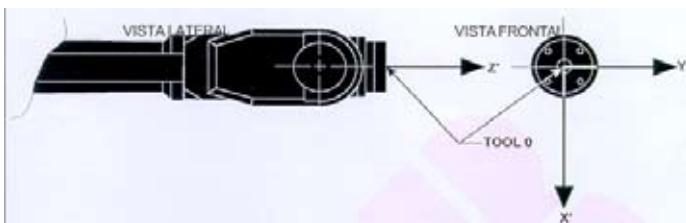
8.3.4.2 – Calibração de um Robô e Acessórios

Os modos descritos anteriormente para movimentação dos robôs são associados com a melhor forma de aprendizagem, função principalmente de uma boa especificação funcional. Isto permitirá a realização de tarefas mais precisas e complexas sem a necessidade da utilização de posições determinadas analiticamente, ou informações provenientes de sensores de percepção externos; embora implique numa necessidade do aumento do tempo útil de trabalho, tendo em

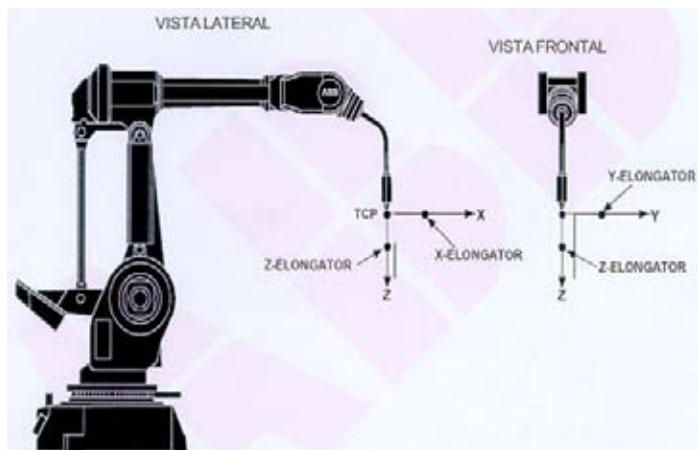
vista que, durante a programação das tarefas pelo método de aprendizagem, o manipulador obrigatoricamente é utilizado.

Existe a necessidade de realizar alguns procedimentos ao inicializarmos um robô industrial, dentre eles devemos considerar, para um perfeito funcionamento do sistema e melhor performance do robô, dois procedimentos básicos:

- Sincronização das Juntas: Os contadores das informações provenientes dos encoders (ou resolvers no caso de interrupção de energia ou desconexão do robô) deverão ser zerados, com o robô posicionado nas marcas de calibração.
- Calibração da Ferramenta Terminal (TCP): Nesta etapa devemos inicialmente informar ao sistema de controle do robô (painel de comando) as características principais da ferramenta (massa, dimensão, etc.), para que o sistema de controle possa realizar todo o procedimento de calibração da ferramenta terminal do robô (figura 8.9).



a) Definição do Ponto Central da Ferramenta (TCP) – Robô ABBTM.



b) Ferramenta Terminal.

Figura 8.9: Procedimento de Calibração da Ferramenta Terminal.

8.4 – Métodos de Programação de Robôs Industriais

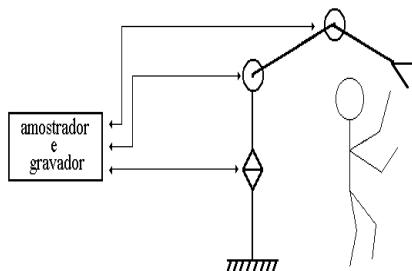
8.4.1 - Programação por aprendizagem

Programar visa o estabelecimento de uma sequência de operações a serem executadas pelo robô. A programação das tarefas pode ser realizada através de uma programação por aprendizagem ou a partir de uma linguagem de programação de computadores. A programação por aprendizagem pode ser realizada pelos seguintes métodos (figura 8.10):

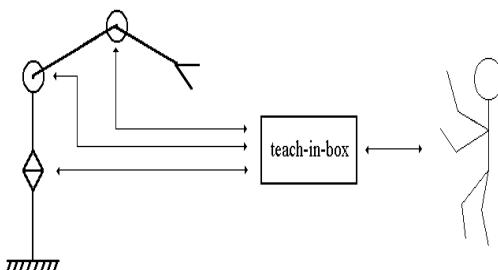
- *Aprendizagem direta (Walk Through)*: Neste método o operador guia fisicamente o robô por seu órgão terminal. Enquanto isto, os sensores de posição de cada junta do robô são utilizados para memorizar os pontos importantes da tarefa a ser executada. Este método é indicado para robôs de estrutura mecânica leve, de movimentos reversíveis e para a utilização em ambientes não “hostis”.
- *Aprendizagem por simulação física*: Neste método, o operador guia um simulador físico, que tem a geometria e os

sensores idênticos ao robô original. Uma vez memorizada a tarefa, esta é transferida para o sistema de controle do robô. Este método é indicado para robôs de grande porte ou com estruturas mecânicas não reversíveis, bem como a programação em ambientes que exijam distância. Este método necessita de um modelo preciso e conhecido.

- *Aprendizagem por tele-comando:* Neste método, um dispositivo de telecomando (teach pendant ou teach-in box) é utilizado para mover cada junta do robô isoladamente ou fornecer a posição e orientação da garra. Este método é adequado para qualquer tipo de robô e é mais barato que o método anterior, caso o telecomando seja feito a partir de um dispositivo de programação que permita a atuação sobre cada ligação independentemente.



a) O operador guia o robô diretamente



b) Utilização de simulador ou telecomando

Figura 8.10: Programação por aprendizagem.

8.4.2 - Método de Aprendizagem (Teaching)

No processo de aprendizagem, um “*Teach-in-box*” (painel de acionamento e controle) é utilizado para movimentação do robô através de algumas sequências de movimentos e interação com o processo por meio da ferramenta terminal, como, por exemplo, a alimentação do arame num processo de soldagem, um processo de manipulação com abertura e fechamento de uma garra.

Um *Teach Pendant* consiste em teclas e chaves de controles, onde podemos movimentar os diferentes graus de liberdade do robô (movimento das juntas ou em relação à ferramenta de trabalho). O controle do robô deve estar ajustado no modo de ensinamento para a utilização do processo de ensinamento. Informações de movimentação e outros dados necessários são gravados pelo controle do robô com o robô guiado através do caminho desejado durante o processo de ensino. A figura 8.11 apresenta um exemplo de método de aprendizagem de tarefas utilizando o *Teach Pendant* para o robô Motoman™ da Yaskawa, e um exemplo de programa na linguagem INFORM II, cujas informações referentes ao tipo de movimento e velocidade de execução são apresentadas na cor azul.



a) Método de Aprendizagem.

```
//INST
///DATE 2002/04/08 19:55
///ATTR SC,RW
///GROUP1 RB1,BS1
NOP
MOVJ C0000 BC0000 VJ=50.00
SET B010 0
SUB P020 P020
*A
MOVJ C0001 BC0001 VJ=50.00
MOVL C0002 BC0002 V=50.0
MOVL C0003 BC0003 V=33.3
TIMER T=1.00
DOUT OT#(9) OFF
DOUT OT#(10) ON
TIMER T=1.00
MOVL C0004 BC0004 V=66.7
SFTON P020 UF#(20)
MOVL C0005 BC0005 V=75.0
MOVL C0006 BC0006 V=50.0
TIMER T=1.00
DOUT OT#(10) OFF
DOUT OT#(9) ON
TIMER T=1.00
MOVL C0007 BC0007 V=75.0
SFTOF
ADD P020 P021
INC B010
JUMP *A IF B010<3
MOVJ C0008 BC0008 VJ=50.00
MOVJ C0009 BC0009 VJ=50.00
END
```

b) Exemplo de Programação.

Figura 8.11: Programação por Aprendizagem (Teaching).

Em determinados pontos de uma determinada trajetória o operador poderá também posicionar ou programar equipamentos periféricos relacionados com o volume de trabalho do robô, utilizando interfaces para outros equipamentos (normalmente realizado através de I/O's).

Através da edição de um programa torna-se possível a estruturação de um programa e o incremento de informações adicionais para o programa de controle de movimento para operação automática do robô ou a um periférico associado, correção ou modificação de um programa de controle existente para alteração de pontos incorretos e velocidades de trabalho.

Na programação por aprendizagem, os pontos da trajetória, na aprendizagem, são amostrados com uma taxa elevada e armazenados na memória. Estes pontos podem ser amostrados no controle segundo uma taxa programada, conduzindo o robô, continuamente, por inércia ou por “arrasto”, de maneira precisa e em uma velocidade que é função da taxa de amostragem.

8.4.3 - Programação PTP (Point to point)

A programação de tarefas de robôs através da aprendizagem de pontos de passagem (PTP) consiste na movimentação das juntas a partir de uma determinada posição até a próxima através de um processo de aprendizagem. Neste modo de programação não é necessário o conhecimento do caminho a ser executado pelo manipulador. Geralmente, cada eixo movimentase numa porcentagem da velocidade máxima de cada junta até que ele alcance a posição desejada.

Para garantir a sincronização de movimentos das juntas, todos os graus de liberdade iniciarão e complementarão os seus movimentos simultaneamente, através de velocidade máxima de junta indicada pelo operador.

Na Programação ponto a ponto são gravados apenas os pontos essenciais da trajetória do órgão terminal, de modo que o movimento entre dois pontos consecutivos gravados fica determinado pelos algoritmos de controle utilizados nos

servo-mecanismos das diversas ligações. Este método é adequado para aplicações que não requeiram um controle preciso da trajetória e da velocidade intermediárias aos pontos essenciais da tarefa.

8.4.4 - Programação CP (Controlled Path)

A programação de tarefas de robôs através do posicionamento/orientação da ferramenta, também chamada *Controlled Path* (CP), concerne no controle das diferentes juntas utilizando a modelagem cinemática inversa, de modo a realizar um caminho desejado entre dois pontos programados. Neste método de controle, cada eixo move suavemente e proporcionalmente para fornecer um previsível movimento de caminho controlado.

8.4.5 – Programação por Aprendizagem Direta

Neste método o programador posiciona manualmente o braço do robô através da movimentação dos seus diferentes graus de liberdade, por meio do contato físico, deslocando e controlando o braço do robô através de posições desejadas dentro do envelope de trabalho do robô, conforme mostra a figura 8.12.

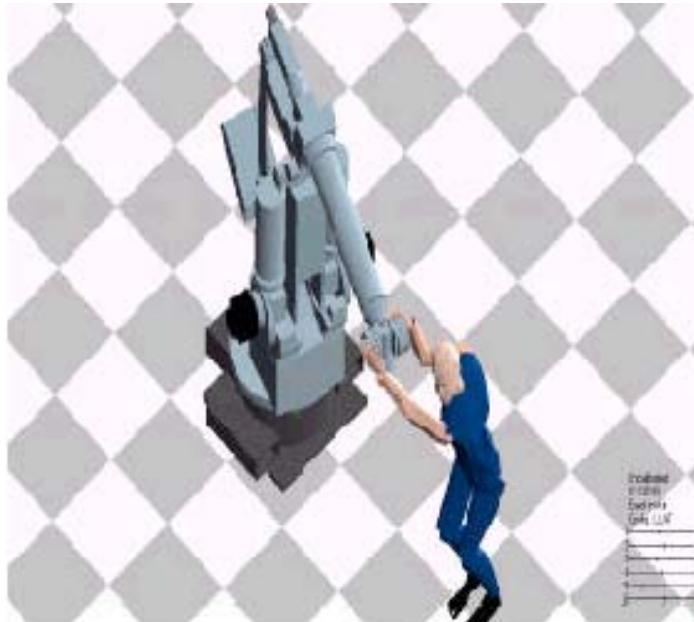


Figura 8.12: Programação por Aprendizagem Direta.

Durante esse tempo o controle do robô irá varrer e armazenar valores de coordenadas de posição dentro de uma base de tempo fixada. Serão gravadas as posições e outras informações funcionais dentro de uma memória, para rodar em uma forma “playback” (executando a repetição das posições gravadas e suas informações funcionais), como foi ensinado durante a produção. Neste método a velocidade pode ser controlada de forma independente.

O algoritmo utilizado para controle de movimentos das juntas robóticas é realizado através da interpolação linear e filtragem dos pontos adquiridos durante o procedimento de gravação de uma trajetória desejada, *método “Continuous Path” (CP)*.

A partir da posição de cada junta, que foi gravada pela unidade de controle dentro de uma base de tempo constante pela varredura dos sinais dos “encoders” de posição das juntas durante o movimento do robô, são inseridos pontos espaçados próximos

aos que foram gravados ao longo de um caminho desejado. O algoritmo permite a repetição deste movimento, atendendo condições de velocidade e aceleração impostas pelo operador.

8.4.6 – Programação Master-Slave

Outra forma muito utilizada de processo de aprendizagem é a utilização de um sistema do tipo mestre-escravo (*master-slave*), em que, a partir da movimentação do sistema robótico mestre, o sistema robótico escravo realiza as movimentações das juntas. A figura 8.13 apresenta o sistema *master-slave Kraft-GripsTM* utilizado para intervenções submarinas automatizadas.



a) Manipulador Escravo (Slave).



b) Manipulador Mestre (Master).

Figura 8.13: Manipulador Submarino Kraft-GripsTM – Arquitetura *master-slave*.

8.5 – Linguagem de Programação de Robôs

Com o atual avanço na tecnologia de desenvolvimento de hardware e software, a programação de robôs através de linguagens está cada vez mais próxima da realidade industrial. Estes avanços incluem uma maior sofisticação dos controladores, melhor precisão de posicionamento e incremento no número e tipos de sensores.

A programação utilizando linguagens pode ser considerada como o processo pelo qual os programas são desenvolvidos sem a necessidade do uso do robô propriamente dito, mas através da utilização de um ambiente computacional de programação de computador. Com o atual avanço na tecnologia de desenvolvimento de hardware e software, a programação de robôs por linguagens está cada vez mais próxima da realidade industrial. Estes avanços incluem uma maior sofisticação dos controladores, melhores precisão de posicionamento e incremento no número e tipos de sensores.

As linguagens de programação são normalmente interativas, permitindo a implementação de programas estruturados e fácil interatividade com o usuário. Elas variam de acordo com o fabricante de robôs.

8.5.1 - Níveis de programação

As linguagens de programação de robôs podem ser classificadas nos seguintes níveis:

- Nível de junta: As linguagens classificadas neste nível requerem a programação individual de cada junta do robô para que uma dada posição seja alcançada.
- Nível de manipulador: Neste nível é necessário apenas fornecer a posição e orientação do órgão terminal e o sistema se encarrega de obter, através do modelo geométrico inverso do robô, as posições de cada junta.
- Nível de objeto: Neste nível são necessários apenas as especificações relativas ao posicionamento de objetos no interior do volume de trabalho do robô. Deste modo, é necessária a existência de um modelo matemático que represente o ambiente de trabalho no qual o robô se encontra.
- Nível de objetivo: Neste nível a tarefa não é realmente descrita, mas definida, como por exemplo montar as peças A, B e C. Neste caso é necessário, além do conhecimento do modelo do ambiente, um conjunto de dados relativos a uma determinada tarefa.

8.5.2 - Vantagens da utilização de programação através de linguagens

Entre as principais vantagens da utilização da programação através de linguagens é possível citar:

- Redução do tempo em que o robô está fora da linha de produção, pois ele pode continuar operando enquanto a sua próxima tarefa está sendo programada;
- O programador não necessita entrar em contato com o ambiente de trabalho;
- Possibilidade de integração com sistemas CAD-CAM permitindo uma maior integração entre as fases de projeto e de produção e, consequentemente, reduzindo o tempo do processo de produção;
- Simplificação da programação de tarefas com alto grau de complexidade. As linguagens de programação facilitam a elaboração de tarefas complexas, pois possuem formas analíticas de geração de trajetórias e análise de dados provenientes de sensores externos, além de várias estruturas de controle etc;
- Segurança na geração de trajetórias. As linguagens podem internamente gerar um modelo geométrico do ambiente de trabalho do robô e, através de algoritmos de detecção de colisão, produzir trajetórias seguras e simulá-las por software antes da execução final.

8.5.3 – Procedimentos Básicos para Implementação e Execução de Programas

8.5.3.1 – Aspectos Relacionados à Segurança

Dentro os aspectos relativos à segurança na utilização de robôs industriais está a programação online, principalmen-

te no método de aprendizagem direto. O programador se encontra dentro da região de trabalho do robô, com potencial de risco de acidente, necessitando assim de dispositivos de segurança. Os seguintes procedimentos deverão ser considerados:

- 1- Respeitar sempre a área de trabalho utilizando sempre o bom senso.
- 2- Utilização de travas de segurança (utilizando sensores nas entradas I/O), como, por exemplo, porta com sensor, cortina de luz, tapete com sensor, sensor de presença, etc...).
- 3- Modo de Operação: durante a aprendizagem na criação de pontos, trabalhar sempre em modo velocidade reduzida (< 250mm/s).
- 4- Utilizar chave de três posições (off/on/off) - *Safety Pad*, que possibilita energizar os motores em modo manual.
- 5- Chave de Emergência (stop mecânico): normalmente vermelho, localizado ao lado da unidade de programação e no gabinete controlador do robô, e chave de Parada (Stop), realizada através de software, permitindo uma parada suave do robô.

8.5.3.2 – Planejamento do Programa

Para melhorar a performance de um programa e diminuir o seu tempo de implementação e validação, antes de iniciar a implementação de um programa num robô industrial o usuário deverá tomar os seguintes procedimentos básicos:

1. Planejamento do programa.

- a. Conhecer o processo a ser automatizado.
- b. Conhecer as variáveis de controle.
- c. Saber sequência de lógica do processo.
- d. Visando a segurança.
- e. Definir nomes, rotinas, I/O.

2. Verificar o Sincronismo das juntas.

3. Criar, definir e ativar o TCP da ferramenta.

8.5.4 – Programação de um Robô utilizando linguagem própria

O desenvolvimento deste item foi baseado na linguagem de programação RAPIDTM, disponível nos robôs industriais fabricados pela ABBTM. Os comandos e estrutura de programação utilizados são baseados na linguagem computacional PASCAL, tendo forte ênfase a programação estruturada. Assim, as funções e procedimentos podem ter argumentos e as variáveis podem ser globais ou locais (a uma função ou procedimento), com possibilidades de implementação de recursos e módulos. A figura 8.14 apresenta a lista de instruções da linguagem de programação RAPIDTM do robô ABBTM.

Instruções Básicas	IO's
9. MoveJ 11. WaitUntil 10. MoveL 12. WaitTime 11. MoveC 13. Wait Di 12. Test 14. Set 13. ProcCall 15. Reset 14. Return 16. Break 15. For 17. Exit 16. While 18. Stop 17. Compact If 110. Label 18. If 20. Comment	1. InvertDo 2. PulseDo 3. Reset 4. Set 5. SetAO 6. SetDO 7. SetGO
Comunicação	Interrupções
1. Close 2. ErrWrite 3. Open 4. TpErase 5. TpReadFK 6. TpReadNum 7. TpWrite 8. Write	1. Connect 2. IDDelete 3. IDDisable 4. IEnable 5. ISignalDI 6. ISleep 7. ITimer 8. IWatch
Parâmetros	Funções Matemáticas
1. AccSet 11. PdispSet 2. ConfJ 12. SingArea 3. ConfL 13. SoftAct 4. EoffsOff 14. SoftDeact 5. EoffsOn 15. VelSet 6. EoffsSet 16. ActUntil 7. GripLoad 17. DeactUntil 8. LimConfL 18. SearchC 9. PdispOff 110. SearchL 10. PdispOn 20. SpotL	1. := 2. Add 3. Clear 4. Decr 5. Incr
Erros e Sistema	
1. Exit 2. Raise 3. Retry 4. Return 5. ClkReset 6. ClkStart 7. ClkStop 8. Test	

Figura 8.14: Lista de Instruções da linguagem RAPID™ do robô ABBTM.

A programação de um robô industrial utiliza uma linguagem de programação direta, como por exemplo a instrução dada pela linha de programação: ***MoveL phome, v500, z5, tool0*** significa que o robô está se movendo para a posição ***Phome***, a uma velocidade de 500 mm/s, com uma zona de precisão de 5mm, e como TCP a ferramenta ***Tool0***.

Para a implementação de programas iniciais pelo usuário, sem conhecimento aprofundado em linguagens de programação estruturada, na implementação dos primeiros programas, o usuário poderá implementar tarefas de um robô utilizando três tipos de comandos básicos para movimentação:

MoveJ → é o movimento que o robô faz saindo de um ponto e chegando ao ponto de destino movendo todas as juntas necessárias e de qualquer forma para chegar. Neste movimento obtém-se as maiores velocidades.

MoveL → é o movimento utilizado para fazer trajetórias retas (movimento linear), ou seja, o robô funciona reorientando a ferramenta de trabalho na direção dos três eixos cartesianos (X,Y,Z) e sempre movimentando os eixos necessários para fazer uma trajetória reta.

MoveC → é o movimento utilizado para fazer trajetórias circulares (movimento circular), ou seja, o robô funciona reorientando a ferramenta de trabalho em torno da rotação dos eixos X, Y, Z.

A seguir apresentaremos alguns exemplos utilizando a linguagem própria do robô industrial ABBTM, permitindo ao leitor o entendimento dos procedimentos básicos para estruturação de um programa. Segue alguns exemplos baseado na linguagem de programação RAPIDTM utilizada no robô ABBTM.

A figura 8.15 apresenta dois exemplos de programas implementados num Robô ABB 140TM, correspondentes a execução pela ferramenta terminal de um quadrado e de um círculo. Podemos observar que na instrução MoveL (movimento linear) o robô funciona reorientando a ferramenta de trabalho na direção dos três eixos cartesianos (X,Y,Z) e sempre

movimentando os eixos necessários para fazer uma trajetória reta, e na instrução MoveJ (movimento circular), o robô funciona reorientando a ferramenta de trabalho em torno da rotação dos eixos X, Y, Z.

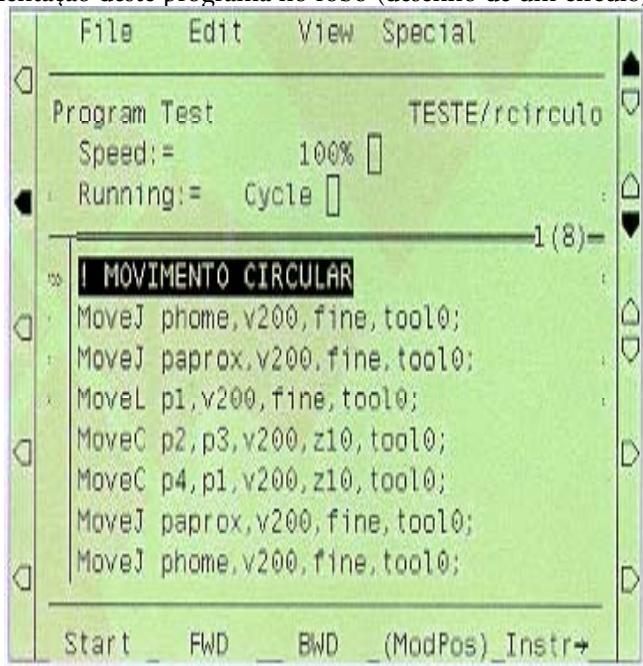
moveJ phome,v1000,z10,tool0 moveJ paprox1,v1000,z10,tool0 moveL p1,v500,fine,tool0 moveL p2,v500,fine,tool0 moveL p3,v500,fine,tool0 moveL p4,v500,fine,tool0 moveL p1,v500,fine,tool0 moveJ paprox1,v1000,z10,tool0	moveJ phome,v200,fine,tool0 moveJ paprox,v1000,fine,tool0 moveL p1,v200,fine,tool0 moveC p2,p3,v200,z10,tool0 moveC p4,p1,v200,z10,tool0 moveJ paprox,v200,fine,tool0 moveJ phome,v200,fine,tool0
---	--

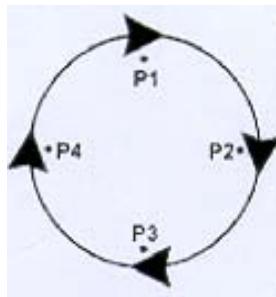
a) Desenho de um quadrado.

b) Desenho de um círculo.

Figura 8.15: Exemplos de Programas – Robô ABBTM.

Na figura 8.16 é apresentado um exemplo de implementação deste programa no robô (desenho de um círculo).





P1: Ponto inicial

**P2: Ponto de passagem
(ponto intermediário)**

P3: Ponto destino

P4: Ponto de passagem (ponto intermediário)
**P1: Ponto destino
(ponto final)**

Figura 8.16: Implementação de Programa – Robô ABBTM.

Para ilustrar os comandos básicos apresentados anteriormente (moveJ, moveL, moveC), na figura 8.17 é mostrado a implementação de um programa completo para desenho da bandeira brasileira para o Robô ABBTM.

```
moveJ phome,v1000,z10,tool0
moveJ paprox1,v1000,z10,tool0
moveL p1,v500,fine,tool0
moveL p2,v500,fine,tool0
moveL p3,v500,fine,tool0
moveL p4,v500,fine,tool0
moveL p1,v500,fine,tool0
moveJ paprox1,v1000,z10,tool0
moveJ paprox2,v1000,z10,tool0
moveL p5,v500,fine,tool0
moveL p6,v500,fine,tool0
moveL p7,v500,fine,tool0
moveL p8,v500,fine,tool0
moveL p5,v500,fine,tool0
moveJ paprox2,v1000,z10,tool0
moveJ paprox3,v1000,z10,tool0
moveL p9,v500,fine,tool0
moveC p10,p11,v500,z10,tool0
moveC p12,p9,v500,z10,tool0
moveJ paprox3,v1000,z10,tool0
moveJ phome,v1000,z10,tool0
```

Figura 8.17: Exemplo de Programação –
Desenho da bandeira brasileira – Robô ABB™.

A maioria dos robôs industriais disponíveis no mercado vêm adotando linguagens com características semelhantes, com pequenas mudanças de sintaxe e forma de estruturação, de modo que os exemplos apresentados neste tópico poderão ser facilmente traduzidos para qualquer robô industrial que apresente configurações cinemáticas semelhantes ao robô utilizado - IRB 140 da ABBTM (robô 6R).

8.5.5 – Programação Estruturada de Robô Industrial

Para programação estruturada de um robô o usuário deverá ter conhecimento das estruturas de programação apresentadas a seguir:

- **MAIN ROTINE:** Todo o programa deverá ter uma rotina principal;
- **IF – THEN:** Rotinas para tomada de decisões em função de condições lógicas.

Sintaxe:

IF condição THEN

Instrução, Programa, Condição Lógica, etc

ELSE

Instrução, Programa, Condição Lógica, etc.

ENDIF;

- **FOR:** É usado para repetir um dado número de vezes, uma ou diversas instruções.

Sintaxe:

FOR variável (I) FROM valor inicial TO valor final

[\STEP valor do passo] DO

Programa, Instrução, Condição Lógica, etc.

ENDFOR;

Observação:

- O valor de “I” só pode ser lido por instruções dentro do comando **FOR**.
- O argumento **STEP** determina a quantidade de incremento ou decremento.

- **TPREADNUM:** Rotinas de decisão utilizadas na leitura de um número no teach pendant, carregamento num registrador, para posterior utilização, como no caso de uso junto com o comando **FOR**, para executar um número de vezes uma rotina ou sub-rotina. A figura 8.18 apresenta um exemplo de utilização desta instrução.

```

PROC main()
    TPReadNum reg2,"Quantidade: ";
    FOR i FROM 1 TO reg2 DO
        MoveL phome,v400,z10,Bic;
        MoveL paprox1,v400,z10,Bic;
        .
        .
        .
        .
        MoveL p20,v400,fine,Bic;
    ENDFOR
    ENDPROC
    ENDMODULE

```

Figura 8.18: Exemplo de utilização da instrução TPREADNUM.

- **TPWRITE:** comando utilizado para colocar uma mensagem no início da operação com o robô.
- **WAITTIME:** comando utilizado para dar um tempo (dado em segundos) antes ou depois de executar uma determinada instrução ou uma sub-rotina.
- **TPERASE:** comando utilizado para limpar a tela do teach-pendant.

A figura 8.19 apresenta um exemplo de programação, utilizando as instruções TPWRITE, WAITTIME, TPERASE, para a execução de desenhos pela ferramenta de trabalho correspondentes a um retângulo, losângio, círculo da bandeira brasileira.

```

PROC main()
    TPWrite "Oi Operador..."; 
    WaitTime 2;
    TPErase;
    TPReadFK reg1,"Escolha o desenho:","Quadrado","Losa
    ngo","Círculo","Home",stEmpty;
    IF reg1=1 THEN
        quadrado;
    ELSE
        IF reg1=2 THEN
            losango;
        ELSE
            IF reg1=3 THEN
                círculo;
            ELSE
                MoveJ phome,v400,z50,Bic;
            ENDIF
        ENDIF
    ENDIF
    ENDPROC
ENDMODULE

```

Figura 8.19: Exemplo de programação: Bandeira Brasileira.

Por outro lado, na figura 8.20 esse programa é apresentado sob a forma de uma rotina principal contendo três subrotinas: retângulo, losângio, círculo.

```

PROC main()
  TPWrite "Oi Operador...";
  WaitTime 2;
  TPErase;
  TPWrite "O tempo foi de:"\Num:=ClkRead(clock1);
  TPReadFK reg1,"Escolha o desenho:","Quadrado",""
  "Losango","Círculo","Home",stEmpty;
  ClkReset clock1;
  ClkStart clock1;
  IF reg1=1 THEN
    quadrado;
  ELSE
    IF reg1=2 THEN
      losango;
    ELSE
      IF reg1=3 THEN
        círculo;
      ELSE
        MoveJ phome,v400,z50,Bic;
      ENDIF
    ENDIF
  ENDIF
  ClkStop clock1;
ENDPROC
ENDMODULE

```

Figura 8.20: Exemplo de programação: Bandeira brasileira –
Sub-rotinas.

8.6 – Programação off-line de Robôs Industriais

Na programação online é realizado todo processo de aprendizagem de tarefas de um robô, a partir da apreensão e posicionamento de objetos (peças, dispositivos de fixação e posicionamento, robôs, periféricos, etc...) em torno da própria produção dentro da célula de trabalho. O robô e sistema asso-

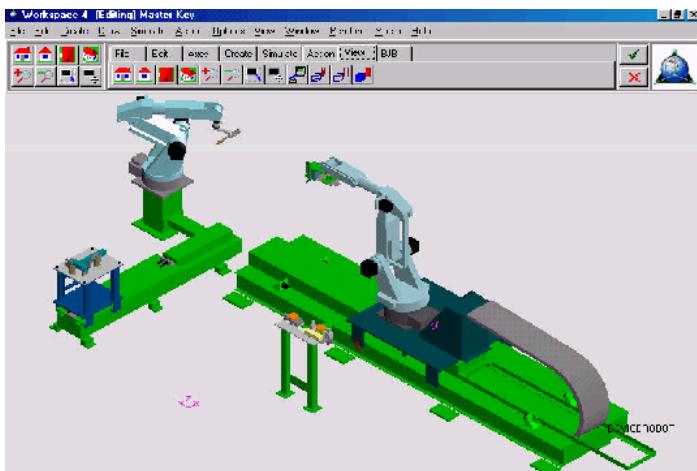
ciado (célula de trabalho) são integrados e programados com o auxílio de um “*Teach-in-Box*”.

Uma das principais vantagens da programação offline é a não necessidade de ter disponíveis os equipamentos para desenvolvimento do programa, necessitando numa fase inicial apenas do desenho da planta e especificações funcionais e tecnológicas dos elementos da célula, depois da geração de movimentos do manipulador e posteriormente com a lógica de programação e cálculos. A figura 8.21 exemplifica a implementação de um projeto de uma célula robotizada utilizando programação off-line.

Isto possibilitará um planejamento de tarefas, ambientes e ferramentas, não necessitando a parada dos equipamentos durante a implantação de novos programas em células existentes, e no caso de implementação de novas células, toda a programação e estudo de viabilidade poderão ser realizados sem investimento nestes equipamentos a partir da alteração de documentação inicialmente estabelecida, considerando inclusive tempos de execução, permitindo assim a orientação de clientes à tomada de decisões em cima de uma solução robotizada.



a) Célula Real



b) Célula Virtual (programação off-line).

Figura 8.21: Programação Off-line – SENAI

Mecatrônica (São Caetano).

A programação de robôs on-line consiste na programação do sistema em loco com acesso ao robô e equipamentos, dispositivos de fixação e peças, enquanto na programação off-line é necessário o modelo do robô num ambiente de trabalho para a programação do mesmo. Este tipo de programação apresenta muitas vantagens, entre elas:

- Possibilidade de estruturação na programação e num único ambiente de simulação possuir uma grande variedade de robôs, com possibilidade de programação sem a necessidade de se conhecer a sintaxe da linguagem de programação dos mesmos;
- Escolha da melhor solução em termos de espaço de trabalho, especificação de equipamentos e ferramentas, levando-se em consideração custos e desempenho;
- Possibilidade de escolha de parâmetros de soldagem e pintura;
- Possibilidade de simulação em computador com visualização de operação e sensores, criação de AVI's, viabilizando a implementação do projeto;
- Elaboração de especificações detalhadas sob a forma de documentação;
- Possibilidade de integração com sistemas CAD/CAM existentes;
- Redução do ciclo de paradas numa célula robotizada e Planejamento de Produção;
- Remoção do programador em áreas de riscos e meios potencialmente perigosos;
- Simplificação de atividades complexas e possibilidade de verificação de colisões (figura 8.22);
- Projeto e otimização de disposições *lay-outs* e produtos;
- Formação e Educação em Robótica.

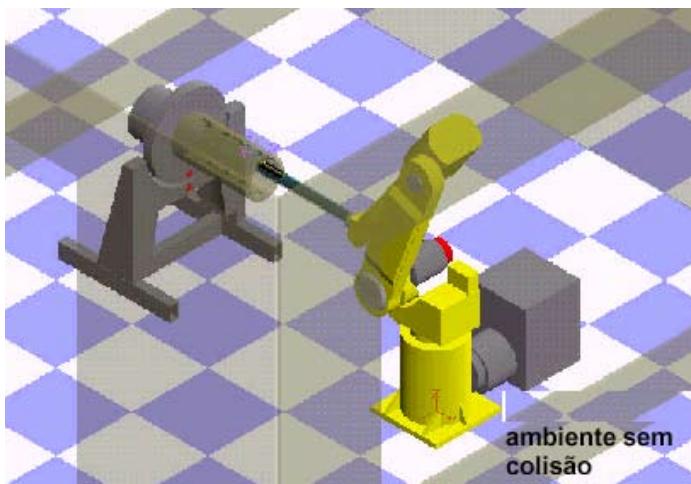


Figura 8.22: Programação Off-line – Tratamento de Colisões.

A utilização de ferramentas para programação off-line, permite a depuração de programas e a utilização de ferramentas suportes para o processo de programação, por exemplo, otimização do processo de soldagem, montagem, pintura e etc... Muitas aplicações de robô envolvem processo de produção em massa, como por exemplo, soldagem em linhas automotivas, onde o tempo de reprogramação exigido é ausente ou mínimo.

Para a aplicação de robôs industriais no campo de pequenos e médios grupos de produção, onde o tempo de programação possa ser relevante, um sistema de programação off-line torna-se essencial e imprescindível, levando-se em conta aspectos relacionados ao custo-benefício. Dentro desse contexto, a utilização de robôs vêm crescendo e o aumento da complexidade das aplicações industriais, principalmente nas operações de montagem, soldagem, pintura e paletização, torna a programação off-line mais atrativa.

De uma forma mais abrangente, podemos alternar os métodos de programação em on-line e off-line, conforme a aplicação. Esse variante de programação é denominado de “Programação Híbrida”, que E.Trostman descreve que um programa de robô consiste principalmente de duas partes: Localização (posição e orientação) e Lógica de programação (estrutura de controle, comunicação, cálculos).

A programação lógica pode ser desenvolvida off-line com a efetiva facilidade de depuração e simulação. A principal parte de comandos de movimentos pode ser implementada no modo off-line pela reutilização dos dados de CAD (desenhos eletrônicos) e pela interação do programador. Comandos de movimentos para localização e colocação de peças dentro de células robótizadas podem ser programados on-line se for necessário. Dentro desta ótica ambos os métodos podem ser vantajosos e utilizados propiciando uma melhora de flexibilidade na produção. Após a verificação do programa do robô na simulação o programa é descarregado na unidade de controle do robô (processo tratado como “download”). A correta combinação de programação on-line e programação off-line levam principalmente uma redução de custo dentro dos ajustes de produção.

8.7 - Conclusão

A programação das tarefas de robôs industriais envolve a integração de três fatores básicos:

1. A coordenada do ponto de movimento deve ser identificada e armazenada na unidade de controle. O ponto pode ser armazenado como coordenada de eixo junta individual ou coordenada geométrica do tipo que o robô possa armazenar.
2. As funções serão executadas nos pontos específicos, devem ser identificadas e gravadas.
3. Os dados dos pontos e funções são organizados dentro de um caminho de sequência lógica e subsequências. Isto inclui estabelecer que caminhos deveriam ser feitos sobre condições específicas e quando várias supervisões de estados serão feitas.

A utilização da programação off-line permite a capacitação do profissional para desenvolver a potencialidade de trabalho no software, investimento de aquisição da ferramenta e em seguida pela credibilidade de uso na integração de processos, sendo já disseminados esses conceitos sob a forma de treinamento e adequação ao contexto, somado a esforços dos fabricantes de robô que importam soluções de seus países de origem, para atendimento dos clientes.

8.8 - Referências Bibliográficas

Nof, S. Y: “Handbook of Industrial Robotics”, 2^a Ed. New York, John Wiley and Sons, 1991.

Romano, V. et al: “Robótica Industrial – Aplicação na Indústria de Manufatura e Processos”, Editora Edgard Blücher Ltda, 2002;

Sugimoro, N.: “Como usar de maneira correta os robôs industriais”, São Caetano do Sul, Escola SENAI Armando de Arruda Pereira, 2001.

CAPÍTULO 9

Programação Off-line de Robôs



CAPÍTULO 9

Programação Off-line de Robôs

Neste capítulo são abordados aspectos concernentes à programação off-line de robôs, através da apresentação dos requisitos básicos necessários para a programação estruturada e simulação, conceitos básicos de modelagem cinemática, enfatizando o problema de orientação da ferramenta, sistemas de coordenadas, procedimentos de inicialização e calibração da ferramenta terminal, direcionando o enfoque para a concepção de sistemas automatizados dentro da engenharia, baseada em modelos virtuais, propiciando análises e decisões mais próximas do real.

9.1 - Introdução

Normalmente todos os fabricantes de robôs industriais possuem os seus sistemas específicos de programação off-line, otimizando os modelos desenvolvidos para os equipamentos, de forma que propicie a sua utilização em sistemas baseados em plataforma “PC”. A representação gráfica propicia uma interação mais agradável e real no ambiente virtual, alinhado com as atuais capacidades de processamento (há mais variedade relativamente grande de possíveis configurações nos micro-computadores - PC's que acabam gerando variações significativas em alguns estudos de casos e aplicações), que torna o processo muito realístico.

Somando a possibilidade de controle dos robôs virtuais, tem-se os recursos de CAD, para a modelagem do ambiente virtual, criando a representação gráfica de peças e dispositivos, possibilitando, por exemplo, a interação completa de robôs dentro da célula de manufatura robotizada. O somatório dessas características permite executar um estudo de planejamento de layout, planejamento de movimentos do robô, equipamentos, dispositivos, considerando possibilidades de segurança e detecção de colisões, etc...

Consequentemente, a solução é dispormos de um pacote de software de simulação “aberto”, “universal”, onde podemos fazer uso de qualquer modelo de robô de diferentes fabricantes. Dos pacotes de software de simulação e programação off-line “universal” disponível comercialmente, existem alguns de uso consagrados nos mercados norte americano, canadense, europeu e asiático. Algumas vantagens que podem ser citadas no uso destes pacotes estão na possibilidade de criarmos modelos geométricos e cinemáticos, tanto direto como inverso baseado na convenção D-H, implementando nos dispositivo a serem modelados os parâmetros cinemáticos de D-H associado ao robô em estudo (comprimento e distância entre “links”, variações angulares do deslocamento das juntas de rotação, velocidade, aceleração, etc...), geralmente disponíveis nos catálogos dos fabricantes de robôs.

Este erro de desvio é facilmente compreendido, já que o modelo cinemático inverso da unidade de controle do robô e os algoritmos de controle da trajetória são métodos numéricos proprietários e os modelos utilizados pertencem ao fabricante do software “universal”. O que é visualizado na tela do computador, graficamente, não exprime a trajetória real de robô industrial, lembrando que a posição dos pontos ensinados são os mesmos. Isto, se o modelo gráfico está calibrado de acordo com o modelo real. Normalmente este tipo de erro de desvio já é minimizado e acontece em torno de milímetros e abaixo de 3% para o tempo de ciclo.

Para conseguir essa redução de erro de trajeto na programação off-line é necessário utilizar um modelo específico, usando o modelo do software “proprietário” ou usando uma interface com modelo conhecido como RRS (Realistic Robot Simulation – versão atual 1.3 – dezembro de 2002), interface desenvolvida pela necessidade dos usuários de robótica, com a finalidade de reduzir os desvios de imprecisão em relação às posições e ao tempo de ciclo.

9.2 – Softwares de Simulação e Programação Off-line

Nos últimos anos vivemos um contexto que requer uma quebra dos paradigmas da engenharia tradicional, como país que se caracteriza de forma global em usuário de tecnologia, salvo algumas áreas de excelência. Nos processos industriais de fabricação, nas indústrias montadoras e, mais ainda, nos fabricantes de autopeças responsáveis pelos fornecimentos às montadoras esse contexto é crítico.

Com a necessidade de atender a nova, não tão nova assim, demanda industrial mundial que forneça qualidade, produtividade e preço, precisamos popularizar as tecnologias de automação, não só como usuários, mas como desenvolvedores. e isso significa passar por todos os estágios do processo, da formação de profissionais, concepção (projeto) à fabricação.

A simulação propicia esta quebra de paradigma, onde a visualização não é tradicional no mundo da engenharia. Os problemas que são comuns em algumas situações no desenvolvimento de projetos são muito mais fáceis de serem identificadas pela visualização do que pela solução analítica. Ao mesmo tempo, isto representa uma redução significativa de custos na aquisição de equipamentos necessários para a montagem de células automatizadas.

Os processos industriais estão sofrendo uma revolução tecnológica, uma nova era da engenharia é um desafio que devemos encarar com seriedade, para que não se transforme numa questão de sobrevivência.

Muitos dos trabalhos encontrados relatam a pesquisa e desenvolvimento robótico. Alguns destes trabalhos são distribuídos com controle de robôs industriais, o que possibilita a automatização de sistemas robóticos e melhoria de seus alcances de aplicações nos processos de manufaturas (Craig, 1989), (Groover, 1987), (Gong, 1998), (Fu & González, 1998), dentre outros.

“lay out”, atualmente as companhias industriais estão usando estações de trabalho (Workstation) de ultima geração que possui capacidade de processamento superior aos computadores Pessoais (PC), possibilitando um maior grau de elaboração e fidelidade, modelo gráfico mais detalhado em condições reais. Existem alguns pacotes de softwares de simulação disponíveis comercialmente com diferentes capacidades e preços.

Algumas pesquisas foram realizadas em sistemas de simulação de eventos discretos para planejamento e propósitos de projetos. A motivação para se usar simulação é que ela pode oferecer captura e descrição de interações complexas com um sistema flexível de manufatura “FMS” em particular, cujos métodos analíticos provavelmente falham. Simulação é utilizada, novamente, para “insights” em sistemas de produção.

Fabricantes	Software	Endereço
Tecnomatix Technologies ltd	RobCad®	http://www.tecnomatix.com/
Dessault Delmia	Igrip®	http://www.delmia.com/
Flow software Technologies	Workspace®	http://www.Workspace5.com/
Easy-Rob	Easy-Rob®	http://www.easy-rob.com/

Tabela 9.1: Exemplos de softwares comerciais disponíveis no mercado para simulação e programação off-line de robôs.

Fabricantes	Software	Endereço
ABB	RobotStudio®	http://www.abb.com/
Motoman	MotoSim®	http://www.yaskawa.com/
Reis	ProSim®	http://www.reis.com/
Fanuc	SimPro®	http://www.fanuc.com/

Tabela 9.2: Exemplos de softwares de simulação e programação off-line desenvolvidos por fabricantes de robôs industriais.

9.3 – Sistemas de Coordenadas e Modelagem Cinemática

O conhecimento desses métodos e ferramentas de programação off-line é de fundamental importância aos usuários de robô facilitando e permitindo a obtenção de posições e orientações precisas, melhorando, assim, a fidelidade do modelo.

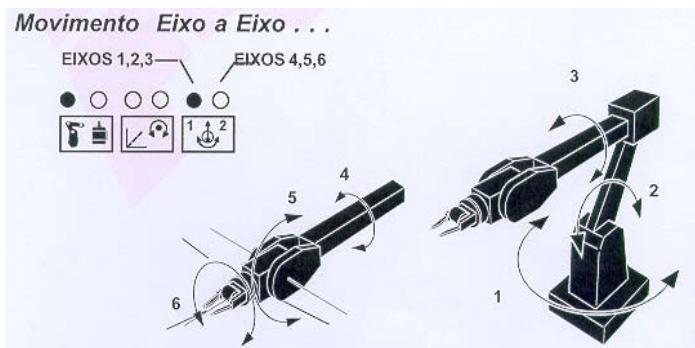
Atualmente percebe-se uma necessidade latente, principalmente em relação às montadoras automobilísticas, da utilização de soluções por intermédio de simulação e programação off-line e visualização da posição e orientação da ferramenta terminal de um robô, em função de diversos ganhos que podem ser agregados nos processos e a crescente utilização de robôs industriais.

O método de programação mais frequentemente utilizado em robôs industrial é a aprendizagem ponto-a-ponto

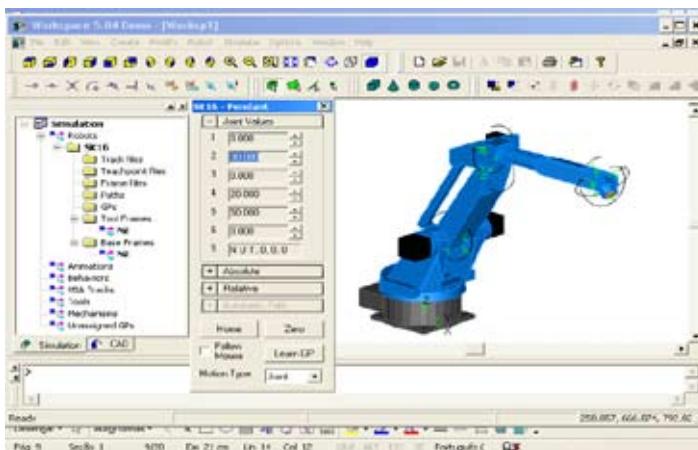
(PTP) onde as juntas de um robô são movimentadas individualmente até as posições desejadas, através de um processo de aprendizagem de tarefas. Normalmente, esta programação pode ser realizada através da movimentação angular das juntas, movimentação cartesiana da ferramenta e reorientação da ferramenta, conforme é apresentado a seguir.

9.3.1 - Movimentação Angular das Juntas

Neste método são gravados pontos de referência fornecidos pelos transdutores de posição localizados em cada junta e a partir desses pontos são geradas trajetórias (angulares) através da utilização de algoritmos de interpolação e filtragem. A simplicidade deste método não exige o conhecimento do modelo geométrico do robô (figura 9.1).



a) Através do Teach-in box de programação do robô.



b) Através de software de Programação off-line.

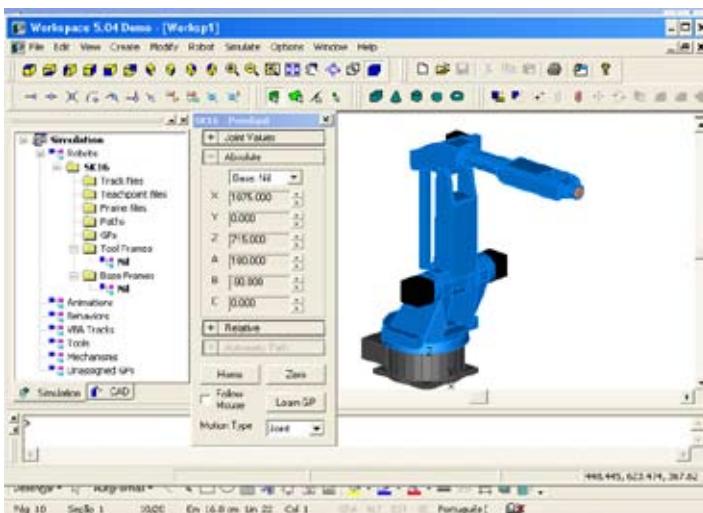
Figura 9.1: Movimento Angular das Juntas.

9.3.2 - Movimento na Direção Cartesiana

O procedimento para movimentação na direção cartesiana é idêntico ao descrito anteriormente, mas a obtenção dos pontos de referência é realizada utilizando o modelo cinemático do manipulador, possibilitando assim a operação do robô seguindo as direções X, Y e Z (figura 9.2).



a) Através do *Teach-in box* de Programação do Robô.



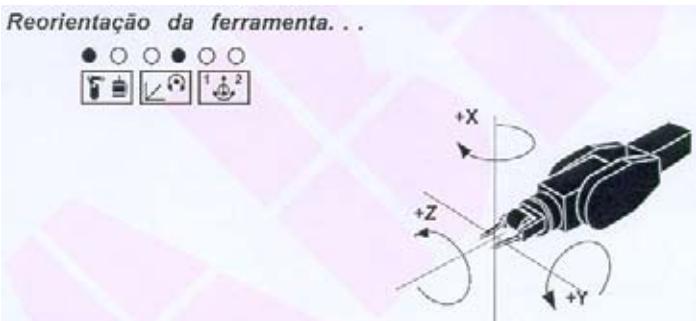
b) Através de Software de Programação Off-line.

Figura 9.2: Movimento em Coordenadas Retangulares.

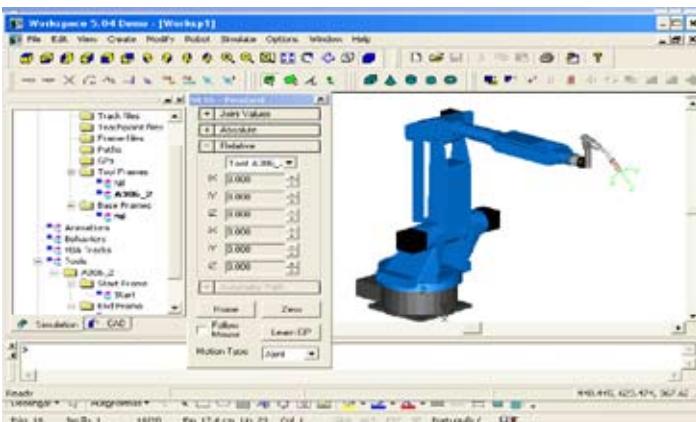
9.3.3 Movimento de Reorientação da Ferramenta

O procedimento para o movimento de reorientação da ferramenta é idêntico ao descrito anteriormente, mas a obtenção dos pontos de referência é realizada utilizando a orientação da ferramenta em torno da rotação dos eixos X, Y, Z, correspondendo as três rotações *Roll*, *Pitch*, *Yaw* (figura 9.3). A ferramenta terminal de um robô é designada de TCP (Tool Center Point) ou ponto central da ferramenta.

Reorientação da ferramenta...



a) Através do *Teach-in box* de programação do Robô.



b) Através de software de Programação off-line.

Figura 9.3: Reorientação da Ferramenta.

9.4 - Programação através de Visualização Gráfica

A utilização de técnicas de desenho auxiliado por computador (CAD) tem encontrado extensivo uso na recolocação e melhora de processos de desenhos de engenharia, desenhos de arquitetura e muitas outras aplicações. Contudo, um processo em engenharia envolvendo partes em movimento só pode ser totalmente entendido através do processo de simulação.

Técnicas iniciais foram envolvidas utilizando um desenho de engenharia (CAD) de uma parte mecânica e criando uma simulação animada de movimentos que as máquinas de usinagem “CNC” devem seguir para criar a peça a partir de uma linha de blocos. Também melhorando a visualização de processos, um arquivo pode ser criado contendo as instruções requeridas para as maquinas CNC. O arquivo pode então ser executado para criar a peça.

Essa extensão do CAD para o CAM aparentemente é desejável para robôs industrial. Contudo, a cinemática envolvida nos movimentos dos robôs é consideravelmente mais complexa do que o relacionado com as máquinas de usinagem CNC, e a relação entre as **varreduras das curvas** pela ferramenta de final de braço do robô e as variáveis das juntas não é uma cinemática direta. Adicionalmente, para uma simulação de robô ser de uso geral, deve ser capaz de simular uma grande variedade de tipos de robôs e configurações.

Apesar desta dificuldade, a simulação aparece para prover uma conveniente interação do ambiente gráfico para melhorar a facilidade com a qual os robôs industriais são programados. Benefícios, tal como a habilidade para detectar colisões off-line entre robôs e objetos, e a habilidade para evoluir e otimizar o tempo gasto para uma sequência de movimentos off-line têm também proporcionado maior incentivo para pesquisas e desenvolvimento dentro de simulação robótica.

9. 5 - Exemplos de Aplicações

A figura 9.4 apresenta um exemplo simulado graficamente e programado off-line, que possibilita um trabalho de estruturação da linguagem trabalhada e uma explanação de um ambiente completo de programação (figura 9.5).

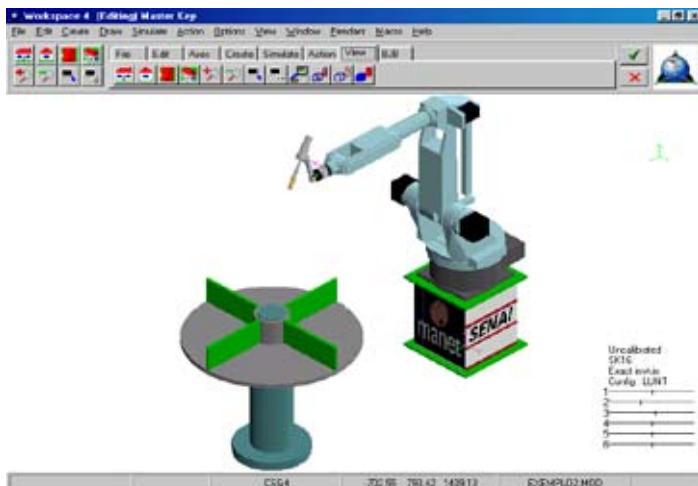


Figura 9.4: Exemplo de uma célula de trabalho com solda robotizada

```
PROGRAM
-- ! LANGUAGE KAREL 2
-- ! MEMORY 8192
-- ! ROBOT SK16
-- TEACHPOINT DECLARATIONS
VAR
TP1 : POSITION
TP2 : POSITION
TP3 : POSITION
TP4 : POSITION
TP5 : POSITION
TP6 : POSITION

BEGIN
$UTOOL=POS(-11.5265,2.7334,307.2149,0,45,-160,")
$USEMAXACCEL=TRUE
%INCLUDE TEACH POINTS#
...
END PROGRAM
```

Figura 9.5: Programa implementado em Linguagem Karel 2
do Controlador FANUC.

Para exemplificar a diferença de um programa off-line de um programa off-line estruturado será utilizada uma situação em que um robô execute um processo de soldagem em uma peça de trabalho que tem como característica quatro posições simétricas a serem soldadas, ilustrado nas figuras 9.6 à 9.9.

```
WITH $MOTYPE=LINEAR
    MOVE TO TP1
    WITH $MOTYPE=LINEAR
        MOVE TO TP2
    -- ! ARCWELDON 250, 25
    WITH $MOTYPE=LINEAR
        MOVE TO TP3
    WITH $MOTYPE=LINEAR
        MOVE TO TP4
    WITH $MOTYPE=LINEAR
        MOVE TO TP5
! ARCWELDOFF
    WITH $MOTYPE=LINEAR
        MOVE TO TP6
```

Bloco de deslocamento e execução da
1^a soldagem, executando a
aproximação, soldagem e afastamento
da primeira divisória.

Figura 9.6: Bloco de deslocamento e a 1^a soldagem.

```
-- ! TURNOBJ 'MESA',0.00,0.00,1.00,0.00,-  
0.00,792.00,90.00,50  
WITH $MOTYPE=LINEAR  
MOVE TO TP1  
WITH $MOTYPE=LINEAR  
MOVE TO TP2  
-- ! ARCWELDON 250,25  
WITH $MOTYPE=LINEAR  
MOVE TO TP3  
WITH $MOTYPE=LINEAR  
MOVE TO TP4  
WITH $MOTYPE=LINEAR  
MOVE TO TP5  
-- ! ARCWELDOFF  
WITH $MOTYPE=LINEAR  
MOVE TO TP6
```

Bloco de deslocamento e execução da 2^a soldagem, executando a aproximação, soldagem e afastamento da segunda divisória.

Figura 9.7: Bloco de deslocamento e a 2^a soldagem.

```

-- ! TURNOBJ'MESA',0.00,0.00,1.00,0.00,-0.00,792.00,90.00,50
WITH $MOTYPE=LINEAR
MOVE TO TP1
WITH $MOTYPE=LINEAR
MOVE TO TP2
-- ! ARCWELDON 250,25
WITH $MOTYPE=LINEAR
MOVE TO TP3
WITH $MOTYPE=LINEAR
MOVE TO TP4
WITH $MOTYPE=LINEAR
MOVE TO TP5
-- ! ARCWELDOFF
WITH $MOTYPE=LINEAR
MOVE TO TP6

```

Instrução para 3^a rotação do dispositivo de posicionamento

Bloco de deslocamento e execução da 4^a soldagem, executando a aproximação, soldagem e afastamento da quarta divisória.

Figura 9.8: Bloco de deslocamento e a 3^a soldagem.

```

-- ! TURNOBJ'MESA',0.00,0.00,1.00,0.00,-0.00,792.00,90.00,50
WITH $MOTYPE=LINEAR
MOVE TO TP1
WITH $MOTYPE=LINEAR
MOVE TO TP2
-- ! ARCWELDON 250,25
WITH $MOTYPE=LINEAR
MOVE TO TP3
WITH $MOTYPE=LINEAR
MOVE TO TP4
WITH $MOTYPE=LINEAR
MOVE TO TP5
-- ! ARCWELDOFF
WITH $MOTYPE=LINEAR
MOVE TO TP6

```

Instrução para 3^a rotação do dispositivo de posicionamento

Bloco de deslocamento e execução da 4^a soldagem, executando a aproximação, soldagem e afastamento da quarta divisória.

Figura 9.9: Bloco de deslocamento e a 4^a soldagem.

Nos quatro cordões de solda realizados no processo o trajeto é o mesmo e é representado pelos pontos TP1; TP2; TP3; TP4; TP5 e TP6, que foram ensinados no robô a instrução --! TURNOBJ ‘MESA’,..., , exclusiva do software de simulação, possibilitando a rotação do dispositivo de posicionamento “MESA”, para cada uma das posições desejada, levando a completar e executar o processo de cada uma das partes discretas.

```
-- WORKSPACE teachpoint file
TP1 = POS(847.2979,226.6498,-171.5,152.7637,41.641
1,161.1183,’LUFB’)
TP2 = POS(800,0,-2611.5,-
148.5703,17.462,169.8249,’LUFB’)
TP3 = POS(1000,10,-263,-
148.5703,17.462,169.8249,’LUFB’)
TP4 = POS(1199.95,10,-263,-
148.5703,17.462,169.8249,’LUFB’)
TP5 = POS(1200.1093,10,-1011.3819,-
171.2326,22.7317,168.6665,’LUFB’)
TP6 = POS(914.8239,-84.3288,18.3028,152.7637,41.64
11,161.1183,’LUFB’)
```

Figura 9.10: Coordenadas de Posição e Orientação
dos seis pontos ensinados (TP).

9.5.1 – Comentários da Implementação

O programa descrito na figura 9.5 foi implementado na linguagem KAREL 2, utilizado nas unidades de controle dos robôs industrial FANUC.

O robô utilizado na simulação é um modelo SK 16 do fabricante Yaskawa/Motoman, reforçando a ideia da flexibilidade na programação off-line, onde a linguagem independe do robô a ser simulado.

Os caracteres -- ! representam instruções exclusiva do software de simulação. Quando transferido para uma unidade de controle do robô, é entendido como um simples comentário. Dessa forma o software pode utilizar funções de controle, comandos, etc., exclusivos de sua sintaxe. O exemplo acima, cujas funções ARCWELDON e ARCWELDOFF ligam e desligam as ações de soldagem, é interpretado com um comentário em qualquer unidade de controle de robôs.

A instrução TURNOBJ rotaciona o objeto desejado em uma posição também controlada. Na implementação real é necessário trabalhar com sinais de I/O, para enviar a necessidade de entendimento da mudança de posição no momento desejado.

Apesar de ter sido realizada uma programação off-line, não houve uma preocupação em estruturar o programa, utilizando os recursos de estruturação que a linguagem dispõe. O processo de estruturação pode ser realizado textualmente, já que os pontos e orientações necessários ao processo, demonstrado na figura 9.10, já foram definidos através da programação ponto-a-ponto virtual, baseando-se nos modelos gráficos. A linguagem permite a utilização das funções, FOR... ENDFOR, REPEAT ... UNTIL, WHILE ... ENDWHILE. Estes comandos permitem de forma distintas o mesmo resultado, já que o processo programado de soldagem utiliza o mesmo bloco de movimentos do robô, alterando somente a posição de soldagem através do dispositivo de posicionamento, uma instrução de “looping” atende de forma satisfatória a repetição do movimento do robô em um numero conhecido de quatro repetições. Ao implementar as mudanças propostas teremos algumas variantes de programação interessantes de serem analisadas, conforme exemplos apresentados a seguir.

9.5.2 – Outros Exemplos

- Instrução For ... End-For (figura 9.11)
- Instrução Repeat... Until (figura 9.12)

- Instrução While... EndWhile (figura 9.13)
- Exemplos Completos de Simulação (figura 9.14 e 9.15)

```

VAR
    TP1 : POSITION
    TP2 : POSITION
    TP3 : POSITION
    TP4 : POSITION
    TP5 : POSITION
    TP6 : POSITION
VAR
    I: INTEGER      Declaração de variável
BEGIN
    I=0            Condição inicial de valor na variável
    $UTOOL=POS(-11.5265,2.7334,307.2149,0,45,-160,")
    $USEMAXACCEL=TRUE
    %INCLUDE TP#
    FOR I=0 TO 3 DO  Condição para execução do programa
        WITH $MOTYPE=JOINT
        MOVE TO TP1
        WITH $MOTYPE=LINEAR
        MOVE TO TP2
        -- ! ARCWELDON 300,3
        WITH $MOTYPE=LINEAR
        MOVE TO TP3
        WITH $MOTYPE=LINEAR
        MOVE TO TP4
        WITH $MOTYPE=LINEAR
        MOVE TO TP5
        -- ! ARCWELDOFF
        WITH $MOTYPE=LINEAR
        MOVE TO TP6
        -- ! TURNOBJ 'MESA',0.00,0.00,1.00,0.00,0.00,792.00,90
        .00,30
        I = I +1  Lógica para implementar a contagem para
                  execução
    ENDFOR

```

Figura 9.11: Utilização da Instrução For... EndFor

```

VAR
  TP1 : POSITION
  TP2 : POSITION
  TP3 : POSITION
  TP4 : POSITION
  TP5 : POSITION
  TP6 : POSITION
VAR
I: BOOLEAN Declaração de variável utilizada para
a contagem
BEGIN
  $UTOOL=POS(-11.5265,2.7334,307.2149,0,45,-
160,")
  $USEMAXACCEL=TRUE
  %INCLUDE TP#
  REPEAT           Instrução a ser executada
  WITH $MOTYPE=JOINT
  MOVE TO TP1
  WITH $MOTYPE=LINEAR
  MOVE TO TP2
  -- ! ARCWELDON 300,30
  WITH $MOTYPE=LINEAR
  MOVE TO TP3
  WITH $MOTYPE=LINEAR
  MOVE TO TP4
  WITH $MOTYPE=LINEAR
  MOVE TO TP5
  -- ! ARCWELDOFF
  WITH $MOTYPE=LINEAR
  MOVE TO TP6
  -- ! TURNOBJ 'CSG3',0.00,0.00,1.00,0.00,0.00,0.792
  .00,90.00,30
  UNTIL FALSE      Condição para execução

```

Figura 9.12: Utilização da Instrução Repeat... Until.

```

VAR
    TP1 : POSITION
    TP2 : POSITION
    TP3 : POSITION
    TP4 : POSITION
    TP5 : POSITION
    TP6 : POSITION
VAR
I : INTEGER Declaração de variável utilizada para a
contagem
BEGIN
    $UTOOL=POS(-11.5265,2.7334,307.2149,0,45,-
160,")
    $USEMAXACCEL=TRUE
    %INCLUDE TP#
    WITH $MOTYPE=JOINT
    MOVE TO TP1
WHILE I<4 DO Condição a ser executada, em função
do valor armazenado na variável.
    WITH $MOTYPE=LINEAR
    MOVE TO TP2
    -- ! ARCWELDON 300,30
    WITH $MOTYPE=LINEAR
    MOVE TO TP3
    WITH $MOTYPE=LINEAR
    MOVE TO TP4
    WITH $MOTYPE=LINEAR
    MOVE TO TP5
    -- ! ARCWELDOFF
    WITH $MOTYPE=LINEAR
    MOVE TO TP6
    -- ! TURNOBJ 'CSG3',0.00,0.00,1.00,0.00,0.00,792.
00,90.00,30
I = I+1 Variável utilizada como contador
ENDWHILE

```

Figura 9.13: Utilização da Instrução While... EndWhile

Nas figuras 9.14 e 9.15, respectivamente, são exemplificados um “ambiente completo de programação”, contendo uma célula robotizada para soldagem “GMAW”, com dois robôs e um dispositivo para alimentação e saída de uma peça, após o processo de soldagem, com a possibilidade de troca de ferramenta no final de braço do robô de manuseio.

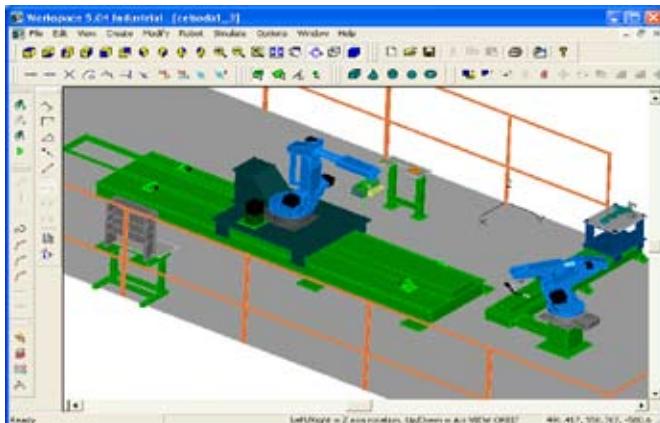


Figura 9.14: Célula de Manufatura Automatizada Robotizada.

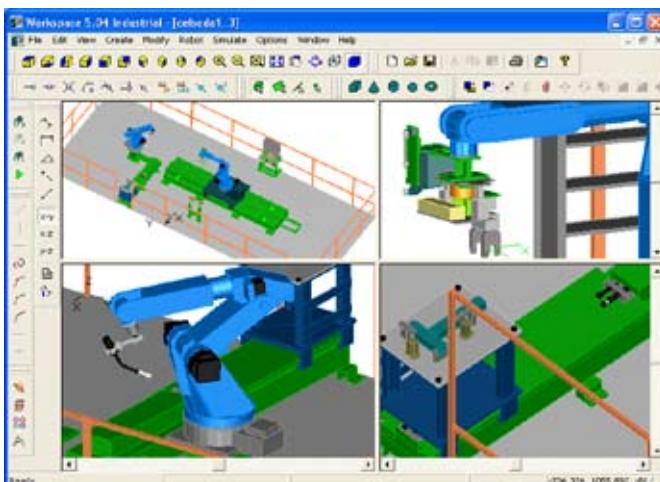


Figura 9.15: Detalhes da Célula de Manufatura Automatizada Robotizada.

Este exemplo de uma célula robotizada industrial mostra a necessidade de se trabalhar em cooperação², a unidade de controle do robô controla de forma independente 13 graus de liberdade no processo de soldagem, com possibilidade de extensão de controle de até 21 graus de liberdade.

Com diversas abordagens de programação e comunicação entre robôs e dispositivo, a complexidade da integração entre eles enriquece o estudo e a possibilidade de realizar mudanças em todos os níveis de integração, bem como no planejamento de novas tarefas. É possível ainda objetivar novas possibilidades no trato da programação, desenvolvendo situações e arranjos limitados ao equipamento e layout preconizado no ambiente disponível. A arquitetura definida na célula é proprietária, “fechada”, definida pelo integrador e fabricante. É possível alterar a lógica de trabalho implementando novas tarefas, tornando o sistema mais flexível.

9.6 - Conclusão

Neste capítulo foram apresentados conceitos básicos de Programação off-line de Robôs, através da apresentação dos requisitos básicos necessários para a programação estruturada e simulação, conceitos básicos de modelagem cinemática, enfatizando o problema de orientação da ferramenta, e sistemas de coordenadas e procedimentos de inicialização e calibração da ferramenta terminal, direcionando o enfoque para a concepção de sistemas automatizados dentro da engenharia, baseada em modelos virtuais, propiciando análises e decisões mais próximas do real. Neste capítulo foi também apresentado um ambiente completo de programação off-line de um robô industrial que será utilizado nos próximos capítulos.

2 Cooperação, processo de trabalho entre robôs de forma que possibilite a sincronia de movimentos em tarefas simultâneas ou não. Na cooperação, a unidade de controle controla todos os robôs de forma multiplexada, através de algoritmos implementados nos modelos cinemáticos, possibilitando inclusive controlar eixos externos, dispensando a necessidade de integração de comunicação por porta serial via I/O's.

9.7 - Referências Bibliográficas

Acar, M.: Mechatronics Challenge for the Higher Education World, *IEEE Transactions on Components, Packing, and Manufacturing Technology*, vol. 20, no. 1, p. 1420, 1997.

Anton, S., et al. A framework for Realistic Robot Simulation and Visualisation Disponível em <<http://www.easy-rob.com/data/Frame-RRS.pdf>>. Acesso 29 mar. 2003

Asfahl, C. Ray. Building Blocks of Automation in: Robots and Manufacturing Automation, 2nd ed. New York: John Wiley & Sons, 1992 p 23-53.

Ashley, S.: Getting a Hold on Mechatronics, *Mechanical Engineering, ASME*, maio de 97, p. 6063, 1997.

Brunson, B. Emerging Trends in Robotic Integration. Robotics online e-news. Michigan, Nov.2001. Disponível em: <<http://www.roboticsonline.com/public/articles/archive-details.cfm?id=577>> Acesso 19 jan. 2004.

Brumson, B., Robots for small business: A Growing Trend. Robotics online e-news. Michigan, Out.2003, Disponível em: <<http://www.roboticsonline.com/public/articles/archivedetails.cfm?id=1231>>, Acesso 19 jan. 2004.

Camelot, Ropsim, Robot Off-line Programming and Simulation system for industrial robots. Disponível em: <<http://www.camelot.dk/english/offvson.htm>> Acesso 26 jun. 2003.

Campbell, J.E., Robot Motion Control: How Special is it? Robotics online e-news. Michigan, Set.2003, Disponível em: <<http://www.roboticsonline.com/public/articles/archivedetails.cfm?id=1174>>, Acesso 19 jan. 2004.

Easy-Rob,Easy-rob,The Company for 3D Robot Simulation, Disponível em: <<http://www.easy-rob.com>> Acesso 03 Set. 2003.

EF-Robotertechnik GmbH, Cosimir® Professional, Manual do usuário, 2002.

Fiedler, P. J., Schilb, C. J. **Open Architecture Robot Controllers and Workcell Integration. Robotics online e-news. Technical Papers, Michigan.** Disponível em: < http://www.roboticsonline.com/public/articles/Open_Architecture_Genesis.PDF > Acesso 17 set. 2003.

Fiedler, P. J., Dehof, M.K. Workcell Communications: Conectivity, Man-Machine interfaces, and multi system management . Robotics online e-news. Technical Papers, Michigan. Disponível em: <http://www.roboticsonline.com/public/articles/Workcell_Communications_ANS98_Genesis.PDF> Acesso 17 Set. 2003.

Flow Software Technologies, Workspace5, Robotic simulation and off-line programming software. Disponível em: <<http://www.workspace5.com> > Acesso 03 Set. 2003.

Flow Software Technologies Ltd, Workspace 5.04, Manual do usuário, 16 maio 2003.

Fu, K.S., Gonzáles,R.C., Lee,C.S.G. **Robotics: Control, Sensing, Vision, and Intelligence, McGraw-Hill International, 1987, 580p**

Groover, Mikell P. Automation in: Dorf, Richard C. (Ed), Nof, Shimon Y. International Encyclopedia of Robotics: Applications and Automation, New York: John Willey & Sons, 1988 p 136-151

Hardin, W., **Robotic Integrators take their Specialties Seriously. Robotics online e-news. Michigan, Abr.2002**

Disponível em: < <http://www.roboticsonline.com/public/articles/archivedetails.cfm?id=723> > Acesso 17 set. 2003.

Hewit, J.R., King, T.G.: **Mechatronics Design for Product Enhancement, IEE & ASME Transactions on Mechatronics, vol. 1, no. 2, p. 111119, 1996.**

Nagle, B. Your First Robot. **Robotics Industry Directory - 2002-03 Edition**, A publication of robotic Industries Association, p 8.

Nitsche, A.T., Romeiro Filho, E., **aplicação de Tecnologias de CAD/CAE/CAM em Desenvolvimento de Produtos**. In: **Encontro Nacional de Engenharia de Produção – ENEGP, XVII, 2002. Curitiba – PR.**

Nitsche, A.T., Romeiro Filho, E., Gonçalves, J. B., Gomes, D.T., Uso das Ferramentas de CAD/CAE/CAM no desenvolvimento de produtos: O Estado das Práticas em pequenas e médias empresas. *2ºCOBEF – 2003*

Robot Simulation ltd, Workspace 4.01, **Manual do usuário, 1999.**

Rosário, J.M. Modelagem e Controle de Robôs in: Romano, V.F. (Ed) Robótica Industrial: Aplicação na Indústria de Manufatura e Processos, São Paulo: Edgard Blucher Ltda, 2002 p 20-46

Roos, E., Behrens, A., Anton, S., RDS – Realistic Dynamic Simulation of Robots. Disponível em < <http://www.easy-rob.com/data/rds.pdf>>. Acesso 29 mar. 2003

Roos, E., Behrens, A. Off-line Programming of industrial robots – Adaptation of simulated user programs to the real environment. Computers in industry, 1997 33(1): p 139-150.

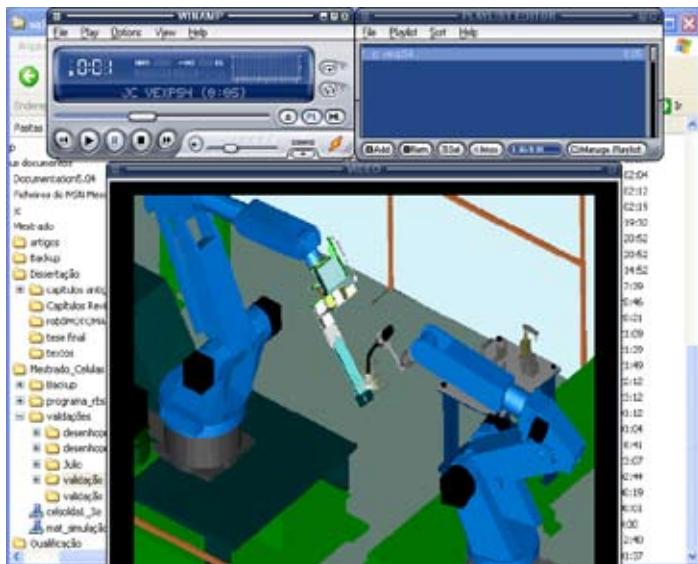
Salminen, V: Ten Years of Mechatronics Research and Industrial Applications in Finland, IEEE-ASME Transactions on Mechatronics, vol. 1, no. 2, p. 103105, 1996.

Trostman, E., Conrad, F., Trostman, S., Nielsen L.F. **Robot Off-line programming and simulation as a true CIME subsystem**. In: IEEE International Conference on Robotics and Automation. 1992

Vollmann, K. . **Realistic Robot Simulation: Multiple Instantiating of Robot Controller Software** in: **IEEE ICIT'02, International Conference On Industrial Technology, 2002** p 1194-1198.

CAPÍTULO 10

Integração de Robôs em Células de Manufatura



CAPÍTULO 10

Integração de Robôs em Células de Manufatura

Neste capítulo é apresentado inicialmente um exemplo de implementação de programa off-line para programação de tarefas de um manipulador robótico, através da implementação e bibliotecas dedicadas. A seguir são apresentados exemplos práticos de utilização de alguns softwares comerciais de programação off-line de robôs industriais.

São apresentados dois exemplos de aplicações: uma célula automatizada implementada na Faculdade SENAI de Tecnologia Mecatrônica “Armando de Arruda Pereira”, SP, e um modelo de uma célula virtual robotizada implementada no Laboratório de Automação Integrada e Robótica da Faculdade de Engenharia Mecânica da UNICAMP. Nesses dois exemplos são enfatizados aspectos concernentes aos processos de programação avançada de robôs, onde existe a possibilidade real de ganho no processo, enfocando uma utilização racional nos processos de programação e que venha a agregar de forma significativa a concepção de programação.

Finalmente, neste capítulo ainda é apresentado um estudo relativo à calibração de robôs, a partir de obtenção de parâmetros cinemáticos durante a simulação, possibilitando a realização de uma análise da aplicabilidade, cujo enfoque é a concepção e planejamento no uso da simulação e programação off-line, dificuldades e facilidades, ganhos reais diretos e indiretos.

10.1 – Implementação de Software de Programação Off-Line e Simulação

A necessidade da visualização e validação das operações de robôs antes delas serem realizadas faz da programação “off-line” uma ferramenta poderosa. Para isto é necessário que o programa seja de fácil uso e aplicação.

O programa computacional apresentado neste anexo é capaz de suportar dois tipos de programação de trajetória: “off-line” e “on-line”, para qual serão instalados interfaces A/D, D/A e digital no microcomputador que permitirá o acesso a todos os sinais (monitoramento e controle) do robô.

O objetivo da modularidade do programa está no fato que este programa pode ser usado para qualquer tipo de robô, mudando apenas as bibliotecas que contenham as dimensões e os parâmetros do manipulador em estudo. Inicialmente estará considerando as dimensões e os parâmetros do robô MANUTECH r3, apresentado no capítulo 4 deste livro. O programa, denominado Simula, foi implementado em linguagem computacional apresentando alto grau de estruturação, o que permite simplificações na programação de tarefas com grau de alta complexidade. A tela principal deste módulo é apresentada na figura 10.1.

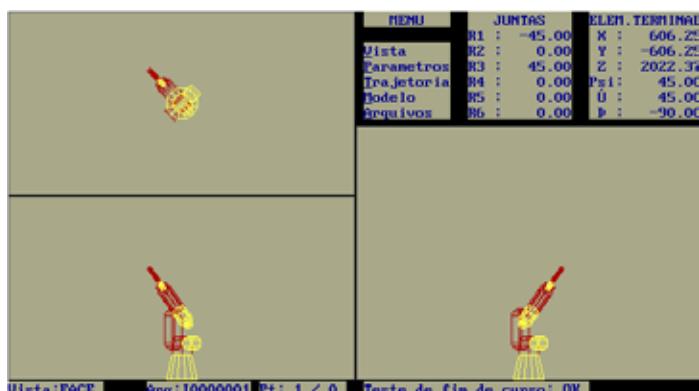


Figura 10.1: Tela principal do módulo de simulação.

A geração de trajetórias através do uso do modelo cinemático inverso utilizando o método de Gauss apresentou excelentes resultados. A partir do software desenvolvido pode-se introduzir novos módulos complementares, tais como módulos de detecção de colisões, entre outros, que poderão ser implementados em tempo real, os quais terão como principal característica a fácil utilização e/ou modificação pelo usuário.

Este software permitirá o estudo da acerácea, repetibilidade e estabilidade do manipulador ao descrever a trajetória automaticamente, e a implementação das eventuais modificações (nos sensores do manipulador e/ou no software desenvolvido) decorrentes deste estudo, possibilitando a efetiva utilização do manipulador para realizar tarefas automatizadas em diferentes ambientes de atuação.

10.1.1 - Pacote Computacional Desenvolvido

A programação “off-line” de um dispositivo robótico (manipulador ou robô) é assim denominada porque ocorre quando o dispositivo está desconectado do computador.

A definição da tarefa - e, portanto, da trajetória - a ser executada ocorre no próprio computador, a partir do modelo geométrico do robô em estudo, que aparece na tela junto com os modelos do dispositivo e das ferramentas utilizadas.

Facilidades de edição gráfica permitem a construção dos modelos, e recursos adicionais permite realizar simulação gráfica, detecção de colisões e geração de trajetórias. Ao final, a trajetória pretendida é gerada, podendo então ser enviada por interface serial ou paralela para o controlador do dispositivo, que poderá ser acionado para a execução da tarefa.

10.1.2 - Implementação de Bibliotecas

Um pacote computacional deverá ser desenvolvido de modo a atender a essas características, através da implementação de um programa modular com o desenvolvimento de bi-

bibliotecas dedicadas para a robótica. A partir dessas bibliotecas serão implementados três módulos de programação off-line: SIMULA, TRAJETÓRIA e GERAÇÃO DE OBSTÁCULOS. A figura 10.2 apresenta um esquema geral das bibliotecas, e a seguir a descrição detalhada de cada biblioteca.

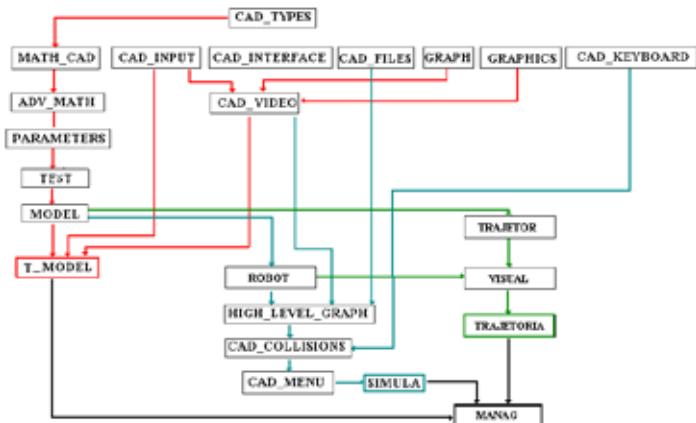


Figura 10.2: Esquema geral das bibliotecas.

10.1.3 - Descrição das Bibliotecas de um Programa Off-Line

a) Bibliotecas Básicas

- CAD_TYPES: contém a definição dos diversos tipos e variáveis utilizadas em outros pacotes.
 - CAD_INPUT: possui os recursos necessários para a entrada e saída de dados.
 - CAD_FILES: possui recursos para a abertura, leitura, alterações e fechamento de arquivos.
 - PARAMETERS: encontram-se os parâmetros do robô em estudo.
 - CAD_KEYBOARD: possui os recursos necessários para a entrada de dados pelo teclado.

b) Bibliotecas Matemáticas

- MATH_CAD: definição das diversas operações matemáticas básicas utilizadas nos pacotes.
- ADV_MATH: definição de operações matemáticas avançadas.

c) Bibliotecas Gráficas

- CAD_VIDEO: recursos básicos de tela gráfica.
- GRAPH: biblioteca gráfica básica para os outros pacotes.
- GRAPHIC: biblioteca gráfica básica para os outros pacotes.
- HIGH_LEVEL_GRAPH: rotinas gráficas para a implementação de modelo geométricos de robôs, (visualização) como, por exemplo, o manipulador Kraft.

d) Bibliotecas Modelo

- ROBOT: modelo gráfico do robô em estudo.
- MODEL: procedimentos que fazem o cálculo do modelo cinemático inverso e direto de robôs.
- T_MODEL: programa para o teste do modelo geométrico do robô em estudo.
- TEST: contém as funções para testes em geral, tais como fim das articulações.
- CAD_COLLISIONS: contém os procedimentos para o teste de colisão.

e) Bibliotecas Gerenciamento Modelo

- CAD_INTERFACE: funções para o controle das interfaces A/D e D/A e digital para acessar todos os sinais (monitoramento e controle) do robô.
- CAD_MENU: biblioteca auxiliar que gerencia as telas gráficas e as funções do programa principal SIMULA.

- TRAJETORIA: biblioteca auxiliar que executa as funções do programa principal Trajetória.
- MANAGER: gerencia os executáveis Simula, trajetória e T_model.

10.1.4 - Módulo de Simulação (SIMULA)

Este módulo possibilita a edição gráfica e interativa da trajetória, realizando a simulação e visualização do cenário completo de atuação, que contém o dispositivo robótico (manipulador ou robô), base móvel, ferramentas dedicadas, acessórios, etc. (figura 10.3). Ele apresenta as seguintes características:

- Simplicidade na utilização e modificação, com uma interface amigável com o operador;
- Modelo inverso numérico da cadeia representando o robô;
- Modelo geométrico com o uso de modelos sólidos;
- Possibilidades de testes de verificação de colisão;
- Imagem gráfica e simulação da tarefa no ambiente de trabalho.



Figura 10.3: Módulo de Simulação – Programa SIMULA.

10.1.5 - Módulo de geração de ferramentas, obstáculos e ambiente de atuação (OBSTÁCULO)

Este módulo possibilita a edição gráfica do ambiente de atuação do robô, incluindo obstáculos e ferramentas dedicadas. Estes elementos são construídos recursivamente a partir de elementos primitivos (cilindro, esfera e paralelepípedo). A figura 10.4 apresenta a tela de visualização deste programa.

10.1.6 - Módulo de geração de trajetórias (TRAJETÓRIA)

Este módulo, responsável pela geração de trajetórias, cria um arquivo de dados compatível com os protocolos de comunicação do robô em estudo para ser enviado para o programa SIMULA. A figura 10.5 apresenta uma tela correspondente aos resultados (a) e descrição da evolução angular das juntas (sinais de referência do controlador de posição) (b).

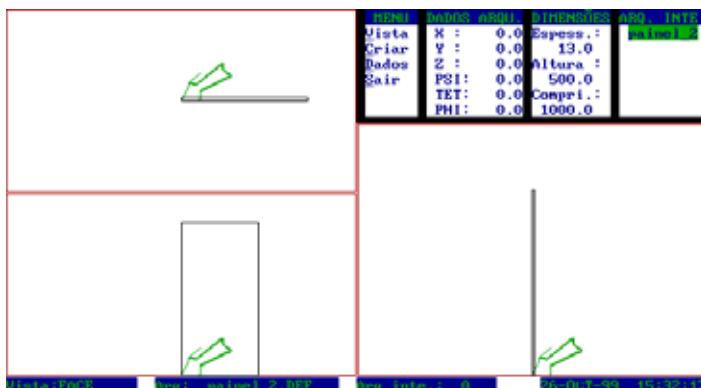


Figura 10.4: Módulo de criação de ambientes e ferramentas – Programa OBSTÁCULO.

Orientação	Posição (mm) -- Desejada
-70.7 70.7 -0.0	400.0
-70.7 -70.7 0.0	-0.0
0.0 -0.0 100.0	500.0

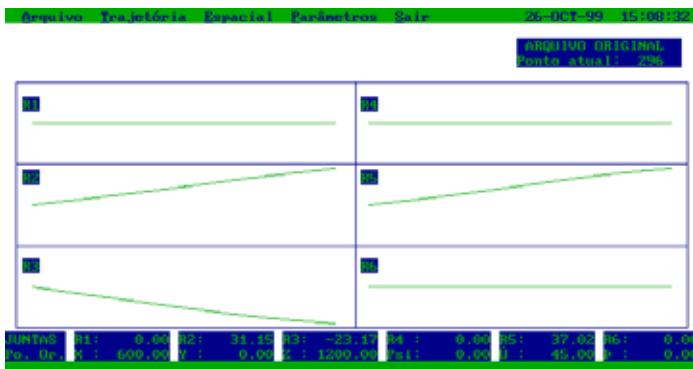
Orientação	Posição (mm) -- Atual
-70.7 70.7 0.0	399.6
-70.7 -70.7 -0.0	-0.3
0.0 -0.0 100.0	500.2

Erro : orientação = 0.0 0.0 0.0 posição = 0.4 0.3 -0.2

Número de iterações = 69

Junta(s) 2 4 6 fora do(s) limite(s)

a) Módulo de criação de trajetórias



b) Evolução angular das juntas no tempo.

Figura 10.5: Geração de trajetórias – Módulo TRAJETÓRIA.

10.2 – Integração de Robôs

Os conceitos apresentados anteriormente direcionam à realização de uma análise da aplicabilidade enfocando a concepção e planejamento no uso da simulação e programação off-line através de estudo de casos, permitindo-se uma análise crítica do problema, com ênfase na sua estruturação. Assim, se torna possível a integração de dispositivos mecatrônicos utilizando-se de forma objetiva e racional os recursos disponíveis na sintaxe de programação de robôs, deixando ao operador um número reduzido de operações realizadas pelo processo de aprendizagem.

O fato de possuir um ferramental para simular e programar um novo processo não garante uma programação econômica que atenue a gravação de pontos que caracterizam o processo. É preciso principalmente planejar o processo, definindo a melhor estratégia de aprendizagem, garantindo menos pontos de aprendizagem, e consequentemente um menor tempo de programação do operador no processo de aprendizado de pontos.

Esta ferramenta possibilita uma visualização do ambiente no qual, está inserido o processo, podendo ser previsto uma maior eficiência de alguns movimentos que comprometem a tarefa. A estruturação do programa é realizada textualmente, reprogramando a sintaxe da linguagem no qual se deseja trabalhar. Nesta fase é preciso conhecer os níveis de possibilidades que uma sintaxe disponibiliza somado à rotina de movimentos que se deseja implementar. Este conjunto de concepções leva um programa à condição de estruturado.

10.3 – Programação Off-Line utilizando Softwares Comerciais

A integração de dispositivos automatizados exige um alto nível de estruturação em automação utilizando programação off-line, sendo imprescindível a utilização de ferramentas para a estruturação das possíveis atividades em processos de concepção, implementação e projeto, propiciando a utilização da ferramenta como uma metodologia no auxílio da engenharia. Redes de Pétri (RdP) e o GRAFCET são ferramentas importantes de serem conhecidas e utilizadas para esta finalidade.

Os conceitos de programação off-line apresentados anteriormente são exemplificados a partir da utilização do *software* comercial “universal”, disponível no mercado, *Workspace*³ 5.04. Este aplicativo é considerado “*middle-end*”³, sua

3 1 “Midle end”. O conceito atribuído a esta expressão é a restrição na característica atribuída à interação de processo, tal como ajuste de parâmetros de soldagem, como a amperagem, tensão e etc. Não é possível ser tratado de forma a ser implementada no processo real, é dada uma característica meramente funcional para enriquecer a visualização na simulação.

performance é satisfatória, justificado pela relação de Custo / Benefício em relação a outros softwares “*high-end*”, envolvendo os fatores básicos de programação, métodos de programação, integração de dispositivos automatizados e redes de comunicação industriais e virtuais.

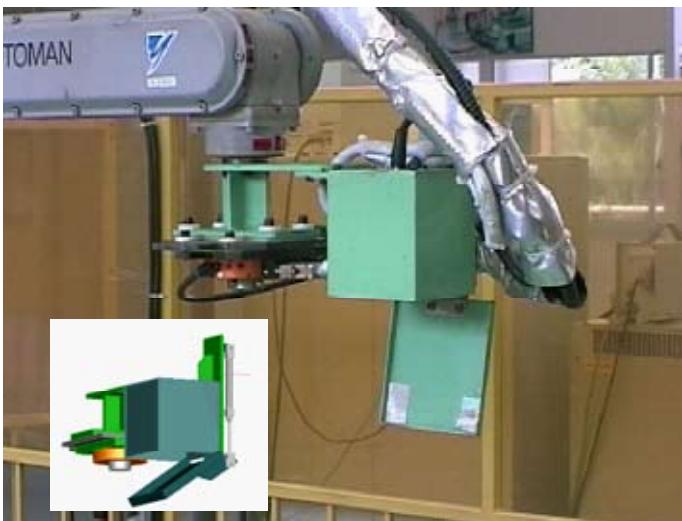
A partir da utilização do software Workspace®, constatamos a tecnologia que implementa as abordagens cinematáticas com suas possibilidades e variantes, a utilização de métodos numéricos iterativos, possibilitando uma convergência possível de solução, através do método recursivo, utilizando os cálculos dos modelos cinematográfico direto e inverso através da resolução da matriz Jacobiana.

Para exemplificar estes conceitos serão apresentados a seguir duas soluções industriais distintas, a primeira considerando um sistema de célula automatizada robotizada existente na Escola SENAI e Faculdade SENAI de Tecnologia Mecatrônica “Armando de Arruda Pereira”, São Paulo, onde a integração e arquitetura foram desenvolvidas pelo fabricante e integrador. Essas soluções permitem enfatizarmos aspectos concernentes aos processos de programação avançados, onde existe a possibilidade real de ganho no processo, e uma utilização racional nos processos de programação que venha a agregar de forma significativa na concepção e integração de dispositivos mecatrônicos.

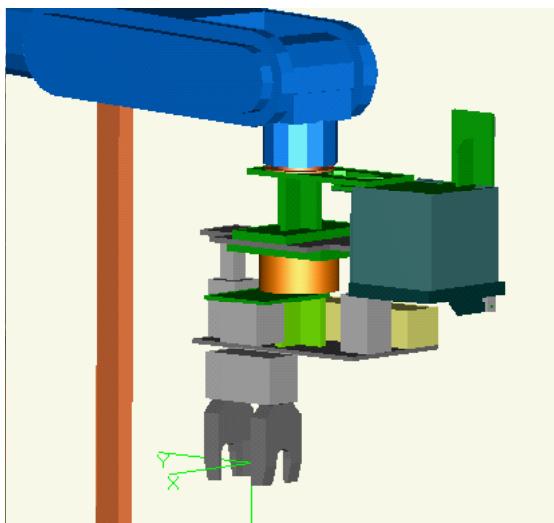
10.4 – Integração de Células Robotizadas

A maior parte de células automatizadas robotizadas existentes no mercado se caracteriza como sistema proprietário “fechado”. As frequentes e necessárias alterações de configuração exigem que o mercado utilize metodologias direcionadas à concepção de sistemas de integração “abertos” entre os robôs e dispositivos periféricos, permitindo a concepção e *upgrade* de novas células robotizadas. Uma das características de um sistema proprietário “fechado” é a dificuldade de utilização do seu sistema pelo usuário, fazendo uso somente em relação às tarefas projetadas e disponíveis.

Ao fazer uso de uma metodologia, para conceber e implantar novos projetos com a concepção de programação off-line de robôs, torna-se necessário o desenvolvimento de ferramentas dedicadas e dispositivos periféricos, para serem integradas no manipulador robótico, tais como o estudo da implementação dos mecanismos, em que se modelaram os dispositivos periféricos, tratados através do software Workspace®, e a movimentação do mecanismo foi implementando, através da cinemática direta de um sistema com N links, como mostra a figura 10.6.



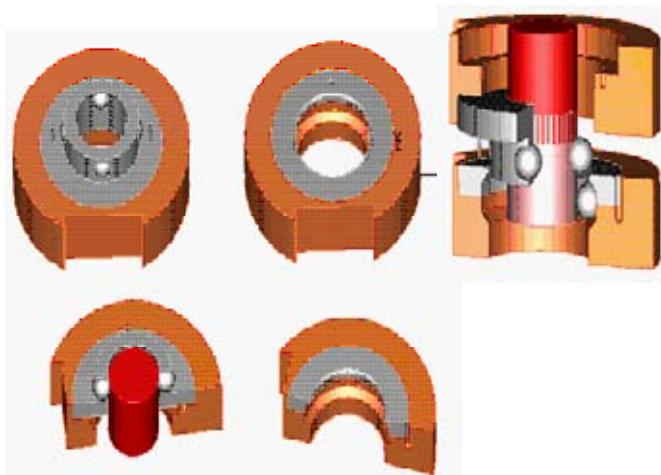
a) Sistema Real Implementado.



b) Implementação Virtual

Figura 10.6: Dispositivo Periférico.

Outro estudo possível foi a utilização da ferramenta para análise de dispositivo, fazendo a análise cinemática de operação do dispositivo periférico, para troca rápida da ferramenta de final de braço do robô, conforme ilustrada na figura 10.7, dispositivo para troca rápida.



a) Dispositivo Implementado.



b) Implementação Virtual.

Figura 10.7: Dispositivo para Troca Rápida de Ferramentas.

10.5 – Integração de Diferentes Dispositivos a partir de E/S

Através da utilização da porta serial de comunicação que possui diversas Entradas e Saídas (E/S) ou I/O (*Input-output*), e a partir do tratamento sequencial das informações, estabelece-se um protocolo de comunicação sequencial entre o robô e dispositivos periféricos integrados na célula automatizada, implementando através do software Workspace®, sendo possível a inclusão de ferramentas de integração entre robô e dispositivo periférico, através de ferramentas de modelagem de sistemas discretos (como, por exemplo, GRAFCET ou SFC (Sequential Flow Chart) e Redes de Pétri (RdP).

A figura 10.8 mostra uma célula de soldagem robotizada implementada no SENAI, onde as diferentes operações entre os elementos constituintes da célula (movimentação da mesa, ferramenta e robô) são realizados a partir do tratamento sequencial entre as entradas e saídas digitais. O programa implementado no *software* de programação off-line Workspace® é apresentado na figura 10.9.

Nos próximos itens apresentaremos dois exemplos implementados: Célula Robotizada Integrada de Manufatura, implementada no SENAI-SP, e Célula Robotizada de Soldagem, implementada no Laboratório de Automação Integrada e Robótica da UNICAMP. A implementação destes exemplos apresentam um alto grau de estruturação na programação necessário para sincronização, utilizando entradas e saídas dos robôs e dispositivos de integração utilizados dentro desta célula de trabalho.

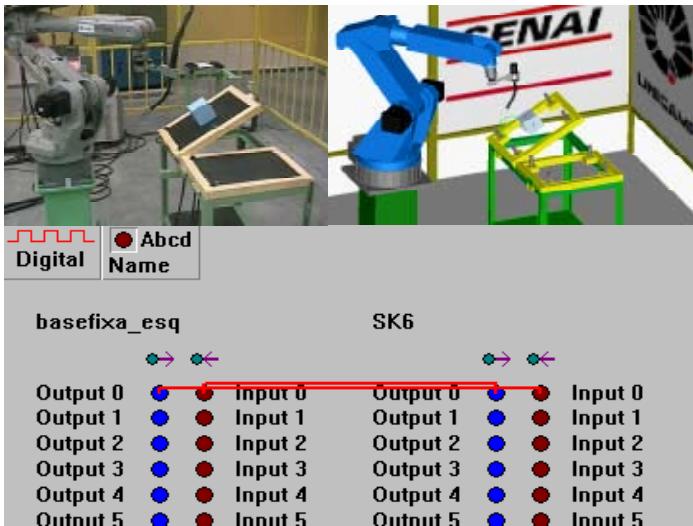


Figura 10.8: Integração Real–Virtual de Robôs e Dispositivos Periféricos através de I/Os.

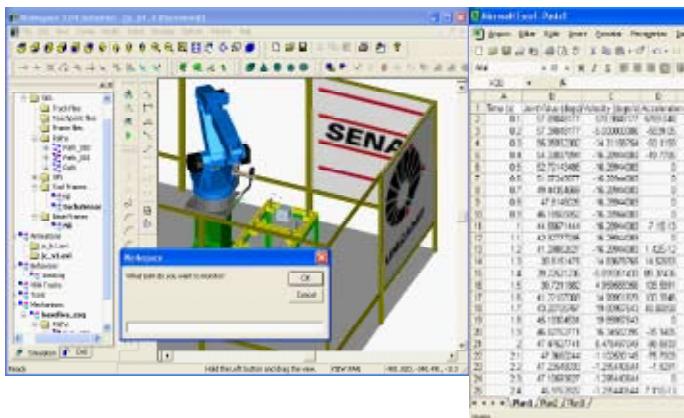


Figura 10.9: Integração de Robôs e dispositivos periféricos através de E/S.

10.6 – Sistemas Robóticos Cooperativos

A figura 10.10 apresenta um exemplo de cooperação robótica, em que uma operação de soldagem é realizada através da cooperação de dois robôs utilizando um modelo cinemático redundante (são considerados como uma cadeia cinemática fechada com 12 graus de liberdade, seis de cada robô), eliminando-se a possibilidade de sincronização E/S descrito anteriormente, e possibilitando a geração de movimentos contínuos, considerando o objetivo final de soldagem expresso em relação à ferramenta terminal dos robôs.

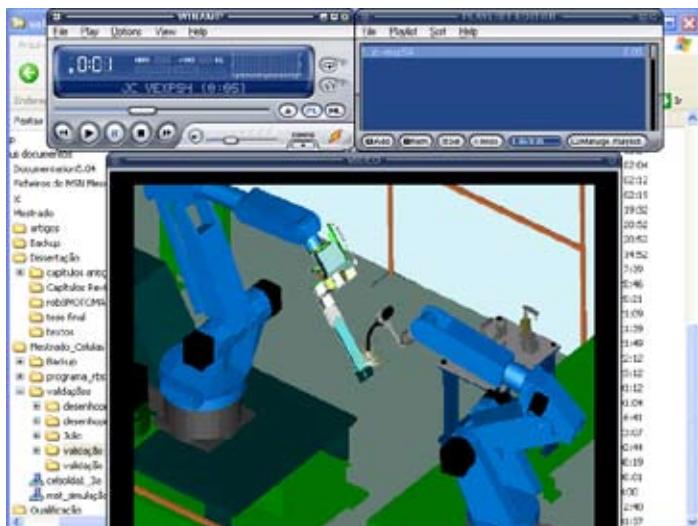


Figura 10.10: Operação de Soldagem Cooperativa.

Assim, um robô fará a operação de preensão, posicionando e orientando o elemento de soldagem, e o segundo robô desenvolverá o processo de soldagem “GMAW”. Essa integração permitirá uma maior flexibilidade de movimentação, o que facilita o posicionamento e orientação satisfatória no processo de soldagem.

A concepção desta célula automatizada foi implementada na Escola SENAI-SP e possui uma característica que possibilita o controle dos braços mecânicos de forma mais eficiente devido a um único modelo cinemático e dinâmico para os dois robôs, contemplando a facilidade de executar trabalhos cooperativos. Essa característica é fornecida pelo fabricante dos robôs que disponibilizou uma única unidade de controle para os dois braços mecânicos (com modelo cinemático embarcado).

De um modo geral, o *software* disponibilizado pelo fabricante não contempla esta possibilidade, pelo menos de forma direta de uso, sendo necessário, na maior parte dos casos, utilizarem lógica sequencial das E/S digitais, disponibilizadas no controlador para integrar diferentes dispositivos, e sincronizar a execução das tarefas cooperativas pré-programadas para uma célula automatizada.

10.7 – Modelagem e Simulação Off-Line

10.7.1 – Modelagem

Um software comercial permite simular o comportamento de um número variado de robôs industriais e de inserir desenhos relativos à modelagem em três dimensões do ambiente completo de trabalho do robô.

Para cada robô integrado à célula robotizada, deverá ser realizado a modelagem do ambiente de trabalho, ou seja, o modelo geométrico relativo ao espaço de trabalho onde serão integrados os robôs e os diferentes dispositivos constituintes desta célula de trabalho.

A partir da realização da modelagem desses diferentes elementos, torna-se conveniente verificar o espaço de trabalho dos robôs, ou seja, o espaço acessível para atuação da ferramenta terminal do robô, para verificar a correspondência entre as zonas de trabalho necessárias e determinar as eventuais regiões de colisões com os diferentes componentes da célula. Com essa finalidade, basta selecionar a opção envelope para visualizar

o espaço de trabalho e realizar uma primeira verificação, como mostra a figura 10.11.

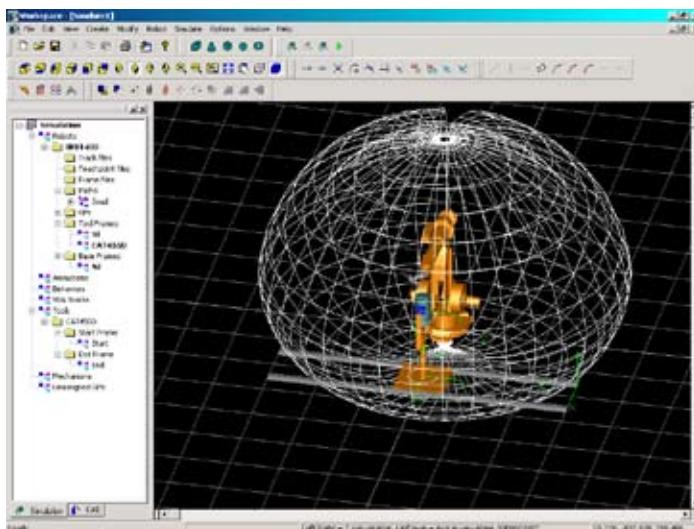


Figura 10.11: Espaço de Trabalho do Robô no interior da Célula.

10.7.2 – Simulação

Definido o ambiente de atuação e características geométricas da célula, a próxima fase de um projeto consiste na simulação e definição das ações dos atuadores. Isto pode ser realizado a partir da definição de trajetórias dos robôs através da programação de diferentes pontos de passagem (GPs), e definição de parâmetros relativos a velocidade, aceleração, tipo de trajetória desejada entre dois pontos (interpolação linear, trajetória PTP, etc.).

O software a ser implementado deverá prever igualmente a sincronização entre os diferentes elementos constituintes da célula automatizada. Ele deverá conter uma tabela com uma lista das entradas/saídas (numéricas e analógicas) de cada dispositivo automatizado, para conexão entre eles e troca de informações digitais para assegurar a sincronização entre os

robôs e elementos automatizada constituintes da célula através dessa tabela.

A ferramenta GRAFCET (SFC) é muito interessante de ser utilizada para esse tipo de aplicação, em que cada ponto de passagem está associado a uma ação, seja pela mudança de estado da porta, ou pela espera de um valor específico na porta utilizada (transição).

10.7.3 – Exemplo

Consideremos o exemplo de um robô trabalhando cooperativamente com outro dispositivo realizando uma operação de soldagem. Este robô deverá atender uma ordem para efetuar a operação de soldagem, necessitando que o outro dispositivo carregue a peça para a operação, e quando esta operação for finalizada deverá efetuar a operação de descarregamento, obedecendo à estrutura proposta na tabela 10.1.

Dispositivo	Robô	Posição	Ação / Transição (valor porta)
GP _d 1		Carregamento	
GP _C 2		Soldagem	0 -> 1
	GP _R 1	Repouso	1 -> 0
	GP _R 2	Soldagem	
	GP _R 1	Repouso	1 -> 1
GP _d 3		Descarregamento	

Tabela 10.1: Estrutura de Troca de Informações E/S.

Após especificação das diferentes ações referente aos pontos de passagem, as entradas/saídas entre os diferentes dispositivos automatizados no painel de conexão virtual ficaram disponibilizadas no *software* de programação *off-line*. A figura 10.12 apresenta um exemplo de integração de dois robôs industriais com outro dispositivo (designado Obj0).

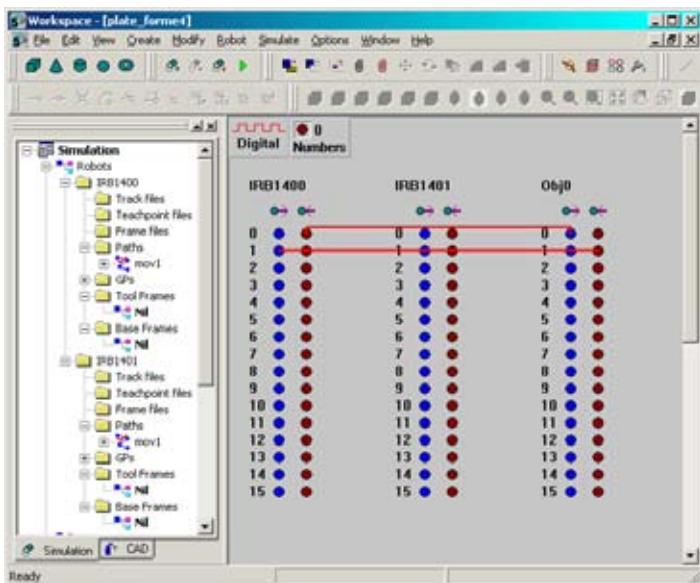


Figura 10.12: Painel de conexão virtual entre robôs e dispositivos automatizados da célula.

Uma vez terminada esta operação, as trajetórias do robô e do dispositivo deverão ser sincronizadas, sendo necessário associarmos outras ações a alguns pontos de passagem de forma a colocar em funcionamento a tocha de soldagem durante a operação de soldagem. Isto pode ser realizado através de uma temporização para uma operação de soldagem ponto-a-ponto ou contínua entre diversos pontos para obtenção de um cordão de solda.

A programação off-line com o software WORKSPACE permite obtermos as linhas de código correspondente à simu-

lação de grande parte das linguagens de programação de robôs industriais (neste caso, a linguagem RAPID para o robô ABB). Este código poderá ser diretamente transferido a unidade controle dos robôs através de uma conexão por rede ou disquete.

Dentro de uma aplicação real, poderemos integrar esses dispositivos a partir de uma aplicação em ambiente *Labview*TM. Neste caso, os modelo cinemático direto e inverso são necessários para implementação do sistema de supervisão e controle do dispositivo. Assim, o modelo cinemático fornece o posicionamento do dispositivo em função das coordenadas articulares e o modelo cinemático inverso fornece as coordenadas articulares em função da posição orientação final do dispositivo.

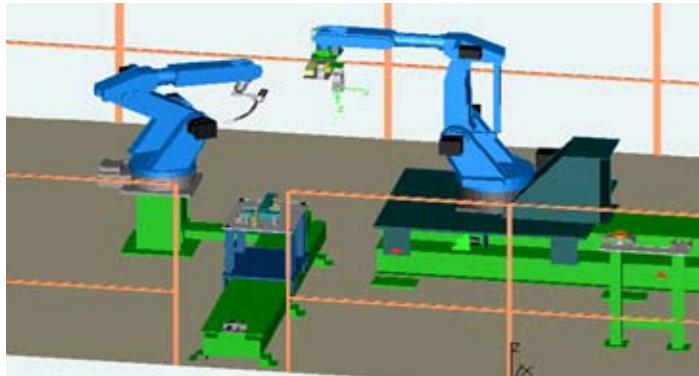
A seguir apresentaremos outros exemplos de programação off-line, utilizando softwares comerciais disponíveis no mercado.

10.8 – Exemplo: Célula Integrada de Manufatura

Na Escola SENAI-SP foi implementada uma Célula Automatizada Robotizada (figura 10.13). Esta célula é constituída de dois robôs industriais do fabricante Motoman / Yaskawa Modelos Sk6 e Sk16. A principal característica da unidade de controle desta célula é a concepção de uma única unidade de controle para os dois robôs, unidade está que pode controlar simultaneamente 21 GL “graus de liberdade” (deslocamentos dos eixos de forma dependente ou independente).



a) Célula implementada.



b) Célula Virtual (Workspace®).

Figura 10.13: Célula Automatizada Robotizada.

Esta célula é constituída de dois robôs SK6. Cada um deles possui uma ferramenta de final de braço. O primeiro robô possui a capacidade de trabalhar com 6 Kgf e está utilizando uma ferramenta de final de braço para processo de soldagem “GMAW” (tocha de solda). O segundo possui uma capacidade de trabalho de 16 Kgf e um sistema de troca rápido de ferramenta, “*Tool Changing*”, possibilitando a utilização de duas ferramentas de final de braço.

É possível a utilização de duas garras, uma para atuar no processo de soldagem, segurando e posicionando a peça a ser soldada, e a segunda, também uma garra, porém com outra configuração, para manusear peças utilizando o sensoriamento de um sistema de visão.

Existe um dispositivo periférico que tem como função o posicionamento e orientação das peças. No início do processo, na primeira operação de união, o dispositivo executa um travamento nas peças, possibilitando o trabalho de solda do robô. Sequencialmente, após a primeira operação de solda, o dispositivo desloca-se, posiciona-se e destrava a peça para, com o auxílio do segundo robô, pegá-la e posicioná-la na sequência da operação de soldagem.

A célula automatizada tem como característica a integração com um “CLP”, que gerencia os programas dos dois ro-

bôs e o sensoriamento dos periféricos. Essa concepção permite rodar dois programas, um em relação à unidade de controle do robô e outro do próprio Controlador Programável – CLP, que gerencia a célula, utilizando uma lógica sequencial de E/S.

10.9 – Identificação de Posicionamento de Robôs

A integração de robôs no ambiente de trabalho necessita do posicionamento real do mesmo em relação às tarefas a serem realizadas através da programação e elaboração de trajetórias pré-definidas em robôs, independente da operação.

Um exemplo típico é o caso de uma célula robotizada de soldagem de veículos (figura 10.14), que exige o desenvolvimento de um conjunto de procedimentos para calibração do sistema de referência do “zero veículo”, que consiste do cálculo do posicionamento real de um determinado robô constituinte da célula em relação a um determinado ponto de referência (“zero veículo”) para reposicionamento do referencial base do robô, permitindo assim a utilização de arquivos de tarefas gerados através de um software de programação off-line.



Figura 10.14: Célula Robotizada de Montagem de Veículos.

Este processo pode ser otimizado através da utilização de softwares industriais para programação off-line, que permitem simular com relativa precisão todo o trabalho a ser realizado pelo robô, desde a sua instalação, otimização

de tarefas e carregamento do programa no robô. Os procedimentos de calibração e identificação podem ser realizados de duas maneiras:

- a) Contato Mecânico: Método tradicional de identificação que utiliza uma ferramenta de calibração para ser anexada à ferramenta terminal do robô ou um procedimento manual através de toques em dispositivo fixado no palete;
- b) Visão Robótica: Através do desenvolvimento de software dedicado em linguagem *LabviewTM* para calibração automática e identificação de pontos do palete através de sistema de Visão Robótica, sem a necessidade de que o operador realize contatos manuais com a peça.

Neste tópico será apresentada uma proposta de implementação de programa computacional em ambiente LabviewTM utilizando o Sistema de Visão da National InstrumentsTM. Este pacote de programas, designado Programa Gestor (figura 10.15), é constituído de quatro módulos básicos:

- a) Calibração da Ferramenta Terminal do Robô;
- b) Identificação do Zero Veículo através de pontos perfeitamente conhecidos no palete de medição, por meio do cálculo da nova posição e orientação do robô utilizado;
- c) Programa de Conversão de Ângulos de Orientação da ferramenta;
- d) Entrada automática de dados do palete.



Figura 10.15: Programa Gestor.

10.9.1 – Metodologia de Calibração

A qualidade da sistemática de calibração apresentada é função do posicionamento real do robô em relação ao referencial zero. A execução de tarefas pré-programadas deverá considerar a identificação exata dos sistemas de referências externos ao robô, através de procedimentos de calibração automáticos. A partir de uma série de medidas simples pré-definidas será possível a determinação do sistema real de posicionamento do robô em relação ao referencial zero e em relação ao sistema externo de medição.

A figura 10.16 apresenta uma proposta de metodologia para correção de erro de posicionamento de um robô (cálculo do posicionamento real). Esta metodologia consiste em um novo cálculo do zero de posicionamento (posição e orientação da base do robô em relação ao palete do veículo), com correção a ser realizada posteriormente no robô.

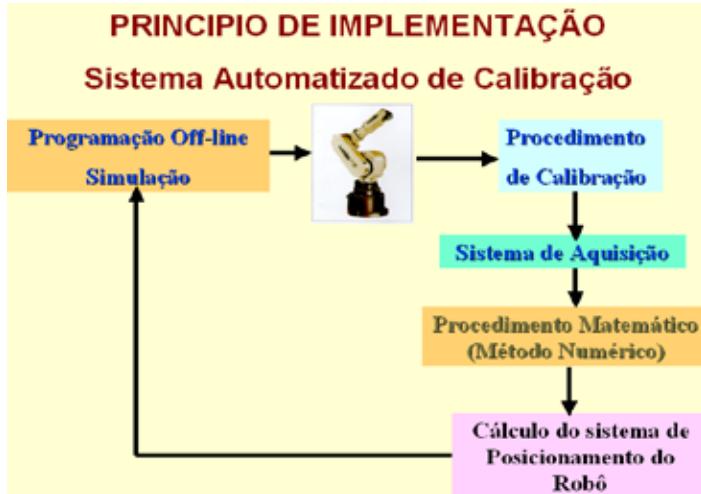


Figura 10.16: Sistemática de Calibração.

Durante o desenvolvimento desse projeto de pesquisa as seguintes etapas foram realizadas:

- Implementação de programa off-line do robô utilizando dimensões reais do processo;
- Calibração da Ferramenta de Calibração desenvolvida através do toque em diferentes orientações da haste (procedimento manual) ou no caso do sistema de visão robótica desenvolvido, através da visualização do diâmetro constante de círculo de esfera colocada num furo de precisão do palete (procedimento Automatizado). Os pontos adquiridos deverão ser feitos diretamente no robô, e juntamente com as dimensões aproximadas da ferramenta servirão de informação de entrada do programa de calibração desenvolvido;
- Processamento destas informações através do módulo computacional CALIBRAÇÃO DA FERRAMENTA, permitindo assim a obtenção das dimensões precisas da ferramenta, que deverão ser utilizadas no programa de cálculo do zero do robô;

- d) Entrada de dados do palete, através de módulo computacional específico desenvolvido (PALETE). Neste módulo deverá ser introduzida a sequência de pontos de calibração que deverão ser utilizadas no programa CALIBRAÇÃO;
- e) Implementação de programa off-line do robô, contendo as dimensões da FERRAMENTA de calibração, e pontos de precisão do paleta e sequência de posicionamento, contemplando etapas de aproximação da ferramenta de calibração (haste ou visão robótica), utilizando as dimensões reais do processo. Com o objetivo de minimizarmos possíveis erros, a orientação da ferramenta deverá ser mantida aproximadamente constante;
- f) Carregamento no robô do programa desenvolvido na etapa anterior, e posicionamento automático da ferramenta em relação ao paleta (utilizando a haste de contato no pino de precisão introduzido nos furos do paleta, ou através de sistema de visão robótica, que deverá visualizar círculo de diâmetro constante na esfera introduzida no furo do paleta). Os pontos adquiridos deverão ser armazenados em arquivo de dados;
- g) Processamento das informações obtidas por este arquivo, que juntamente com as entradas automáticas das dimensões do paleta e sequência de pontos zero do robô, servirão de informação de entrada do módulo computacional PALETE, que permitirá a obtenção do novo zero do robô em relação ao paleta.

10.9.2 – Infraestrutura necessária

Para utilização do procedimento descrito anteriormente, é necessária a utilização da seguinte infraestrutura:

10.9.2.1 - Dispositivos Mecânicos

Como principal dispositivo deverá ser utilizada uma ferramenta universal de calibração contendo dispositivo de

fixação da haste de calibração ou da câmera, com possibilidade de rotação em diferentes orientações, permitindo que a mesma se adapte facilmente ao ambiente e espaço disponível de calibração. Os acessórios necessários serão os seguintes:

- a) Dispositivo de Calibração por toque (procedimento tradicional): constituído de pinos de calibração e uma ponta de precisão a serem introduzidos nos furos de precisão do palete, para posicionamento da haste de calibração;
- b) Dispositivo de Calibração através de Visão Robótica (procedimento automático): constituída de uma esfera de calibração posicionada no furo de precisão do paleta, para aproximação da câmera CCD, fixada na ferramenta de calibração.

10.9.2.2 – Hardware e Software

Além do software industrial de programação do Robô e programação off-line (RobotStudioTM) disponibilizados para a aplicação, deverá ser utilizado um programa Gestor para cálculo do zero do robô e calibração da ferramenta, e um programa de ajuda à aprendizagem de pontos do paleta e calibração da ferramenta utilizando um sistema de Visão Industrial da National InstrumentsTM.

Para o correto funcionamento destes dois aplicativos de calibração que trabalham em ambiente Windows, recomenda-se a seguinte configuração básica (mínima) de equipamentos necessários:

- Interface de Visão National Instruments IMAQ PCI/PXI – 1407 com configuração computacional mínima de memória RAM 128 MB.
- Câmera Digital - Fonte de Alimentação (12V) – GANZ – Modelo YCH-02 – Color-High-Solution – 24 VAC, 12 VDC - 3,5-10,5 mm 1:1.0 IR 1/3 pol.
- Dispositivo para acoplamento a Ferramenta de Calibração.

10.9.3 – Software de Identificação de Pontos no Palete e Calibração de Ferramenta

Para automação do procedimento de identificação dos furos de precisão do palete foi confeccionada uma ferramenta de calibração utilizando uma câmera CCD para Visão Robótica anexada à pinça de soldagem, conforme é mostrado na figura 10.17.

Para esta aplicação propõe a utilização de *software dedicado em linguagem LABVIEWTM* para calibração da ferramenta para identificação de pontos do palete, a partir da utilização de uma câmera CCD, sem a necessidade de contato do operador com a peça. Esses programas computacionais permitiram a obtenção dos dados necessários para utilização do conjunto de programas computacionais descritos anteriormente para cálculo das dimensões da ferramenta e para identificação do *zero veículo* através da medição dos pontos de precisão do palete pela movimentação do robô.

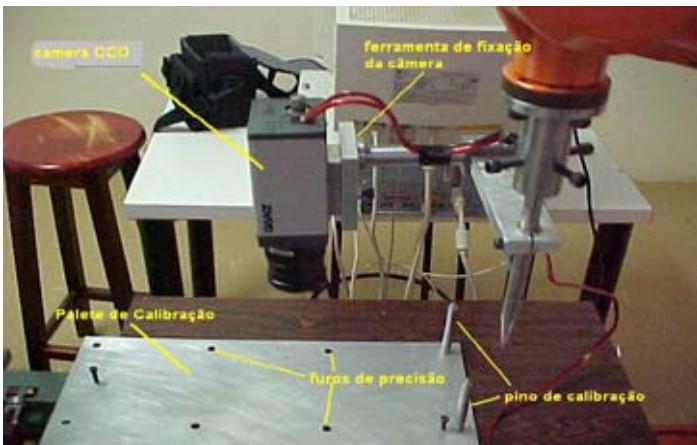


Figura 10.17: Sistema de Visão Industrial.

Este aplicativo de Visão Robótica foi implementado em linguagem LABVIEWTM, permitindo a calibração da ferramenta e palete de precisão. Através deste aplicativo tornou-se possível:

- a) Ajuste focal e obtenção do diâmetro do círculo de referência a partir de uma esfera de calibração: permite a configuração dos parâmetros do dispositivo de visão robótica, que foram utilizados posteriormente para calibração da ferramenta e identificação do zero do robô a partir do palete de precisão;
- b) Calibração da ferramenta utilizando Visão Robótica: permite a obtenção de pontos necessários para calibração da ferramenta de calibração adaptada à pinça de soldagem do robô (leitura de três pontos, através do mesmo posicionamento, mas em diferentes orientações da ferramenta do robô).
- c) Obtenção dos pontos de precisão do paleta pelo robô: permite a obtenção de pontos de precisão do paleta através da movimentação do robô, conservando a mesma orientação da ferramenta.

10.9.4 – Aplicativos Desenvolvidos

10.9.4.1 - Sistema de Calibração

Ao executarmos esse programa é apresentada uma tela inicial do sistema de calibração (figura 10.18), onde podemos observar duas possibilidades:

- a) Configuração de parâmetros: permite configurar todos os parâmetros necessários para calibração da esfera (busca do diâmetro), ajuste focal, tolerância máxima, etc. Esses parâmetros são salvados automaticamente, permanecendo para serem utilizados por *default, até alteração ou nova calibração pelo usuário*;
- b) Efetuar a calibração: neste modo é realizada a calibração tanto da ferramenta quanto do paleta, utilizando os valores de referência obtidos anteriormente e armazenados em variável de memória.



Figura 10.18: Tela inicial do Sistema de Calibração.

10.9.4.2 – Ajuste de Parâmetros de Calibração

Este módulo permite o ajuste de parâmetros de calibração do sistema de Visão Robótica (foco e zoom). Ele contém uma ferramenta de suporte que permite a obtenção do raio do círculo de projeção da câmera sobre a esfera que deverá ser utilizado como referência no procedimento de calibração da ferramenta e a obtenção do novo zero do robô (figura 10.19). Durante o ajuste focal, será permitida a visualização de tolerância de medição, que poderá ser modificada para mais ou menos, em função da precisão desejada durante o processo.

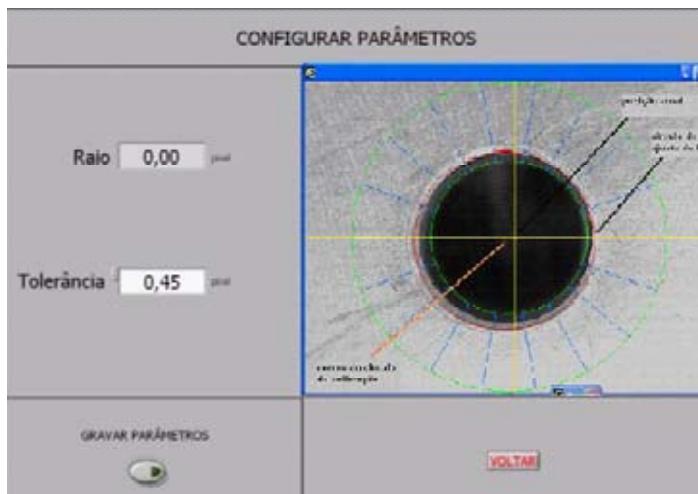


Figura 10.19: Tela de Calibração de Parâmetros.

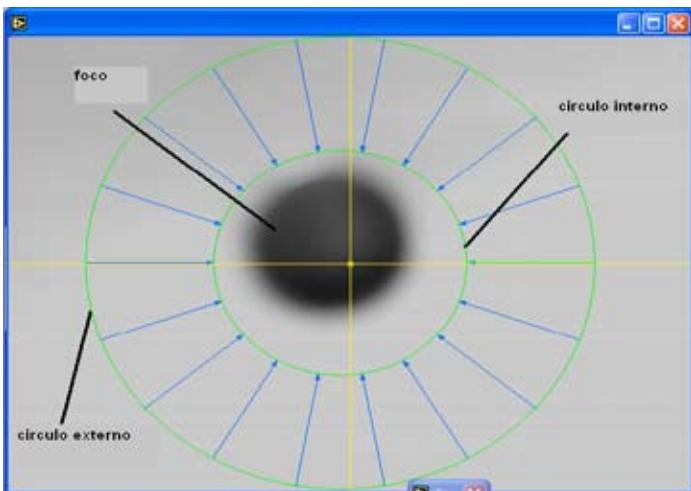
10.9.4.3 - Procedimento de Ajuste Focal

Este procedimento consiste no ajuste do foco e zoom da câmera robótica em torno do diâmetro de uma esfera de calibração, que deverá ser mantido constante até o processo final de calibração da ferramenta e do palete (obtenção do novo zero do robô). Após esses parâmetros serem armazenados, os mesmos são utilizados como default até o final do processo de calibração (o foco e zoom da câmera não poderão ser alterados, pois isto implicaria numa nova etapa de ajuste de parâmetros). Os seguintes passos deverão ser utilizados pelo usuário durante o processo de obtenção do raio de referência para calibração:

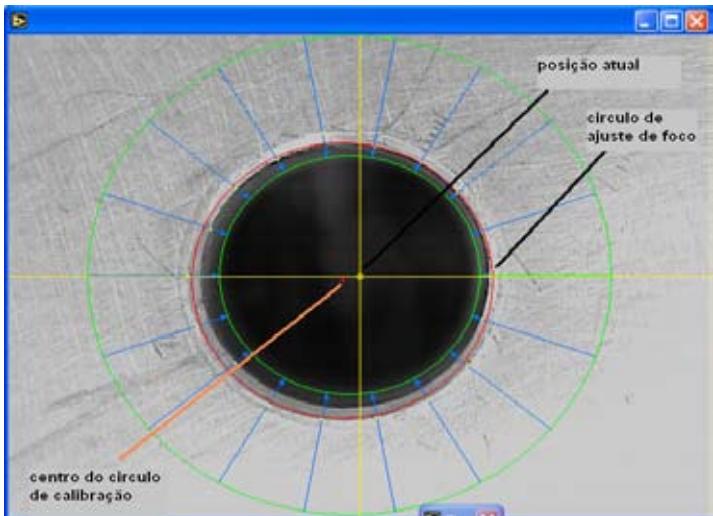
- Passo 1:** regulagem do foco e zoom da câmera robótica (figura 10.20).
- Passo 2:** ajuste da câmera em relação à esfera de calibração de modo a ser identificado um círculo de raio constante, para ser utilizado como sistema de referencia da câmera.

Em outras palavras, em qualquer orientação do robô será visualizado pela câmera um círculo de mesmo raio.

- iii) **Passo 3:** No momento em que todos esses parâmetros estiverem bem ajustados pelo usuário, estes valores deverão ser salvos, e permanecerão como *default* até realização de novo ajuste e salvaguarda utilizando a mesma tela de ajustes.



a) Ajuste de foco – tela típica.



b) Ajuste de parâmetros de foco.

Figura 10.20: Telas típicas obtidas durante o Ajuste Focal.'

A interface com o usuário apresentada na figura 10.21 apresenta o raio do círculo escolhido para referência de calibração. Esse raio aparecerá no momento que o ajuste focal estiver dentro dos limites mínimos e máximos do software desenvolvido. Em função da precisão do processo de medida, o usuário poderá ajustar, através de botão específico, a tolerância estipulada por *default*.

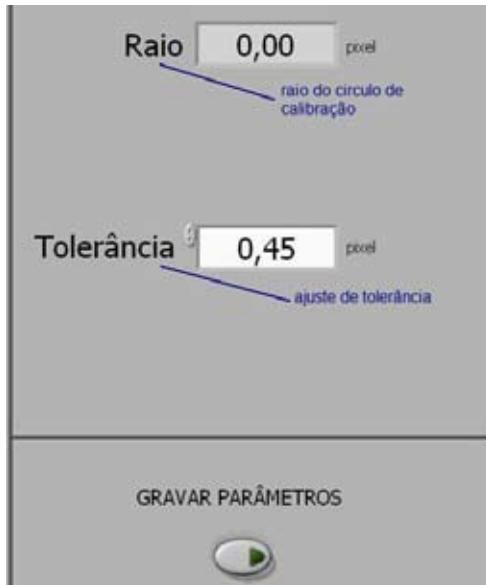


Figura 10.21: Tela de Ajuste do raio de referência da esfera de calibração e tolerância.

10.9.4.4 - Operação de calibração

Este módulo é utilizado para a calibração da ferramenta do robô e do palete (obtenção do novo zero do robô). Quando utilizado para calibração da ferramenta, os três pontos são obtidos a partir do mesmo posicionamento e alteração da orientação da ferramenta terminal. No caso da calibração do palete, a orientação da ferramenta é mantida constante alterando somente o posicionamento da mesma em relação ao palete. A figura 10.22 mostra a tela típica desse módulo que permite:

- a) Visualização do círculo de projeção da esfera de calibração, permitindo ao usuário o reposicionamento do robô através da aproximação do ponto em vermelho (centro desejado) ao ponto amarelo (centro de referência).
- b) Contém ferramentas gráficas que permitem uma

indicação visual do raio do círculo de projeção da câmera sobre a esfera e posicionamento do robô nas direções XY, e também para afastamento ou aproximação da esfera utilizada para calibração, com visualização do erro médio quadrático de distância do ponto (vermelho) ao centro (amarelo) (figura 10.23).

- c) Informações para ajuda ao usuário relativo ao valor do raio e desvio quadráticos atuais e raio e tolerância de referência.



Figura 10.22: Tela típica do programa de calibração.

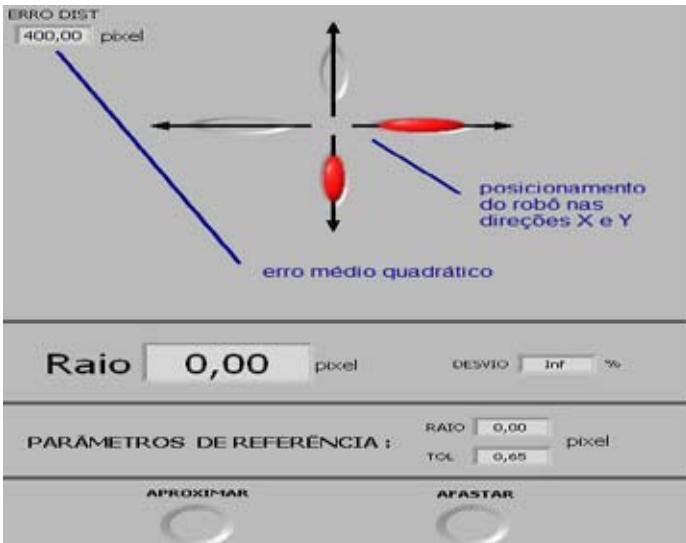


Figura 10.23: Tela de Calibração e parâmetros.

O aplicativo apresentado é um facilitador ao operador do robô para obtenção de três ou quatro pontos necessários para calibração da ferramenta terminal do robô, a partir da orientação da ferramenta terminal sobre uma esfera de calibração colocada como referência em posição fixa no espaço de trabalho do robô. Ele permite obter esses pontos através da mudança de orientação da ferramenta, mantendo sempre o mesmo diâmetro do círculo de projeção da esfera de calibração utilizada.

Para ajuste do posição da posição atual em relação ao centro do círculo, uma tela de ajuda a movimentação do usuário nas direções X, Y do robô, permitirá a movimentação manual do robô pelo operador nestas direções.

Os botões para afastamento ou aproximação da peça serão sinalizados (alteração da cor do botão) no momento que o raio da câmera saia da tolerância arbitrária, indicando ao usuário informações para que este aproxime ou afaste a câmera da esfera na direção do eixo Z dentro da tolerância utilizada. O ponto de calibração será considerado preciso no momento que as setas assinaladas em vermelho desapa-

recentemente, aparecendo uma mensagem informando que o ponto está calibrado.

10.9.4.5 – Aplicativo para Calibração do Paleta

Este aplicativo computacional tem como principal objetivo facilitar o operador a conseguir os pontos necessários para obtenção do posicionamento da ferramenta terminal do robô em relação ao paleta de calibração. O procedimento apresentado anteriormente para a calibração da ferramenta é basicamente o mesmo utilizado para calibração do paleta, utilizando-se as mesmas informações obtidas durante a fase de calibração da ferramenta (diâmetro do círculo de referência da esfera) para obtenção de 3 ou 4 pontos do paleta de precisão. É aconselhável que seja mantida a mesma orientação da ferramenta e altura da mesma em relação ao paleta, garantida através do mesmo diâmetro do círculo de projeção da esfera de calibração utilizada.

10.10 - Conclusão

Neste capítulo foram apresentados, através de exemplos práticos, aspectos concernentes aos processos de programação avançada de robôs, onde existe a possibilidade real de ganho no processo, enfocando uma utilização racional nos processos de programação e que venha a agregar de forma significativa a concepção de programação. Foi enfatizado também um estudo relativo à calibração de robôs, a partir de obtenção de parâmetros cinemáticos durante a simulação, em que a partir desse estudo de casos será possível a realização de uma análise da aplicabilidade enfocando a concepção e planejamento no uso da simulação e programação off-line de robôs. Um procedimento de calibração e identificação de posicionamento de um robô dentro de uma célula de manufatura também é apresentado.

Considerando a importância da metodologia apresentada nesse trabalho de pesquisa, no próximo capítulo será apre-

sentado uma proposta de formação profissional para o Projeto de Células Robotizadas industriais, utilizando conceitos de programação off-line, permitindo a viabilização exequibilidade de implementação dessas ferramentas e procedimentos em empresas industriais de pequeno e médio porte.

10.11 - Referências Bibliográficas

Acar, M.: **Mechatronics Challenge for the Higher Education World**, *IEEE Transactions on Components, Packing, and Manufacturing Technology*, vol. 20, no. 1, p.1420, 1997.

Anton, S., et al. **A framework for Realistic Robot Simulation and Visualisation** Disponível em <<http://www.easy-rob.com/data/Frame-RSS.pdf>>. Acesso 29 mar. 2003

Asfahl, C. Ray. Building Blocks of Automation in: Robots and Manufacturing Automation, 2nd ed. New York: John Willey & Sons, 1992 p 23-53.

Ashley, S.: **Getting a Hold on Mechatronics**, *Mechanical Engineering*, ASME, maio de 97, p. 6063, 1997.

Brumson, B. **Emerging Trends in Robotic Integration**. Robotics online e-news. Michigan, Nov.2001. Disponível em: <<http://www.roboticsonline.com/public/articles/archive-details.cfm?id=577>> Acesso 19 jan. 2004.

Brumson, B., Robots for small business: A Growing Trend. Robotics online e-news. Michigan, Out.2003, Disponível em: <<http://www.roboticsonline.com/public/articles/archivedetails.cfm?id=1231>> Acesso 19 jan. 2004.

Camelot, Ropsim, **Robot Off-line Programming and Simulation system for industrial robots**. Disponível em: <<http://www.camelot.dk/english/offvson.htm>> Acesso 26 jun. 2003.

Campbell, J.E., Robot Motion Control: How Special is it? Robotics online e-news. Michigan, Set.2003, Disponível em:

<<http://www.roboticsonline.com/public/articles/archivedetails.cfm?id=1174>>, Acesso 19 jan. 2004.

Easy-Rob,Easy-rob, The Company for 3D Robot Simulation, Disponível em: <<http://www.easy-rob.com>> Acesso 03 Set. 2003.

EF-Robotertechnik GmbH, Cosimir® Professional, Manual do usuário, 2002.

Fiedler, P. J., Schilb, C. J. **Open Architecture Robot Controllers and Workcell Integration.** Robotics online e-news. Technical Papers, Michigan. Disponível em: < http://www.roboticsonline.com/public/articles/Open_Architecture_Genesis.PDF > Acesso 17 set. 2003.

Fiedler, P. J., Dehof, M.K. Workcell Communications: Connectivity, Man-Machine interfaces, and multi system management . Robotics online e-news. Technical Papers, Michigan. Disponível em: <http://www.roboticsonline.com/public/articles/Workcell_Communications_ANS98_Genesis.PDF> Acesso 17 Set. 2003.

Flow Software Technologies, Workspace5, Robotic simulation and off-line programming software. Disponível em: <<http://www.workspace5.com>> Acesso 03 Set. 2003.

Flow Software Technologies Ltd, Workspace 5.04, Manual do usuário, 16 maio 2003.

Fu, K.S., González,R.C., Lee,C.S.G. **Robotics: Control, Sensing, Vision, and Intelligence, McGraw-Hill International, 1987, 580p**

Groover, Mikell P. Automation in: Dorf, Richard C. (Ed), Nof, Shimon Y. International Encyclopedia of Robotics: Applications and Automation, New York: John Willey & Sons, 1988 p 136-151

Hardin, W., **Robotic Integrators take their Specialties Seriously. Robotics online e-news. Michigan, Abr.2002**

Hewit, J.R., King, T.G.: Mechatronics Design for Product Enhancement, *IEE & ASME Transactions on Mechatronics*, vol. 1, no. 2, p. 111119, 1996.

Nagle, B. Your First Robot. **Robotics Industry Directory - 2002-03 Edition**, A publication of robotic Industries Association, p 8.

Nitsche, A.T., Romeiro Filho, E., **Aplicação de Tecnologias de CAD/CAE/CAM em Desenvolvimento de Produtos**. In: **Encontro Nacional de Engenharia de Produção – ENEGEP, XVII, 2002. Curitiba – PR.**

Nitsche, A.T., Romeiro Filho, E., Gonçalves, J. B., Gomes, D.T., Uso das Ferramentas de CAD/CAE/CAM no desenvolvimento de produtos: O Estado das Práticas em pequenas e médias empresas. *2ºCOBEF – 2003*

Robot Simulation ltd, Workspace 4.01, **Manual do usuário, 1999.**

Roos, E., Behrens, A. Off-line Programming of industrial robots – Adaptation of simulated user programs to the real environment. *Computers in industry*, 1997 33(1): p 139-150.

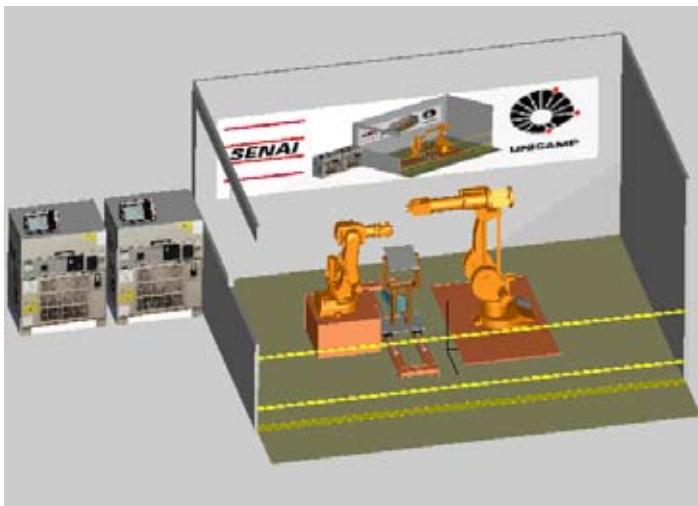
Salminen, V: Ten Years of Mechatronics Research and Industrial Applications in Finland, *IEEE - ASME Transactions on Mechatronics*, vol. 1, no. 2, p. 103105, 1996.

Trostman, E., Conrad, F., Trostman, S., Nielsen L.F. **Robot Off-line programming and simulation as a true CIME subsystem**. In: IEEE International Conference on Robotics and Automation. 1992

Vollmann, K. **Realistic Robot Simulation: Multiple Instantiating of Robot Controller Software** in: *IEEE ICI-T'02, International Conference On Industrial Technology, 2002* p 1194-1198.

CAPÍTULO 11

Exemplo de Implementação de uma Célula Colaborativa Robotizada



CAPÍTULO 11

Exemplo de Implementação de uma Célula Colaborativa Robotizada

Neste capítulo é apresentado um exemplo de implementação de uma célula colaborativa robotizada implementada no Laboratório de Automação Integrada e Robótica da Faculdade de Engenharia Mecânica da UNICAMP. Esta aplicação é baseada na coordenação e integração de dois robôs industriais e uma mesa de posicionamento com três graus de liberdade implementada para auxiliar o trabalho cooperativo de robôs manipuladores convencionais em trabalhos de usinagem e soldagem de dispositivos mecânicos complexos que necessitam de mais graus de liberdade para realizarem trajetórias complexas. Através desta aplicação são enfatizados aspectos concernentes aos processos de programação avançada de robôs e utilização de ferramentas de integração, onde existe a possibilidade real de ganho no processo, enfocando uma utilização racional nos processos de programação e que venha a agregar de forma significativa a concepção de programação. A implementação final foi realizada a partir de dois estudos:

- a) Plataforma WEB colaborativa em Automação: integração de dois robôs industriais junto com dispositivo robótico de 3 GL para realização de tarefas colaborativas automatizadas através de WEB;
- b) Implementação de Sistema de Supervisão e Controle de um dispositivo robótico com 3 GL.

11.1 – Introdução

A integração de robôs e dispositivos mecatrônicos dentro de Células Flexíveis de Manufatura envolve metodologias usando os conceitos de automação. O ambiente modelado por CAD pode ser associado com o desempenho do controle de robô incluindo equipamentos e mecanismos, modelagem matemática do robô (cinemática direta e inversa) e seus componentes acoplados, como, também, a integração e coordenação dos movimentos do robô com outros dispositivos.

Uma aplicação particular pode ser vista na figura 11.1, que é baseada na coordenação e integração de dois robôs industriais e uma plataforma de posicionamento (plataforma hidráulica). Esse dispositivo apresenta três graus de liberdade (robô PRR) e foi desenvolvido para auxiliar os robôs manipuladores em trabalhos de usinagem e soldagem de dispositivos mecânicos complexos (trajetórias complexas). Esta mesa de posicionamento pode auxiliar em tarefas nas quais os manipuladores tradicionais têm dificuldade em alcançar algumas partes da peça a ser trabalhada.



Figura 11.1: Célula Flexível de Manufatura.

Com a utilização destes procedimentos torna-se possível fazer uma implementação virtual que atenda de forma satisfatória uma situação real em relação a uma implementação virtual de um processo. A concepção de uma programação avançada a priori é atenuada no momento em que se busca uma integração com uma cooperação entre robôs e periféricos, visto que o processo de trabalho entre os processamentos da unidade de controle acontece de forma multiplexada, permitindo um entendimento e execução gerenciada no mesmo instante de tempo.

Como uma alternativa para a solução ao integrar e implementar uma programação executa-se as instruções programadas das E/S através de lógica sequencial e posterior comunicação através de porta serial. Essa solução não atende de forma geral a alguns tipos de processos, principalmente em atividades que se exigem respostas rápidas em relação aos movimentos dos braços robóticos, bem como na sincronização de movimentos.

11.2 – Célula Colaborativa Robotizada

Uma célula colaborativa (figura 11.2) é baseada na coordenação e integração de dois robôs industriais e uma mesa (dispositivo mecatrônico) com três graus de liberdade (robô PRR) desenvolvida para auxiliar em trabalhos como: soldagem, usinagem. Esta mesa pode auxiliar em tarefas em que os manipuladores tradicionais têm dificuldade para alcançar algumas partes da peça a ser trabalhada. Para isso, a mesa é sincronizada com os manipuladores, permitindo a realização de tarefas complexas.

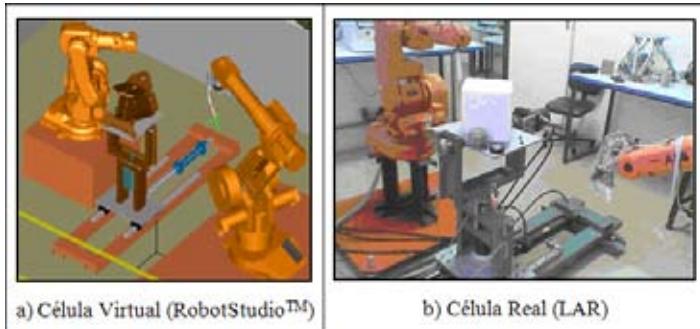
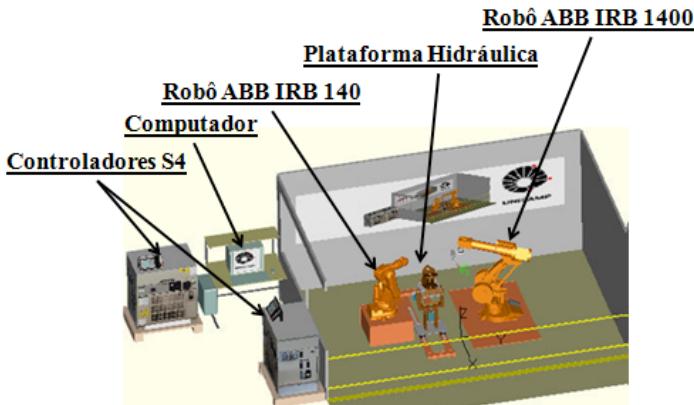


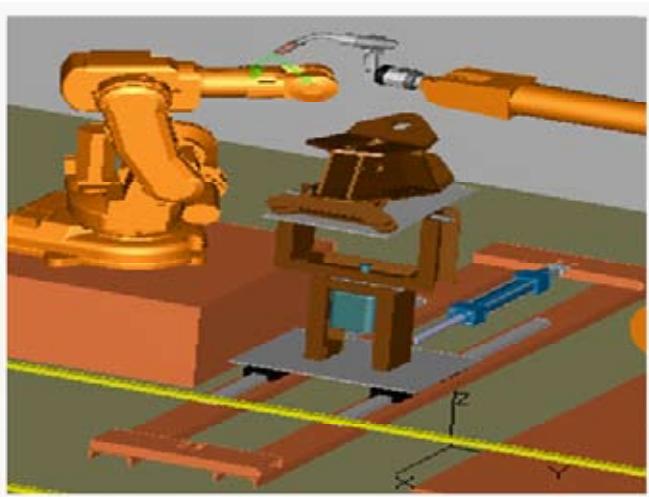
Figura 11.2: Célula Integrada Robotizada.

A figura 11.3a apresenta detalhes da programação virtual do sistema cooperativo implementado no Laboratório de Automação Integrada e Robótica da UNICAMP. Assim foi implementada uma Célula Integrada de Soldagem, constituída de dois robôs ABB, o IRB1400 com capacidade de carga de 14 Kgf, utilizando como ferramenta de final de braço uma tocha de solda, IRB140 com capacidade de carga de 5 Kgf e uma plataforma de posicionamento com 3GL, conforme figura 11.3b.

Para a execução da programação off-line, consideraram-se os 3GL do dispositivo mecatrônico como eixos externos do robô IRB 1400. Esta possibilidade é viável para o robô real, possibilitando os ajustes e integração de eixos externos como dispositivos a serem controlados pela unidade de controle do robô.



a) Elementos da Célula Robotizada.



b) Detalhe da Plataforma de Posicionamento e Ferramenta de Soldagem.
Figura 11.3: Célula Robotizada Virtual implementada na UNICAMP.

11.3 – Plataforma de Posicionamento

A plataforma de posicionamento é constituída de 3 GL (robô PRR) e trabalha de forma cooperativa realizando trabalhos de usinagem e soldagem de dispositivos mecânicos complexos (trajetórias complexas) com dois robôs industriais. Para melhor compreensão do problema, aqui é apresentada a mesa de posicionamento em estudo (figura 3.7). Essa mesa possui três atuadores hidráulicos.

Uma aplicação particular pode ser vista na figura 11.4, que é baseada na coordenação e integração de dois robôs industriais e uma mesa de posicionamento com acionamento hidráulico. Esse dispositivo apresenta três graus de liberdade: um atuador linear e dois rotativos (PRR), sendo desenvolvido para auxiliar os robôs manipuladores em trabalhos de usinagem e soldagem de dispositivos mecânicos complexos (trajetórias complexas). Esta mesa de posicionamento pode auxiliar em tarefas nas quais os manipuladores tradicionais têm dificuldade em alcançar algumas partes da peça a ser trabalhada.



a) Plataforma Real



b) Plataforma Virtual (*SolidWorks*™)

Figura 11.4: Plataforma Hidráulica de Posicionamento.

Para posicionamento das juntas da plataforma hidráulica foram utilizados *encoders incrementais* e *sensores indutivos*. A figura 11.5 apresenta a integração dos diferentes sensores utilizados na plataforma de posicionamento.

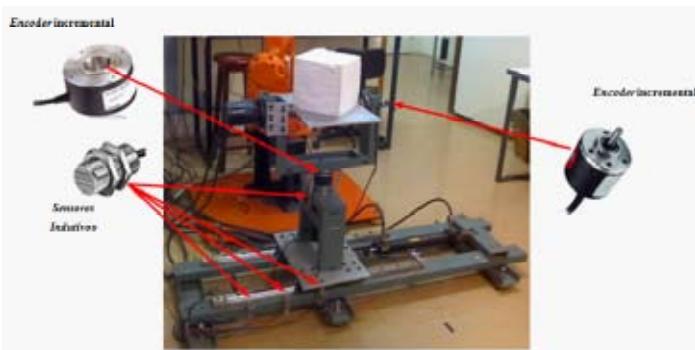


Figura 11.5: Sensores de Posicionamento

11.3.1 – *Encoders Incrementais*

Encoders incrementais são utilizados nas duas articulações rotativas da plataforma robótica, leitura da posição angular da junta à qual o transdutor foi acoplado. Um encoder incremental (figura 11.6) é um transdutor que converte um movimento angular ou linear em uma série de pulsos digitais elétricos. Esses pulsos gerados podem ser usados para determinar velocidade, taxa de aceleração, distância, rotação, posição ou direção.



Figura 11.5: *Encoder* de Movimento Angular.

Um *encoder* incremental fornece normalmente dois pulsos quadrados defasados em 90°, que são chamados usualmente de canal A e canal B. A leitura de apenas um canal fornece somente a velocidade, enquanto que a leitura dos dois canais fornece também o sentido do movimento. Outro sinal chamado de Z ou zero também está disponível e ele dá a posição absoluta “zero” do encoder. Este sinal é um pulso quadrado em que a fase e a largura são as mesmas do canal A.

11.3.2 – Sensores Indutivos

Os sensores indutivos são equipamentos eletrônicos capazes de detectar a aproximação de peças, componentes, elementos de máquinas, etc., em substituição as tradicionais chaves fim de curso. A detecção ocorre sem que haja o contato físico entre o acionador e o sensor, aumentando a vida útil do sensor por não possuir peças móveis sujeitas a desgastes mecânicos.

Os sensores Indutivos são sensores de proximidade, ou seja, geram um sinal de saída quando um objeto metálico (aço, alumínio, cobre, latão, etc.) entra na sua área de detecção, vindo de qualquer direção, sem que seja necessário o contato físico. Na plataforma hidráulica contém quatro sensores indutivos onde três sensores são utilizados para o posicionamento (início, meio e fim de curso do primeiro grau de liberdade do atuador linear)

e um sensor indutivo para ser usado como limite de curso do atuador rotacional (segundo grau de liberdade).

11.3.3 – Sistema de Acionamento

A tecnologia de acionamento escolhida foi hidráulica. A unidade hidráulica proposta é constituída dos seguintes elementos: reservatório hidráulico, onde há o condicionamento do óleo, sistema de filtragem, bomba hidráulica, eletrôs-válvula, mangueiras, conexões, manômetro, conforme mostrado na figura 11.6.



Figura 11.6: Unidade de Acionamento Hidráulico.

Em muitas aplicações industriais, a tarefa do robô é programada por aprendizagem, sem a necessidade de um modelo geométrico. Desse modo, sua trajetória é definida através de um conjunto de ângulos associados ao movimento angular de cada grau de liberdade do robô. Após a interpolação, estes ângulos atuarão como sinais de referência para o controlador de posição de cada junta que compara os sinais que provem dos sensores de posição delas (David e Rosário, 1998). Isso faz com que a configuração da mesa seja controlada a partir de um valor desejado, independente do movimento desejado e da carga transportada pela mesma.

No entanto, os valores das variáveis articulares utilizados como sinal de referência na malha de controle de posição das juntas, quando comparados com os valores das juntas, podem traduzir num erro, que aumenta com a sua velocidade de operação. Assim, a concepção de um controlador de posição para qualquer tipo de robô industrial exige o conhecimento da precisão cinemática de seu movimento.

11.4 – Modelagem Cinemática

Dada uma configuração de ângulos das juntas de um robô, o cálculo da posição e orientação da extremidade do manipulador é chamado de cinemática direta, e é sempre possível obter a solução do problema cinemático direto. Em contrapartida, o problema cinemático inverso é de solução um pouco mais complexa, dependendo das características do robô. Além disso, múltiplas soluções do problema e singularidades podem ocorrer. A figura 11.7 ilustra a relação entre as duas cinemáticas.

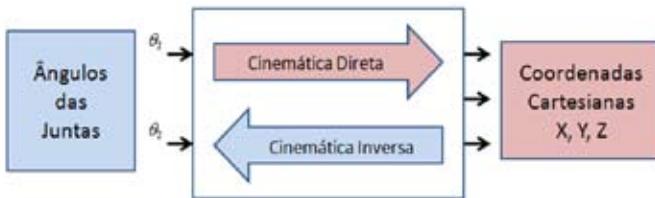


Figura 11.7: Cinemática Direta e Inversa de um Manipulador.

11.4.1 – Cinemática Direta

A obtenção do modelo cinemático direto pode ser realizada utilizando Vetores Locais (VL), ou pela convenção de Denavit-Hartenberg, a partir de coeficientes contidos em tabela correspondente a posição relativa dos diferentes graus de liberdade do dispositivo. Para esta análise é considerado a posição do centro da mesa até o referencial inercial, e o ponto P final da ferramenta em relação ao centro da mesa, θ_1 e θ_2 são os ângulos de rotação das duas juntas rotacionais (rotação da base e rotação da mesa). Na figura 11.8 são mostrados um modelo esquemático da mesa e uma indicação dos movimentos das juntas. O comprimento da haste de revolução será designado de L_h . Na tabela 11.1 são apresentados os parâmetros de DH associados ao dispositivo.

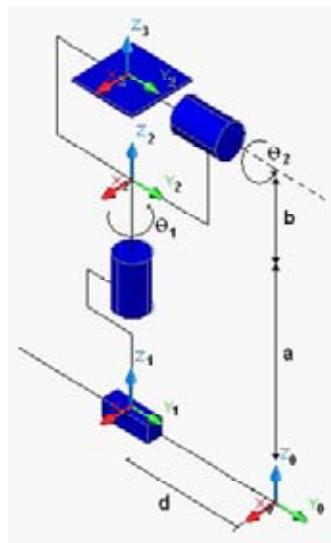


Figura 11.8: Plataforma 3 GL e Movimentos Relacionados.

Junta	α_j	d_j	θ_j	r_j
1	0	D	0	0
2	0	0	θ_1	a
3	θ_2	0	0	b

Tabela 11.1: Parâmetros de DH.

11.4.1.1 – Matrizes de Rotação

i) Devido à rotação em torno do eixo z (base)

$$T_{\theta_1} = \begin{pmatrix} \cos \theta_1 & -\sin \theta_1 & 0 \\ \sin \theta_1 & \cos \theta_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (11.1)$$

ii) Devido à rotação em torno do eixo x (rotação da mesa)

$$T_{\theta_2} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_2 & \sin \theta_2 \\ 0 & \sin \theta_2 & \cos \theta_2 \end{pmatrix} \quad (11.2)$$

11.4.1.2 – Vetores Locais

$$O_0O_1 = \begin{Bmatrix} 0 \\ 0 \\ R \end{Bmatrix} \quad O_1O_2 = \begin{Bmatrix} 0 \\ 0 \\ L_h \end{Bmatrix}$$

$$O_1 = O_0 + T_{\theta_1} * O_{11}$$

$$O_2 = O_1 + T_{\theta_1} * T_{\theta_2} * O_{12}$$

$$\text{Logo, } O_2 = O_0 + O_1 + = \begin{Bmatrix} -S\theta_1 * S\theta_2 * L_h \\ C\theta_1 * S\theta_2 * L_h \\ C\theta_2 * L_h \end{Bmatrix} \quad (11.3)$$

O₀: posicionamento da mesa em relação ao referencial inercial;

O₁: posicionamento da haste em relação ao centro da mesa (x, y);

Fator de Conversão do *Encoder* = X graus/ pulsos; e

Posição em graus = No de pulsos * Fator de Conversão do *Encoder*.

$$\theta_1 = \theta_{1\text{init}} + \theta_{1\text{lido}} \quad (11.4)$$

$$\theta_2 = \theta_{2\text{init}} + \theta_{2\text{lido}}$$

A partir da definição do ângulo fixo de posicionamento ou de um perfil de deslocamento da mesa (θ_2), é realizada a movimentação de uma haste fixa na mesa (L_h) a partir do movimento de rotação da base da mesa (θ_1) descrevendo um cone de revolução (θ_2 diferente de 0), ou um cilindro de revolução (θ_2 igual a 0, no caso da haste estar posicionada a uma distância d do centro da mesa).

11.4.2 – Cinemática Inversa

Considerando que a rotação da haste sobre a mesa (figura 11.9) realizará um cone no espaço para obtenção do modelo cinemático inverso, será levado em conta como parâmetros de entrada o raio do cone de revolução da haste (cilindro, no caso de mudança do centro) de revolução (x), posição atual da mesa ($\theta_{1\text{init}}$) e valor e sentido de deslocamento da haste em graus ($\theta_{1\text{desloc}}$) e velocidade de rotação da base da mesa. No caso da descrição de um cone espacial, o programa calculará, automaticamente, o ângulo de inclinação desejado para mesa ($a = \theta_2$), permitindo a rotação da base da mesa a partir da posição inicial até a posição final desejada ($\theta_{1\text{init}}$ e $\theta_{1\text{final}}$), ou seja:

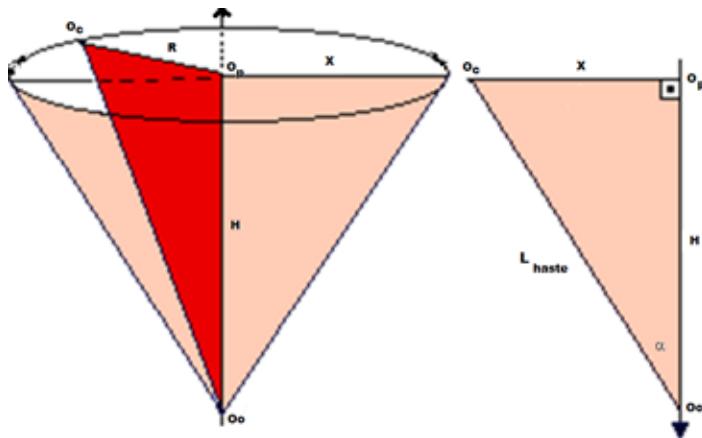


Figura 11.9: Cone de Rotação.

Parâmetros:

X: Raio do círculo de Revolução (distância da haste até o centro de rotação da mesa)

X_o : Distância da haste ao centro de revolução da mesa

L_h : Comprimento da haste fixa a base da mesa

$\theta_{1\text{desloc}}$: Deslocamento desejado da base da mesa em graus

$\theta_{1\text{init}}$: Posição inicial

$$\theta_2 = \text{ATAN2}\left(\frac{X}{\sqrt{Lh^2 - X^2}}\right) \quad (11.5)$$

$$\theta_{1\text{final}} = \theta_{1\text{init}} + \theta_{1\text{desloc}} \quad (11.6)$$

Estudo de Casos:

Caso 1: Cone Espacial com Haste no Centro de Rotação da Mesa: $X_0=0$

Caso 2: Cone Espacial com Haste deslocada do Centro de Rotação da Mesa: $X_0 \neq 0$

Caso 3: Cilindro de Revolução com Haste deslocada do Centro de Rotação da Mesa: $\theta_2=0$

11.5 – Estrutura de Acionamento e Controle

As tarefas realizadas pelos robôs baseiam-se no movimento independente de cada grau de liberdade, coordenado a partir de um gerador de trajetórias baseado no seu modelo cinemático. Na maioria dos casos, a programação das tarefas é realizada através do *modo aprendizagem*, onde ocorre a armazenagem da sequência de movimentos independentes de cada junta, até alcançar a posição final desejada. A partir do conhecimento destas posições articulares, é facilmente implementando um gerador de referências (perfil de velocidades) baseado nas características cinemáticas das juntas.

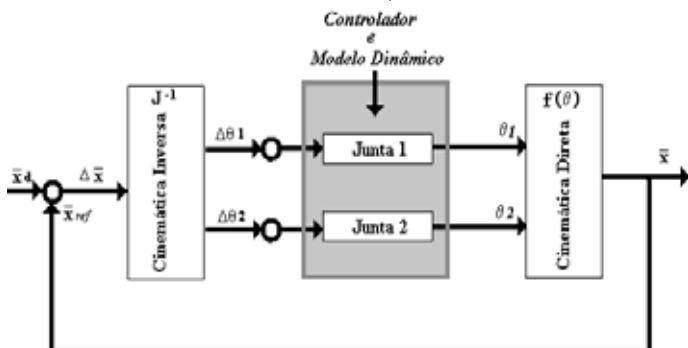


Figura 11.10: Estrutura de Controle.

O sistema completo de controle consiste na implementação da malha de controle em cascata para cada uma das juntas implementada de forma estruturada utilizando blocos do Simulink®. O conjunto de malhas de controle de posição, velocidade e torque podem fazer parte do modelo do sistema de acionamento e controle de uma junta robótica. O controle de posição do manipulador pode ser implementado através de realimentação para cada junta isolada, requerendo o modelo de cada junta. Todas as juntas do dispositivo possuem basicamente o mesmo modelo, com alteração apenas de parâmetros, facilitando a estruturação e implementação de um programa de simulação. Por fim, todas as juntas devem ser coordenadas

como foi apresentada na figura 11.10, para isso o modelo dinâmico da estrutura deve ser definido.

11.6 - Simulador Virtual

Para implementação física do dispositivo em estudo torna-se imprescindível a implementação de um simulador, onde poderá ser implementada a malha de controle de posição acoplada ao modelo completo para cada junta robótica de rotação, utilizando arquitetura aberta, de modo a facilmente serem implementadas diferentes estratégias de controle, para posterior simulação, análise e comparação de desempenhos.

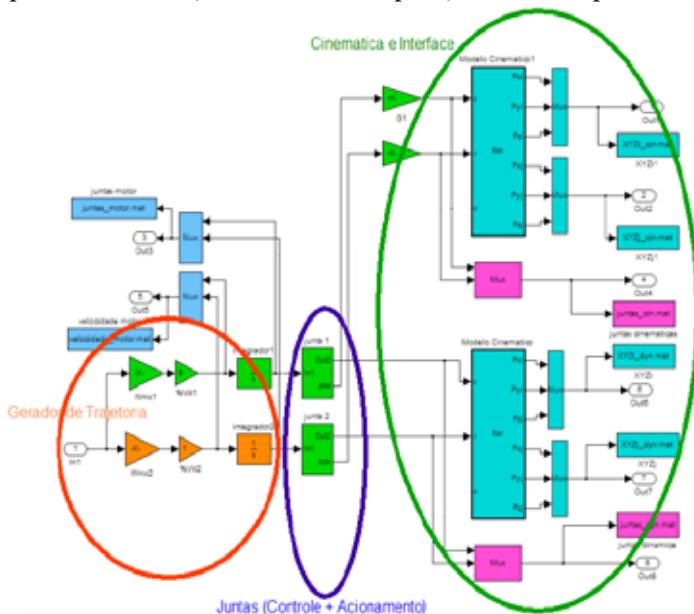


Figura 11.11: Simulador de uma mesa de 2 GL usando ambiente Simulink®

O simulador foi idealizado em arquitetura modular e aberta em ambiente Matlab/Simulink®, facilitando a criação e modificação das bibliotecas nele utilizadas, possibilitando,

também, a implementação de novos módulos que pode ser feita de maneira simples, isso permite uma evolução contínua do simulador e extensão de sua utilização para outros tipos de sistemas mecatrônicos.

Todas as simulações podem ser realizadas com ou sem perturbações e ainda com ou sem ruídos. Os seguintes módulos deverão ser implementados:

11.6.1 - Módulo Geração de Trajetórias

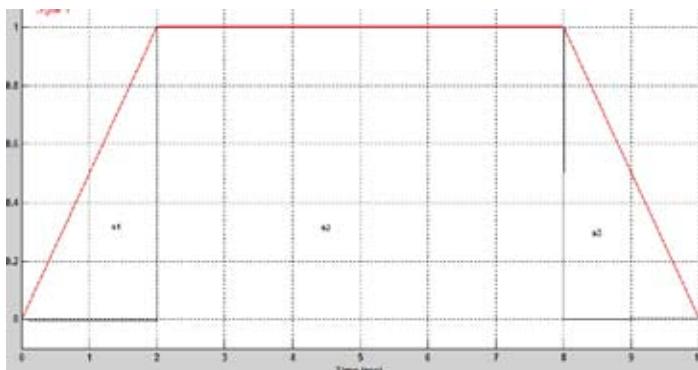


Figura 11.12: Sinal de referência de posição para cada junta.

11.6.2 - Módulo de Acionamento

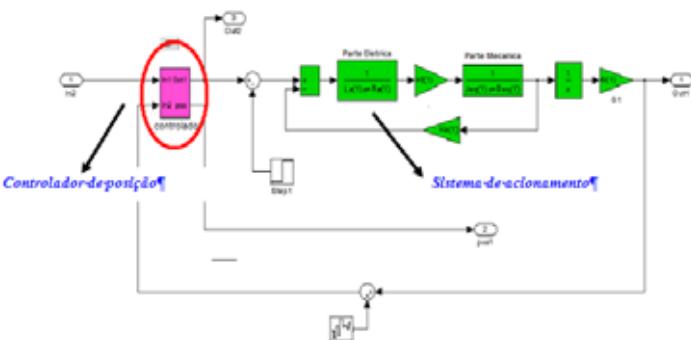


Figura 11.13: Malha de Controle de Posição de uma Junta.

11.6.3 - Módulo Modelo Cinemático

S-function, dentro do ambiente Matlab® integrados nos blocos do Simulink® (figura 11.14).

```
1 function [Px1,Py1,Pz1,Px2,Py2,Pz2] = Icm(v,w)
2 % Função modelo cinemático do Robô
3 % Visualização espacial da mesa (2 rotações e 1 translação)
4
5
6 w0 = 30; % ângulo de orientação da mesa
7 Teti= v; % rotação i
8 v=30;
9 Tet2= w + w0; % rotação 2
10
11 % Modelo Cinemático do robô
12 Cos_Teti = cos (Teti); % rotação em Z
13 Sin_Teti = sin (Teti);
14 Cos_Tet2 = cos (Tet2); % rotação em Y
15 Sin_Tet2 = sin (Tet2);
16
17 % dimensões da base na mesa (origem da referência em relações ao centro da mesa)
18 Xi = 100; % mm
19 Yi = 100; % mm
20 Lb = 100; % mm
21
22 % vetor de posição
23 Px1 = Xi - Lb * (Sin_Teti)*(Sin_Tet2);
24 Py1 = Yi + Lb * (Cos_Teti)*(Sin_Tet2);
25 Pz1 = (Cos_Tet2)* Lb;
26
27 Px2 = Xi - Lb * (Sin_Teti)*(Sin_Tet2);
28 Py2 = Yi + Lb * (Cos_Teti)*(Sin_Tet2);
29 Pz2 = (Cos_Tet2)* Lb;
```

Figura 11.14: Função S-function -
Matlab® implementada nos blocos do Simulink®

11.6.4 - Interface Gráfica

Assim, após as simulações realizadas no domínio do tempo através do simulador implementado em Simulink®, são obtidos arquivos de dados temporais correspondentes as variáveis de estudo (posição angular e cartesiana, velocidade, corrente, sinal de controle), que após tratamento conveniente, torna-se possível verificar importantes resultados disponíveis no menu da figura 11.15.



Figura 11.15: Menu do Simulador Robótico com 2 GL.

Para possibilitar a visualização dos movimentos em relação às coordenadas cartesianas das juntas, foi inserida uma haste de 100 mm de comprimento (entre os pontos A e B da figura 11.16) centralizada na base da mesa e inclinada de um ângulo 30° em relação à mesma.

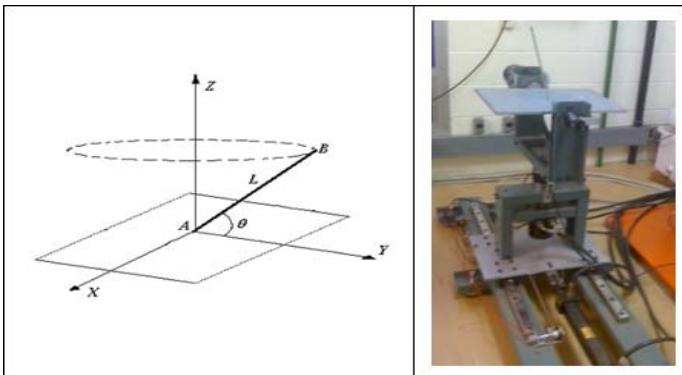


Figura 11.16: Visualização dos Movimentos com Haste no Centro do Dispositivo.

Para efeito de melhor visualizar o comportamento do sistema em estudo, as simulações apresentadas neste capítulo mostram a haste girando em torno do eixo Z, ou seja, movimentando apenas pelo atuador 2. Como a haste encontra-se no centro da mesa, posição $(0,0,0)$, a figura projetada será um cone. Deslocando-se a haste para outra posição $(P_x, P_y, 0)$ e fazer movimentar a junta a figura será um cilindro. À seguir são apresentados alguns resultados obtidos para estas condições de simulação:

11.6.4.1 – Velocidade do Motor

Pode-se verificar o movimento de cada junta separadamente ou as três juntas conjuntamente, sendo que a velocidade máxima de cada junta é escolhida como parâmetro de entrada no simulador.

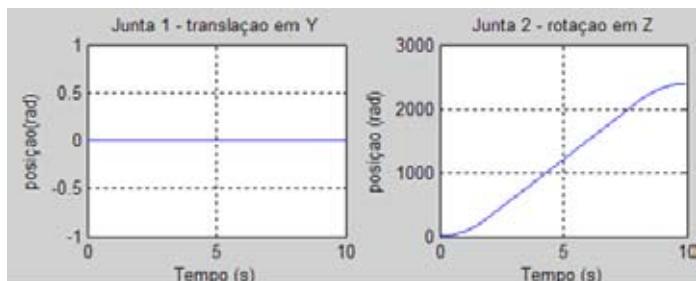


Figura 11.17: Perfil velocidade das juntas
(acionamento somente da junta 2).

11.6.4.2 – Deslocamento do motor

Nesse caso, tem-se o movimento considerando o deslocamento cinemático (figura 11.18), o dinâmico das juntas (figura 11.19), ou seja, com ou sem carga.

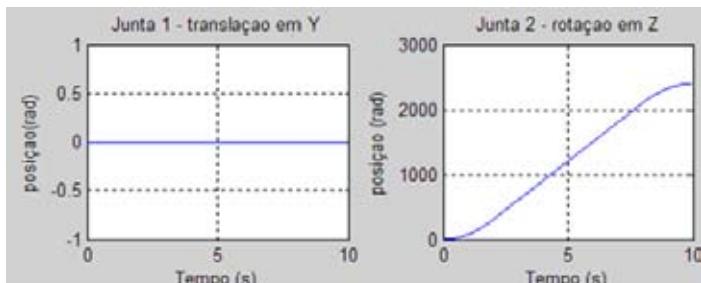


Figura 11.18: Deslocamento Cinemático
(acionamento somente da junta 2).

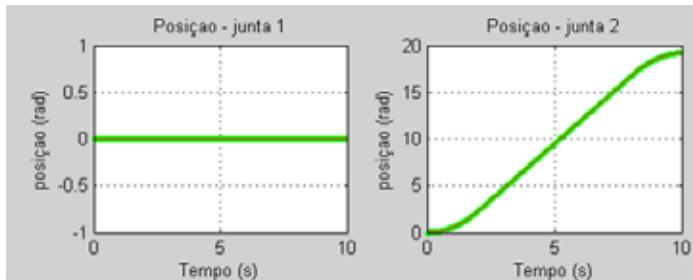


Figura 11.19: Deslocamento Dinâmico
(acionamento somente da junta 2).

11.6.4.3 - Sinal de Controle

O sinal de controle pode ser visualizado na figura 11.20, que apresenta a resposta no simulador, sem a introdução de um ruído.

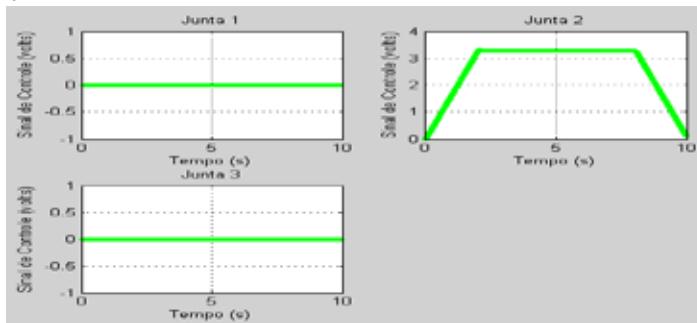


Figura 11.20: Sinal do Controle

11.6.4.4 - Movimento no Centro da Mesa

O movimento do centro da mesa diz respeito à haste posicionada sobre a mesma. A ideia é mostrar através da visualização espacial da posição apresentada pela haste em cada instante de tempo (intervalo de 10 segundos). A figura 11.21 apresenta a posição das coordenadas X, Y, e Z em relação ao tempo, enquanto a figura 11.22 apresenta o movimento espacial da movimentação

da haste fixada na mesa e o movimento espacial da haste fixa na mesa é apresentado na figura 11.23, onde podemos visualizar o movimento espacial da haste, permitindo verificar possíveis erros e perturbações de deslocamento no espaço cartesiano.

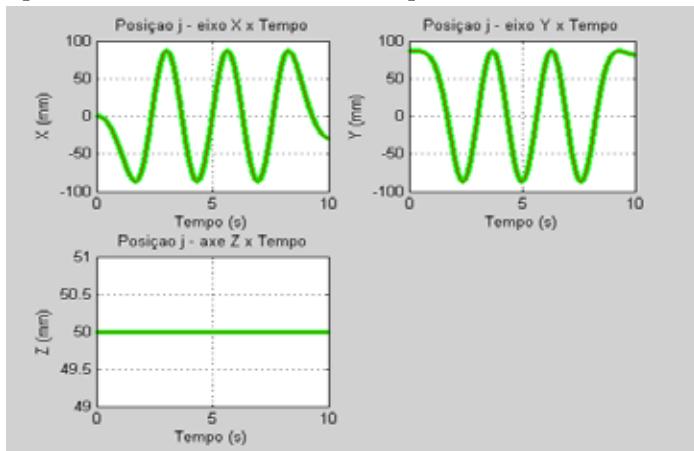


Figura 11.21: Movimento do centro da mesa
(acionamento somente da junta 2).

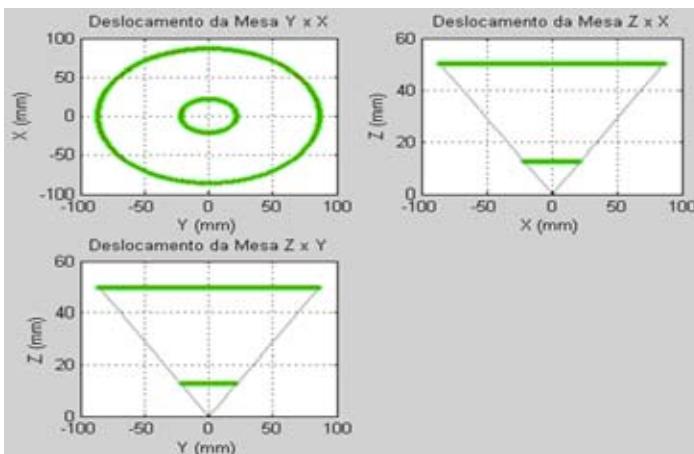


Figura 11.22: Vista dos eixos x, y e z da mesa em movimento (movimento espacial da haste).

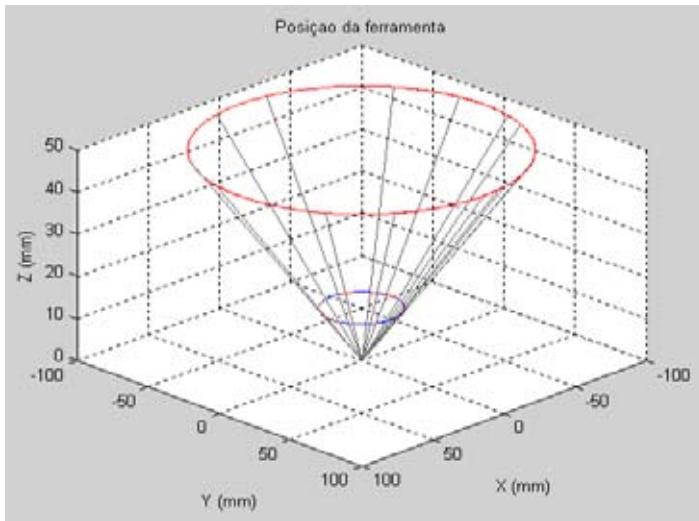


Figura 11.23: Posição espacial da haste -
acionamento somente do atuador 2

11.6.4.5 - Estudo do erro

A figura 11.24 apresenta os erros em relação à posição da mesa e a figura 11.25 apresenta o erro do atuador.

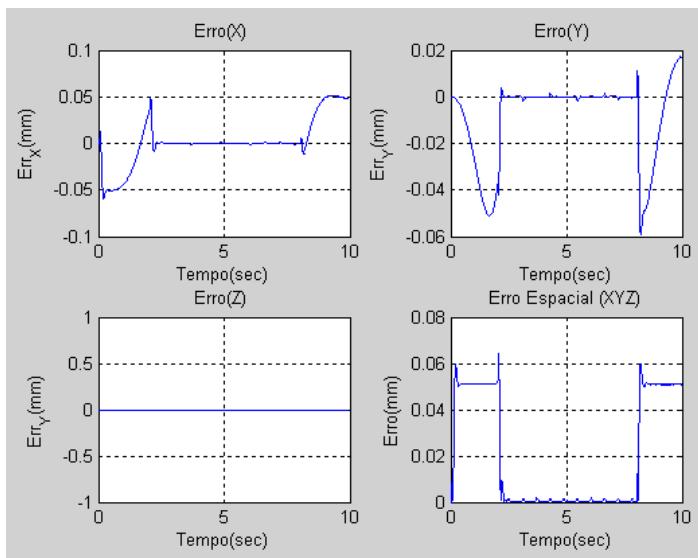


Figura 11.24: Erro de Posicionamento da Mesa.

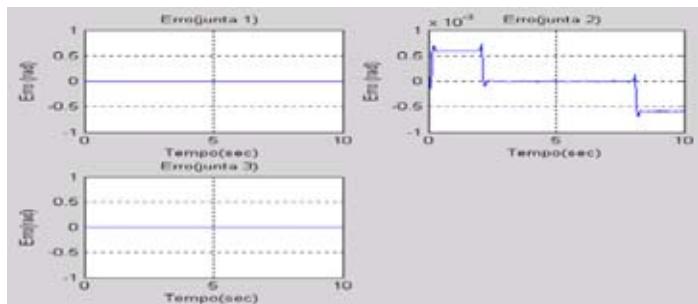


Figura 11.25: Erro dos Atuadores da Mesa.

11.7 – Implementação de Sistema Colaborativo em Automação

11.7.1 - Ambientes Colaborativos para Ensino e Pesquisa baseados em WEB

Através da evolução das Tecnologias da Informação e das Telecomunicações, a Internet passa a revolucionar a ciência, a economia e a sociedade. Desta forma, a utilização de ambientes educacionais com recursos computacionais permite um maior acesso a novos conhecimentos de maneira mais rápida, acarretando impactos sociais e novas áreas de pesquisa e desenvolvimento.

A utilização de ambientes ou plataformas de ensino deve conter informações que possibilitem que o aprendiz evolua conforme seu ritmo e flexibilidade (Traylot, Heer e Fiez, 2003). Para tanto, estes ambientes devem promover a integração de conhecimentos, inovação e experiências para a resolução de pequenos problemas, além de motivar e melhorar a visualização da continuidade do aprendizado. Visando tais aspectos a utilização de laboratórios virtuais e a Internet passam a ser aliados no processo de aprendizagem (Figura 11.26).



Figura 11.26: Aspectos de um Ambiente de Ensino e Pesquisa Virtual.

11.7.2 – Plataforma de Validação

Para validação da plataforma robótica usada para a operação virtual foi implementado um sistema de supervisão e controle desenvolvido em ambiente LabVIEW™, representando uma tarefa de execução de trabalho genérico complexo nas cinco faces de um cubo utilizando a plataforma hidráulica 3 GL descrita anteriormente, trabalhando cooperativamente com dois robôs industriais da ABB (IRB 140 e IRB 1400) e o estudo de acionamento e controle da plataforma 3 GL com interface de acionamento e controle implementada em ambiente LabVIEW™, a partir da movimentação de uma haste disposta na mesa.

A figura 11.27a apresenta uma peça na mesa para ser trabalhada cooperativamente junto com dois robôs industriais, que será posicionada pelo dispositivo mecatrônico para em seguida ser trabalhada pelos robôs industriais. A figura 11.27b apresenta uma haste colocada sob a mesa validação do modelo cinemático e interface de acionamento e controle deste dispositivo.

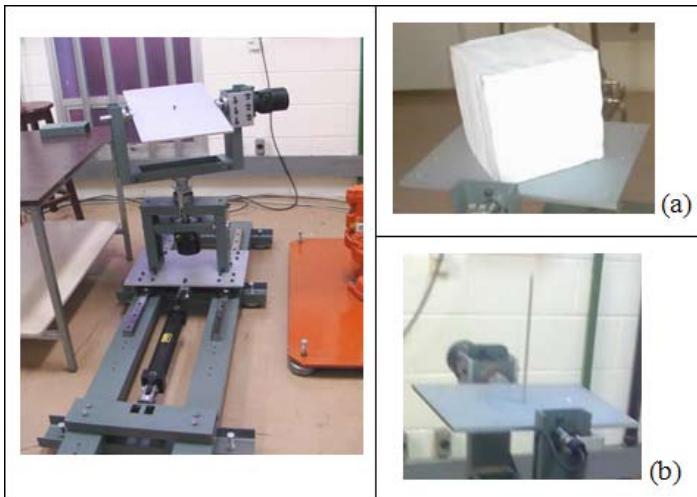


Figura 11.27: Plataforma de Posicionamento.

11.7.3 – Estrutura Proposta

Uma infraestrutura complexa para a realização de trabalhos experimentais na área de robótica nem sempre é possível de ser implementada em consequência dos altos custos envolvidos. A procura de soluções adequadas é constante, com o emprego de modelos matemáticos capazes de representarem parte de um sistema real sem perder sua generalidade, para validar experimentalmente novas estruturas de controle.

Dentro desse propósito, é implementado um ambiente estruturado de trabalho cooperativo envolvendo um trabalho de integração entre dois robôs e uma plataforma 3 GL de posicionamento, apresentando uma estrutura modular, hierarquizada e aberta, pode ser facilmente utilizado para integração de diferentes dispositivos mecatrônicos. A figura 11.28 apresenta a estruturação da tarefa de execução de trabalho genérico complexo, utilizando uma mesa indexada e um ou dois robôs industriais. A figura 11.29 apresenta imagens operativas das tarefas cooperativas executadas.

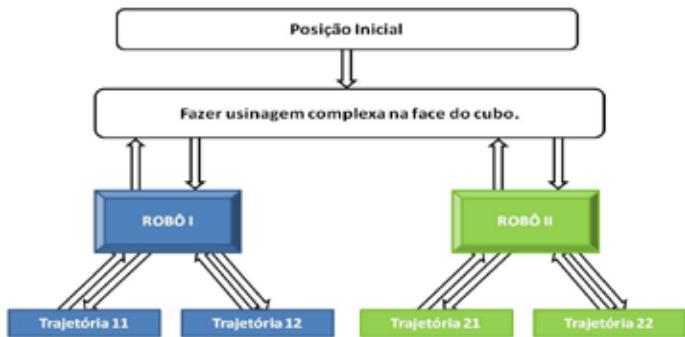


Figura 11.28: Estruturação de Tarefas.

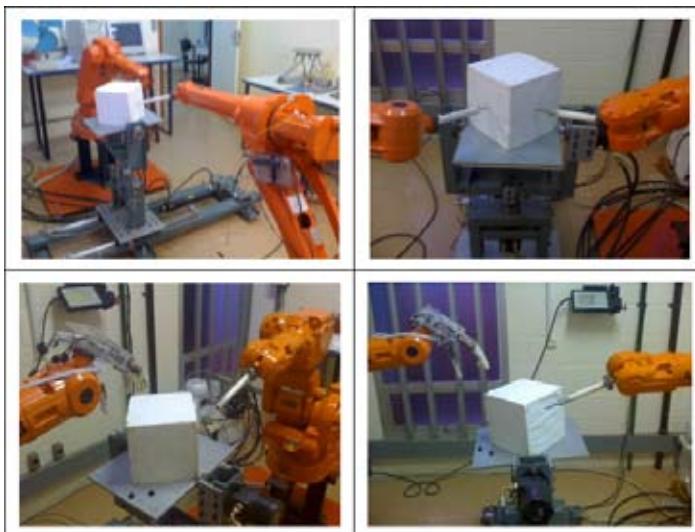


Figura 11.29: Célula Integrada de Trabalho.

A interface de acionamento e controle desta plataforma foi implementada a partir do aplicativo LabView™ e está disponibilizada através de aplicação web no seguinte

endereço: <http://143.106.9.98/web.htm> e a interface de visualização foi implementada a partir de uma IP-CAM, cuja visualização esta disponibilizada no seguinte endereço: <http://143.106.9.151/>. A estruturação proposta e variáveis utilizadas para implementação dos programas computacionais nos robôs ABB, CLP e sistema supervisório serão descritas a seguir.

11.7.4 – Implementação do Sistema Supervisório

11.7.4.1 – Descrição Funcional

O *software* supervisório tem como principal função coordenar as atividades de um sistema de robôs que trabalham cooperativamente, sendo implementado em linguagem Lab-VIEWTM, versão 7.1.

Ele é constituído de botões e chaves L/D (liga-desliga) nas cores vermelho, laranja e amarelo, tem a função de energizar e desenergizar funcionalidades do sistema.

- a) **Botões em amarelo:** disponibilizados na parte superior do sistema supervisório, são utilizados para posicionar os robôs em movimentos conjuntos ou coordenados, para início dos trabalhos;
- b) **Botões em vermelho:** disponibilizados nos lados, esquerdo e direito do supervisório, tem a função de posicionamento e alinhamento dos robôs individualmente, para que o sistema possa ser inicializado corretamente; e
- c) **Botões em laranja:** disponibilizados na parte inferior do sistema supervisório, são utilizados para posicionar a mesa, para início dos trabalhos.

O princípio de funcionamento obedecerá aos seguintes procedimentos:

- I. Após o alinhamento inicial da mesa e dos robôs, o sistema está em condições de executar os trabalhos coordenados e em

- conjunto por parte da mesa e por parte de ambos os robôs;
- II. Para iniciar os trabalhos coordenados basta pressionar o botão INICIA em amarelo, contido na parte inferior esquerda da tela; e
 - III. Para encerrar os trabalhos temos o botão STOP, que para as atividades do sistema coordenado. Caso o operador necessite reiniciar os trabalhos basta apertar o botão START que o sistema reinicia as atividades de onde parou.

11.7.4.2 – Protocolo de Comunicação e Topologia de Implementação

O meio físico (*hardware*) utilizado para a comunicação entre o sistema supervisório e os CLP's do robôs será uma comunicação serial RS485. Ele funciona obedecendo a um sistema Mestre e Escravo.

O sistema Mestre é um computador, plataforma Windows XP, com o *software* LabView™ como sistema Supervisório, e os escravos são os CLP's dos robôs. Este protocolo foi escolhido em virtude da existência de mais de um escravo (a comunicação RS232 não suporta mais de um escravo).

Um servidor *OPC SERVER* transfere e recebe as palavras de comando entre o mestre e os escravos automaticamente. Não se torna necessário a utilização ou implementação de recursos de comunicação para testes de consistência dos dados, como por exemplo: *Header*'s, algoritmos para testes de *Checksum*, polinômios de consistência, etc.

Para transferência das *palavras de controle*, foi utilizado um software servidor rodando no PC, sendo este um *OPC SERVER*, funcionando com os seguintes fabricantes (*Matricon, RS LINX, Aromat, etc.*). Isto possibilita que os problemas de comunicação se tornem transparentes para o projetista.

Foram utilizados *Bytes* para comunicação para comandos que podem ser de apenas Bits, porque a maioria dos Servidores *OPC SERVER* utiliza *Bytes* que são utilizados como Bits no sistema supervisório.

Essas palavras de comando são trocadas entre os *CLP's* e os robôs e o sistema supervisório, elas são trocadas de forma contínua, gerenciadas pelo sistema *OPC SERVER*. No anexo I segue a relação das entradas e saídas do sistema e a lógica de comando supervisor. Cada ***palavra de controle*** é composta de *1 byte* conforme mostra a tabela 11.2.

Byte	Comando Realizado
0	Início
1	Mesa
2	Movimento de Translação da Mesa
3	Rotação R1 da Mesa (Face 1 e 3)
4	Rotação R1 da Mesa (Face 2 e 4)
5	Rotação R2 da Mesa (inclinação) - Robô 1
6	Rotação R2 da Mesa (inclinação) - Robô 2
7	Init_Rob
8	Posição Inicial - Robô (R1) - Traj11
9	Posição Inicial - Robô (R2) – Traj21
10	Posicionamento Cooperativo Robôs (R1 e R2)
11	Task1_Rob
12	Traj12: Trajetória Robô 1 - Face 1 (2)
13	Traj22: Trajetória Robô 2 - Face 1 (4)
14	Tarefa Cooperativa Robôs (R1 e R2)
15	Task2_Rob
16	Traj13: Trajetória Robô 1 - Mesa Inclinada
17	Traj23: Trajetória Robô 2 - Mesa Inclinada
18	Tarefa Seqüencial Robôs (R1 e R2)

Tabela 11.2: Palavras de Controle.

11.7.4.3 – GRAFCET Funcional

As figuras 11.30 e 11.31 apresentam o GRAFCET Tecnológico correspondente à aplicação.

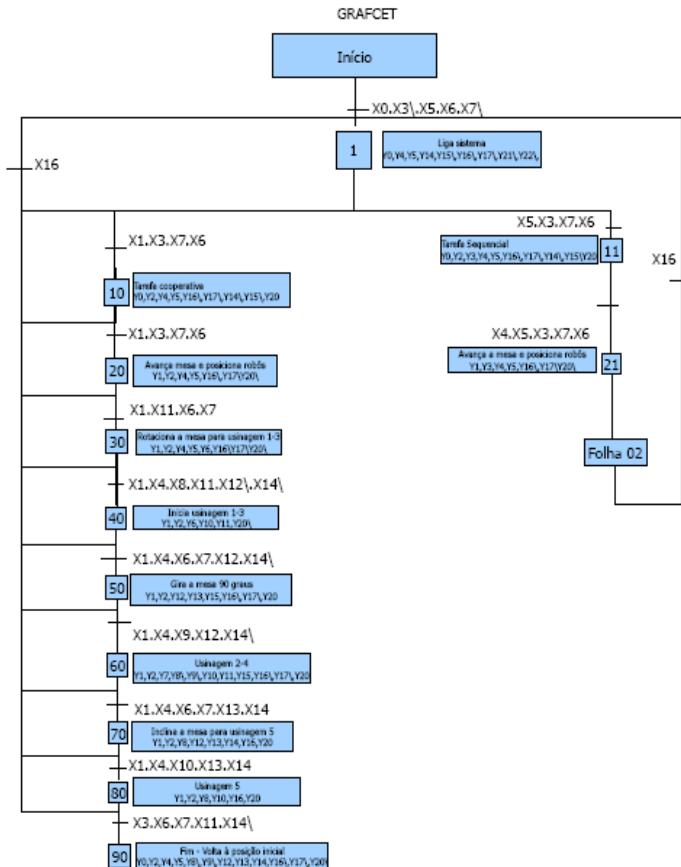


Figura 11.30: GRAFCET Tecnológico (parte 1).

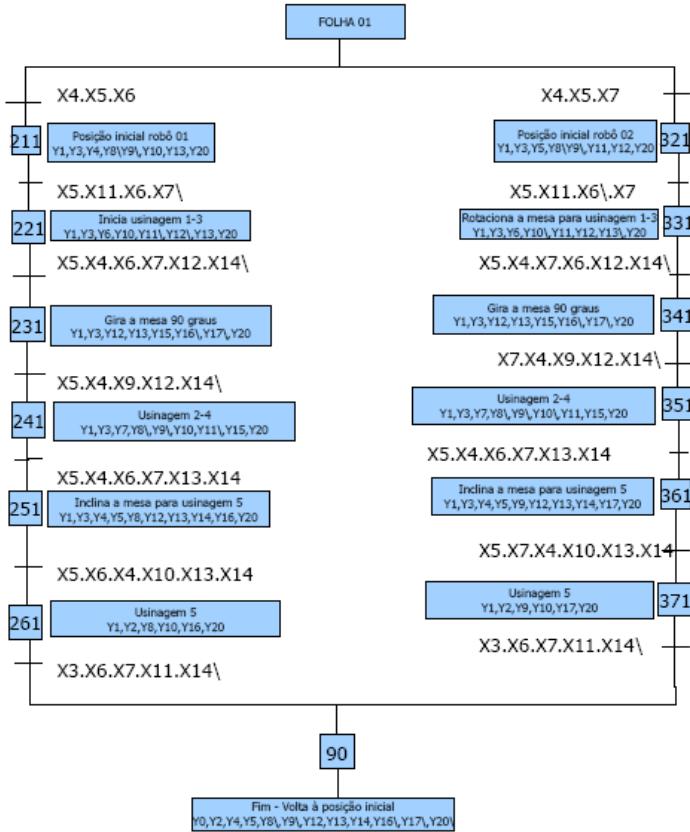


Figura 11.31: GRAFCET Tecnológico (parte 2).

11.7.4.4 – Definição de Variáveis do Sistema

a) Sub-rotina MESA

Nº	Variáveis	Especificação
1	S0	Início do Programa
2	SF	Final do Programa
3	S10	Task2 (inclinação da mesa)
4	S11	Movimento de Translação da Mesa
5	S12	Rotação R1 Mesa (Face 1 - 3)
6	S13	Rotação R1 Mesa (Face 2 - 4)
7	S14	Rotação R2 Mesa (inclinação) – Robô 1
8	S15	Rotação R2 Mesa (inclinação) – Robô 2
9	S16	Traj13: Trajetória Robô 1 – Mesa Inclinada
10	S17	Traj23: Trajetória Robô 2 – Mesa Inclinada

b) Sub-rotina Movimentação dos Robôs

Nº	Variáveis	Especificação
1	S21	Pos_Inic_ – Robô (R1) - Traj11
2	S22	Pos_Inic_ – Robô (R2) - Traj21
3	S23	Posicionamento Cooperativo Robôs (R1 e R2)
4	S31	Tarefa Sequencial Robô (R1)
5	S31	Tarefa Seqüencial Robô (R2)
6	S12	Traj13: Trajetória Robô 1 – Mesa Inclinada
7	S13	Traj23: Trajetória Robô 2 – Mesa Inclinada
8	S24	Task_1_Usinagem de superfície
9	S25	Traj12: Trajetória Robô 1 – Face 1 (2)
10	S26	Traj22: Trajetória Robô 2 – Face 3 (4)
11	S27	Tarefa Cooperativa Robôs de Usinagem (R1 e R2)

c) Lógica de comando do supervisor associados aos subprogramas da CLP

• Blocos de Estágios

Nº	Variáveis	Especificação
1	S21	Pos_Inic_ – Robô (R1) - Traj11
2	S22	Pos_Inic_ – Robô (R2) - Traj21
3	S23	Posicionamento Cooperativo Robôs (R1 e R2)
4	S31	Tarefa Sequencial Robô (R1)
5	S31	Tarefa Sequencial Robô (R2)
6	S12	Traj13: Trajetória Robô 1 – Mesa Inclinada
7	S13	Traj23: Trajetória Robô 2 – Mesa Inclinada
8	S24	Task_1 (Usinagem de superfície)
9	S25	Traj12: Trajetória Robô 1 – Face 1 (2)
10	S26	Traj22: Trajetória Robô 2 – Face 3 (4)
11	S27	Tarefa Cooperativa dos Robôs de Usinagem (R1 e R2)
12	S5	Task_2 (Inclinação da Mesa)
13	S51	Traj13: Trajetória Robô 1 – Mesa Inclinada
14	S52	Traj23: Trajetória Robô 2 – Mesa Inclinada
15	S53	Tarefa Sequencial dos Robôs (R1 e R2)

• Variáveis Auxiliares

Nº	Variáveis	Especificação
1	C1	Lógica de modo Mesa e Robô
2	C2	Lógica de modo Mesa e Robô
3	C3	Supervisor
4	C4	Supervisor
5	C5	Supervisor
6	C6	Supervisor
7	C7	Switch ON/OFF

- **Variáveis do CLP**

Nº	Variáveis	Mnemônico	Especificação
1	X0	L_D	Chave Liga-desliga
2	X1	Coop	Modo Cooperativo

- e) **Estruturação Lógica**

- **Início do Programa – Seleção Modo de Movimentação da Mesa e Robô**

C1	C2	Valor	S	Especificação
0	0	0	S0	Menu de Início de Programa
0	1	1	S1	Modo Movimenta Mesa
1	0	2	S2	Modo Movimenta Robô

- **Modo de Movimentação da Mesa**

C1	C2	C3	C4	C5	Valor	S	Especificação
0	1	0	0	0	0	S10	Movimentação da Mesa
0	1	1	1	1	7	S11	Translação da Mesa
0	1	0	0	1	4	S12	Rotação Mesa (1-3)
0	1	0	1	0	2	S13	Rotação da Mesa (1-4)
0	1	1	0	1	5	S14	Inclinação da Mesa para Robô 01
0	1	1	1	0	3	S15	Inclinação da Mesa para Robô 02
0	0	X	X	X	X	S0	Menu de Início de Programa

- Modo de Movimentação do Robô**

C1	C2	C3	C4	C5	C6	Valor	S	Lógica de Controle
1	0	0	0	X	X	0	S2	MODO – Movimenta Robô
1	0	0	1	X	X	2	S3	INICIALIZAÇÃO DO ROBÔ
1	0	0	1	0	0	2	S31	Posição Inicial – Robô (R1) - Traj1
1	0	0	1	1	1	1	S32	Posição Inicial – Robô (R2) - Traj2
1	0	0	1	1	1	3	S33	Posicionamento Cooperativo Robôs (R1 e R2)
1	0	1	0	X	X	1	S4	Task1 (Faces)
1	0	1	0	0	0	2	S41	Posição Inicial – Robô (R1) - Traj1
1	0	1	0	1	1	1	S42	Posição Inicial – Robô (R2) - Traj2
1	0	1	0	1	1	3	S43	Posicionamento Cooperativo Robôs (R1 e R2)
1	0	1	1	X	X	3	S5	Task2 (Mesa Inclinada)
1	0	1	1	0	0	2	S51	Traj13: Trajetória Robô 1 – Mesa Inclinada
1	0	1	1	1	1	1	S52	Traj23: Trajetória Robô 2 – Mesa Inclinada
1	0	1	1	1	1	3	S53	Tarefa Sequencial Robôs (R1 e R2)
0	0	X	X	X	X	X	S0	Menu de Início de Programa

f) Entradas do CLP e Entradas (Sensores)

Variáveis	Mnemônico	Descrição
X0	LD	Liga-Desliga
X1	COOP	Modo cooperação
X2	ST	Start
X3	CIL_AV	Mesa recuada
X4	CIL_REC	Mesa avançada
X5	SEQ	Modo sequencial
X6	PINI_R1	Posição inicial Robô 1
X7	PINI_R2	Posição inicial Robô 2
X8	F1_3	Usinagem 1-3
X9	F2_4	Usinagem 2-4
X10	F5	Usinagem 5
X11	ROTM1_3	Rotação mesa 1-3
X12	ROTM1_4	Rotação mesa 2-4
X13	ROTM5	Rotação mesa 5
x14	INCL_MESA	Inclinação da mesa
x15	STOP	Interrupção do programa
X16	EMER	Emergência

e) Saídas do CLP

Variáveis	Mnemônico	Descrição
y0	L_M_R	Led de mesa recuada
y1	L_M_A	Lede de mesa avançada
y2	L_COOP	Led tarefa coop.
y3	L_SEQ	Led tarefa seq.
Y4	L_POS_I_R1	Led pos. Inic. Robô 01
Y5	L_POS_I_R2	Led pos. Inic. Robô 02
Y6	L_1-3	Led trajetória face 1-3
Y7	L1-4	Led trajetória face 2-4
Y8	L_I_R1	Led trajetória mesa inclinada Robô 01
Y9	L_I_R2	Led trajetória mesa inclinada Robô 02

Variáveis	Mnemônico	Descrição
Y10	L_RB_OP_01	Led robô 01 em operação
Y11	L_RB_OP_02	Led robô 02 em operação
Y12	L_RB_DE_01	Led robô 01 em descanso
Y13	L_RB_DE_02	Led Robô 02 em descanso
Y14	L_R_M_1-3	Rot. Mesa F 1-3
Y15	L_R_M_2-4	Rot. Mesa F 2-4
Y16	L_I_M_R1	Inclinação mesa R1
Y17	L_I_M_R2	Inclinação mesa R2
Y20	CIL_AV_REC	Acionamento do cilindro de recuo avanço da mesa
Y21	MR	Motor rotação
Y22	MI	Motor Inclinação

f) Entradas do Robô

X4	X5	X6	X7	Valor	Comando
1	0	0	0	16	Modo Robô
1	1	1	0	112	Posição Inicial do Robô R1
1	1	0	1	196	Trajetória Robô 1 – Face 1 (2)
1	1	1	1	240	Trajetória Robô 1 – Mesa Inclinada
0	0	1	0	64	Posição Inicial do Robô R2
0	0	0	1	128	Trajetória Robô 2 – Face 3 (4)
0	0	1	1	192	Trajetória Robô 2 – Mesa Inclinada

g) Saídas do robô

Y3	Y4	Y5	Valor	Comando
1	0	0	8	Modo Robô
1	1	1	56	Posição inicial do Robô 01
1	1	0	24	Traj. Robô 01 para usinar face 1-3
1	0	1	40	Mesa inclinada para robô 01
0	1	1	48	Posição inicial do Robô 02
0	1	0	16	Traj. Robô 02 para usinar face 2-4
0	0	1	32	Mesa inclinada para robô 02

h) Lógica de Entradas do CLP

X0	X1	X2	X3	Valor	Comando
1	0	0	0	1	Operação Mesa
1	1	0	0	3	Pos. Inic. Mesa (na translação)
1	0	1	0	5	Rot. Mesa (1-3)
1	0	0	1	9	Rot. Mesa (2-4)
1	1	1	0	7	Inclinação da mesa para robô 01
1	1	0	1	11	Inclinação da mesa para robô 02

i) Lógica de entrada dos Robôs

X4	X5	X6	X7	Valor	Comando
1	0	0	0	16	Modo Robô
1	1	1	0	112	Posição inicial do Robô 01
1	1	0	1	196	Traj. Robô 01 para usinar face 1-3
1	1	1	1	240	Mesa inclinada para robô 01
0	0	1	0	64	Posição inicial do Robô 02
0	0	0	1	128	Traj. Robô 02 para usinar face 2-4
0	0	1	1	192	Mesa inclinada para robô 02

11.7.4.5 – Programas Implementados nos Robôs ABB

- Robô Industrial ABB IRB 140

PROC main()

! INICIALIZAÇÃO DO ROBO

! Posicao Inicial Robo R1

IF DI10_3=1 AND DI10_4=1 AND DI10_5=1 THEN

pOS_INIT_R1;

Set DO10_4;

Set DO10_5;

Set DO10_6;

WaitTime 2;

Reset DO10_4;

Reset DO10_5;

Reset DO10_6;

ENDIF

! Trajetoria Robo 1 - Face 1(2)

IF DI10_4=1 AND DI10_5=0 AND DI10_3=1 THEN

R1FACE1;

R1FACE2;

Set DO10_4;

Set DO10_5;

```

Set DO10_7;
WaitTime 2;
Reset DO10_4;
Reset DO10_5;
Reset DO10_7;
ENDIF

! Trajetoria Robo 1 - Mesa Inclinada
IF DI10_5=1 AND DI10_3=1 AND DI10_4=0 THEN
R1_MES_INC;
POS_INIT_R1;
Set DO10_4;
Set DO10_5;
Set DO10_6;
Set DO10_7;
WaitTime 2;
Reset DO10_4;
Reset DO10_5;
Reset DO10_6;
Reset DO10_7;
ENDIF
ENDPROC
ENDMODULE

```

- **Robô Industrial ABB IRB 1400**

```

PROC main()
!    OBS.: WObj:=sgua3 e 4 são as duas faces do quadra-
do;
!          WObj:=sgua5 é a face superior quando inclina
o quadrado
! INICIALIZAÇÃO DO ROBO
! Posicao Inicial Robo R2
IF DI10_3=0 AND DI10_4=1 AND DI10_5=1
THEN
pOS_INIT_R2;
Set DO10_6;

```

```

reg1 :=0;
WaitTime 2;
Reset DO10_6;
ENDIF
! Trajetoria Robo 2 - Face 3 e 4
IF DI10_4=1 AND DI10_5=0 AND DI10_3=0 THEN
R2FACE3;
R2FACE4;
Set DO10_7;
reg1 :=0;
WaitTime 2;
Reset DO10_7;
ENDIF
! Trajetoria Robo 2 - Mesa Inclinada
IF DI10_5=1 AND DI10_3=0 AND DI10_4=0 THEN
R2_MES_INC;
Set DO10_6;
Set DO10_7;
reg1 :=0;
WaitTime 2;
Reset DO10_6;
Reset DO10_7;
ENDIF
ENDPROC
ENDMODULE

```

11.7.5 - Telas Implementadas em Ambiente LabViewTM

A figura 11.32 apresenta as telas implementadas em LabViewTM das tarefas cooperativas executadas.



Figura 11.32: Proposta de página HTML - Painel de Controle Implementado.

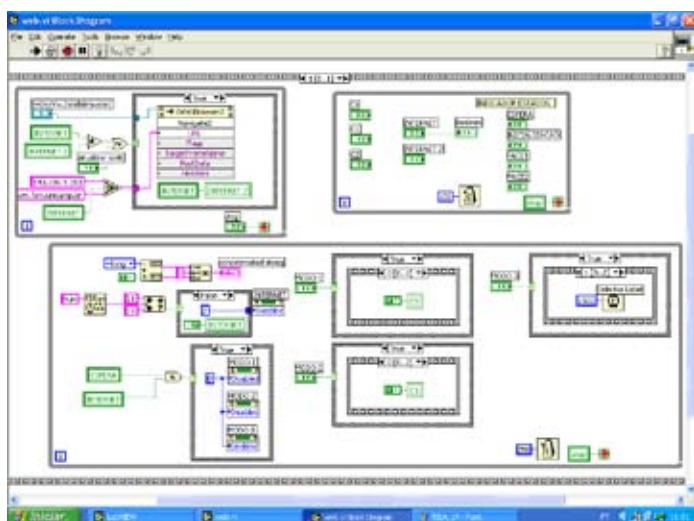


Figura 11.33: Tela típica de Implementação.

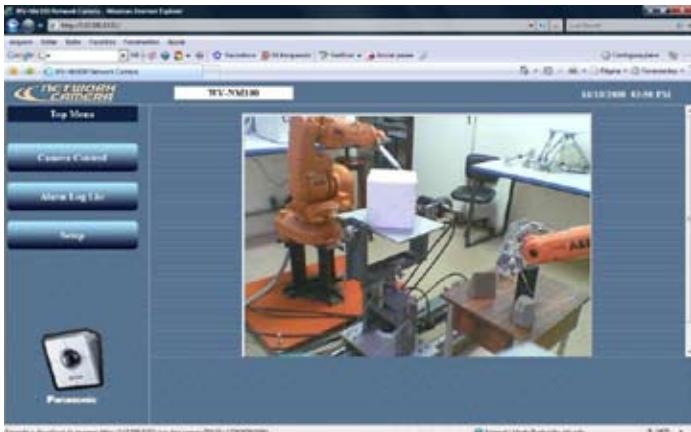


Figura 11.34: Tela de Visualização de Câmera CCD

WEBCAM com IP fixo.

11.8 – Sistema de Supervisão e Controle da Plataforma com 3 GL

Dentro desse propósito, a implementação de um simulador através de blocos estruturados, apresentando uma estrutura modular, hierarquizada e aberta, pode ser facilmente utilizada para as simulações de diferentes dispositivos mecatrônicos. A figura 11.32 apresenta esquematicamente o simulador implementado. Este simulador é constituído dos seguintes módulos:

- Geração de sinais de referências de trajetórias das juntas;
- Modelo cinemático direto e inverso;
- Acionamento e controle;
- Interface gráfica de modo a permitir a visualização dos resultados dos movimentos obtidos através de trajetórias de referências.

A partir da subseção 4.3.5 será apresentada as telas de inicialização, posicionamento, orientação, calibração e controle.

No Laboratório de Automação Integrada e Robótica da Faculdade de Engenharia Mecânica da UNICAMP foi implementada uma plataforma de posicionamento com três graus de liberdade (robô PRR) para trabalhar cooperativamente com dois robôs industriais (IRB 140 e IRB 1400 da AB^{BTM}), com o objetivo de permitir o trabalho cooperativo de robôs manipuladores convencionais em operações de usinagem e soldagem de dispositivos mecânicos complexos que necessitam de mais graus de liberdade para realizarem trajetórias complexas (figura 4.9).



Figura 11.32: Célula Integrada de Manufatura.

Nas próximas subseções será descrito o *software implementado em LabViewTM* para acionamento e controle dos três atuadores hidráulicos de uma plataforma de movimentação capaz de posicionamento de uma base e orientação de uma mesa com 2 GL.

11.8.1 - Descrição da Parte Operativa

Esta plataforma é constituída de atuadores hidráulicos com sensores de posicionamento. O primeiro grau de liberdade possui um cilindro de posicionamento linear para movimentação da base em uma determinada direção (movimento translacional) e sensores indutivos de final de curso e posicionamento central da mesa. Dois graus de liberdade são responsáveis pela orientação espacial da mesma (robô RR), através da movimentação angular de motores hidráulicos rotativos (movimento de rotação) constituídos de sensores de rotação (*encoders incrementais*). Na base da mesa está prevista a colocação de uma haste de calibração, cujas dimensões (comprimento) serão disponibilizadas como parâmetro variável da tela de inicialização.

O *software LabViewTM* pode ser usado para controle e supervisão. O controle das juntas robóticas é realizado através de uma interface A/D que comanda os motores das juntas. Um programa de supervisão e controle, residente em um computador é responsável pelo gerenciamento e controle das informações provenientes de sensores e atuadores do sistema.

Uma interface de visualização implementada em ambiente Windows foi desenvolvida telas de supervisão que coletem as informações dos sensores, dos programas de tratamento matemático (modelagem cinemática da mesa), as informações sobre o sistema (velocidade, parâmetros do controlador, números de pontos da trajetória, etc.), da inicialização e da calibração automatizada.

A interface de aquisição de dados utilizada neste trabalho foi a PCI 9112 da *ADLINK Technology*. Esta interface

recebe as informações provenientes dos *encoders* e do potenciômetro para determinação da posição atual das juntas constituintes da mesa. Estas informações são comparadas com os valores de referência, logo após esta interface calcula o algoritmo de controle envia para a saída do controlador enviando estas informações (posição, velocidade) aos acionadores de cada grau de liberdade da mesa (figura 11.33).

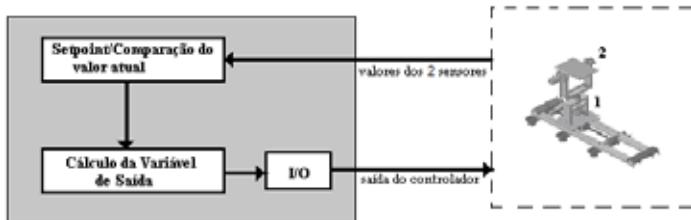


Figura 11.33: Esquema do Sistema Controlado.

Todos os valores de referência são calculados em tempo real, mesmo a cinemática inversa com as altas não linearidades. Outros tipos de sensores podem ser incluídos na bancada ou simplesmente através de uma VI (*Virtual Interface*) do software LabView™. Normalmente, o projeto de uma VI, deverá conter os seguintes elementos de base:

- a) **Painel Frontal:** É uma interface interativa entre o usuário e o programa. É onde o usuário entra com os dados usando o mouse ou o teclado, e então vê os resultados na tela do computador. Isto significa que o Painel Frontal é uma janela de execução, onde são encontradas as entradas de informação (*control*) e as saídas de informação (*indicator*);

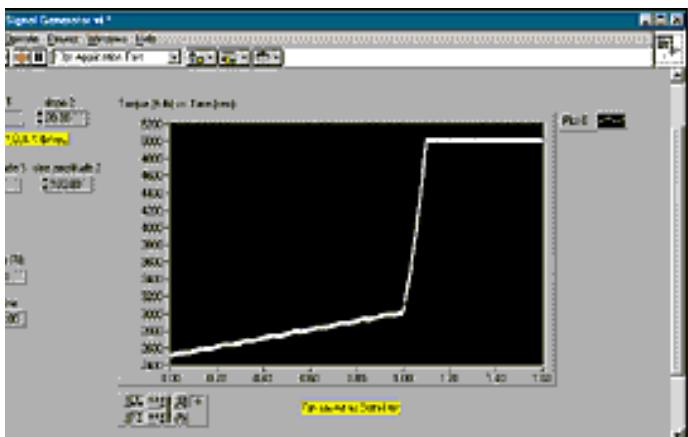


Figura 11.34: Tela - Painel Frontal: Aquisição de um Sinal.

b) **Diagrama de blocos:** É a representação de um programa ou algoritmo. É onde o programador cria o seu programa. O programa consiste de menus que definem a movimentação automática ou manual da mesa. A movimentação manual é determinada apenas indicando quais motores devem ser acionados. A movimentação automática é controlada pelo programa desenvolvido para tal fim;

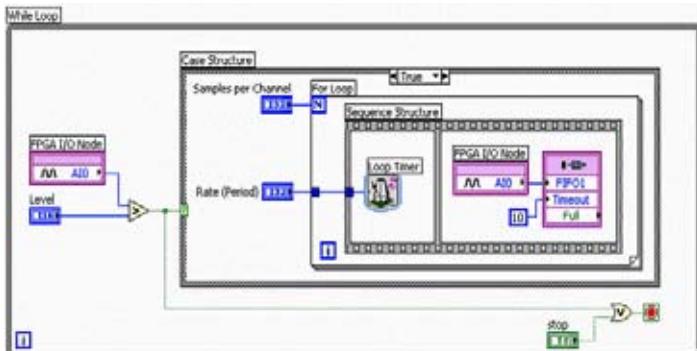


Figura 11.35: Diagrama de Blocos: Entrada analógica com Intelligent DAQ e LabVIEW FPGA.

- c) Tela principal: A ser utilizada para o controle direto da plataforma, foi elaborado um programa em linguagem visual. Esse programa de interface se comunica com a placa de interface com o *hardware*, a qual controla diretamente os motores;
- d) Tela contendo os diagramas de conexão de cada bloco: Os diagramas são as conexões de cada bloco, e apresentam características específicas ao projeto implementado, de acordo com os elementos que serão implementados na VI, as conexões são realizadas neste painel.

11.8.2 - Sistema de Supervisão e Controle

O sistema de supervisão e controle foi implementado em linguagem LabVIEW™, muito utilizado industrialmente em projetos de dispositivos relacionados as áreas de medição e controle. Podem utilizar LabVIEW™ para criar aplicações personalizadas que rodam em plataformas NI (*National Instruments*) com E/S reconfiguráveis baseadas em FPGAs. Juntos, o LabVIEW™ FPGA e o *hardware* NI para E/S permitem a criação de uma plataforma flexível para o desenvolvimento de sistemas sofisticados que antes somente eram possíveis com *hardware* projetado de forma dedicada.

11.8.2.1 - Descrição do Sistema Operativo

Para implementação do Sistema de Supervisão e Controle da Plataforma Hidráulica foi utilizado a interface de comunicação *PCI-9112 16-CH 12-Bit 110 kS/s Multi-Function DAQ Card/ Low-Profile DAQ Card da National Instrumentation™* (figura 11.36), constituída de um módulo com oito entradas analógicas (A/D) para leitura de dois *encoders* incrementais das juntas de movimentação rotativa, um módulo de oito saídas analógicas (D/A) para acionamento dos cilindros rotativos (2) e avanço e recuo do cilindro de movimentação linear, um módulo de oito entradas e oito saídas digitais para leitura

dos sensores de posicionamento do cilindro linear e sensores de final de curso, e acionamento de chaves (por exemplo, chaves de início e final de operação) e acionamento.

11.8.2.2 - Interface de Controle

A interface com o usuário foi desenvolvida utilizando o *software* LabVIEW™ compatível com esta interface, possibilitando o monitoramento e controle de informações por dois modos de funcionamento: aprendizagem e leitura do arquivo de dados. Assim são permitidos o aprendizado e gravação de trajetórias obtidas a partir da modelagem cinemática da plataforma e trajetórias em relação à ferramenta terminal dos robôs industriais utilizados, armazenamento de informações, leitura dos sensores de posição (*encoders* incrementais) através de interface A/D e acionamento do atuador linear hidráulicos para posicionamento da plataforma (avanço e recuo) e atuadores hidráulicos rotativos através da interface de saída digital, após tratamento do regulador PID de cada junta, interface de entrada digital responsável pela aquisição das informações provenientes dos sensores externos (por exemplo, chaves de início e final de operação).

Assim, todo o gerenciamento dessas informações é realizado através de um programa computacional de supervisão e controle residente em um computador PC, onde foi implementada uma interface de visualização, a partir do desenvolvimento de telas de monitoramento das informações dos sensores (final de curso, segurança, chaves lógicas, etc.), e programas de tratamento matemático (modelagem cinemática do dispositivo), informações sobre o status do sistema (porcentagem de velocidade, parâmetros do regulador PID, número de pontos da trajetória, etc.), inicialização e calibração automatizada.



Figura 11.36: Interface de Comunicação.

11.8.3 - Telas Implementadas

11.8.3.1 - Tela de Inicialização

A tela de inicialização do programa permite que o aplicativo computacional funcione em modo manual (controlado através de botões dedicados (calibração e controle) no painel de comando) ou modo computador, através de *menus* acionados pelo *mouse* (figura 11.37). Quatro procedimentos são disponibilizados ao usuário: Calibração, Aprendizado (Manual), Gerar Arquivo e Controle.

No momento inicial de execução do programa, automaticamente o sistema é conduzido ao modo de calibração da plataforma, que realiza o procedimento de leitura e armazenamento das informações provenientes dos sensores de posição de cada um dos atuadores (*encoders*).



Figura 11.37: Tela Principal do Programa.

11.8.3.2 – Modo de Calibração

Este modo permite efetuar a calibração da plataforma, devendo ser utilizado sempre na inicialização do sistema, e em caso de checagem de posicionamento pelo usuário. A tela principal (figura 11.38) é constituída de dois botões localizados no lado esquerdo da tela: efetuar calibração e posição inicial, um display responsável pelas informações referentes aos sensores de posição de cada uma das juntas (potenciômetro e *encoders*), indicando a posição linear (movimento de translação) e angular (referente aos dois movimentos de rotação da mesa).

Outros indicadores luminosos mostrarão ao usuário a posição zero do *encoders* (movimento rotativo) e sensores de final de curso no caso de movimento linear da base da plataforma. Os parâmetros de calibração dos sensores e dimensão da haste de calibração disposta em um ponto referenciado (x,y) da mesa (neste caso, por *default*, $x=0$, $y=0$ e $L_{haste} = 0,20m$) serão introduzidos pelo usuário e não poderão ser mais modificados após a fase de calibração. Dois botões localizados no lado inferiores da tela permitem gravar estas posições e sair desta tela, retornando ao menu principal, apresentado anteriormente.

Cada passo do procedimento de calibração é realizado através de telas interativas com o usuário, de modo a permitir a movimentação dos acionadores, monitoramento dos sensores de posicionamento e rotina de inicialização (posição zero). A figura 11.40 apresenta mensagens típicas do *menu* calibração.



a) Calibração de Posição do Atuador Linear



b) Calibração de Posição do Atuador 2



c) Calibração de Posição do Atuador 3

Figura 11.38: Telas do Programa de Calibração.



Figura 11.39: Tela de Calibração da Haste.

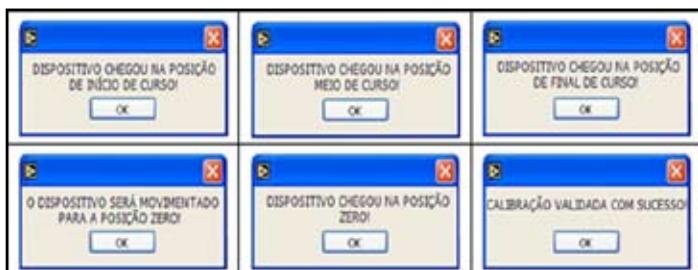


Figura 11.40: Mensagens Típicas do Menu de Calibração.

11.8.4 – Geração de Arquivos

11.8.4.1 - Modo Aprendizado

Após inicialização da plataforma (posicionamento do atuador linear até movimentação da mesa até sensor de posição central, e calibração dos sensores de posicionamento angular para orientação da mesa, operações realizadas no *menu* anteriormente descrito), o usuário poderá entrar no modo aprendizado.

Nesta tela o usuário poderá realizar movimentos no modo junta (rotações dos dois graus de liberdade da mesa) ou no modo cartesiano em relação ao ponto final da haste de calibração (neste modo o programa utiliza o modelo cinemático da mesa, considerando as dimensões e posicionamento da haste em relação ao centro da mesa rotativa). Este modo permite efetuar a geração de arquivos de movimentos de rotação da mesa a partir do movimento de aprendizagem.

No modo angular (figura 11.41a), o usuário realiza as movimentações de cada junta angular, armazenando esses valores para posterior interpolação e geração de movimentos no modo automático. A posição do ponto terminal da haste em coordenadas cartesianas é afixada na tela. No modo cartesiano (ou de posição) mostrado na figura 11.41b, o usuário move a mesa realizando movimentos cartesianos em relação ao ponto terminal da ferramenta utilizada (no caso a haste

de calibração), sendo mostradas na tela as respectivas posições angulares correspondentes ao movimento.

a) Modo Movimentação das Juntas.

b) Modo Cartesiano - Posicionamento da Haste.

Figura 11.41: Tela de operação- Geração de Arquivos de Trajetórias.

11.8.4.2 - Movimentação a partir de arquivo padrão EXCEL™

A partir de arquivo gerado em EXCELTM, representando a movimentação das juntas, foi implementado um módulo utilizando o *software* LabViewTM de modo a permitir:

- I. a partir de modelo cinemático apresentado anteriormente, realizar a conversão desses movimentos em sinais de referência (expressos em volts) a serem comparados com os sensores de posição (potenciômetro de precisão) a serem enviados para um controlador de posição PID, que fornecerá sinais de comando aos atuadores hidráulicos de correspondentes aos dois graus de liberdade de rotação do dispositivo.
- II. Implementação de *Software* de Supervisão e Controle e *Hardware* de Acionamento e Controle em ambiente LabViewTM, que permitirá a geração automática de movimentos da juntas.

11.8.4.3 - Modo Controle

O modo controle permite a leitura de um arquivo de movimentos gerado no *menu* de geração de arquivos apresentado anteriormente, ou a partir da leitura de um arquivo típico de forma de onda gerado em EXCEL™ apresentado anteriormente, que através de um fator de escala permitirá a redução do número de pontos na leitura do arquivo (valor de *default* fixado em 1). A figura 11.42 apresenta exemplos de telas típicas deste módulo.



Figura 11.42: Tela típica do Módulo de Controle.

11.8.4.4 - Tela de Controle (modo automático)

A tela de controle em modo automático (figura 11.43) é constituída de módulos básicos de controle, status do programa, visualização gráfica dos sensores de posição (*encoders* e sensores indutivos de posicionamento da plataforma) e ajuste de parâmetros do controlador de posição (PID).

Ao mesmo tempo ela é constituída de um botão de habilitação do programa de geração de movimentos para o acionamento das válvulas hidráulicas e interrupção (com sinalização em display). A identificação do ensaio é apresentada na tela.

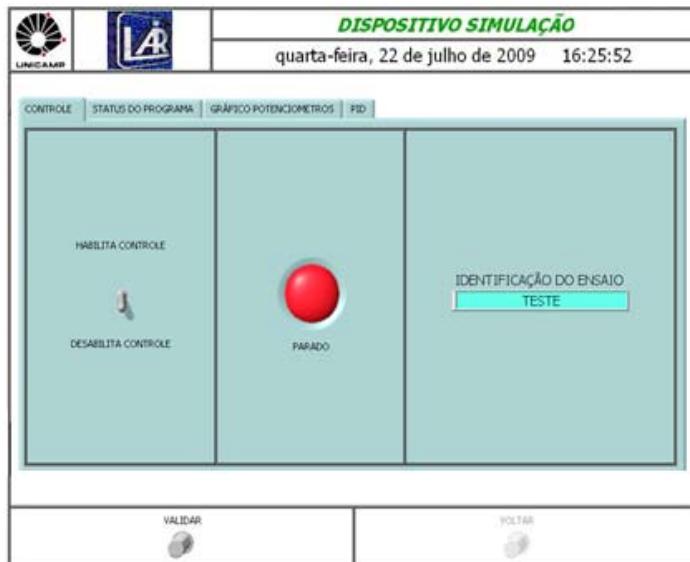


Figura 11.43: Tela de Controle (Modo Automático).

11.8.4.5 - Tela de Status

A tela de status mostra os diferentes passos constituintes da trajetória, sendo indicado o número do passo, o valor de referência atual a ser enviado ao controlador para posterior comparação com o *encoder* e a porcentagem de evolução da trajetória (figura 11.44).

11.8.4.6 - Tela de Monitoramento

A tela de monitoramento permite o monitoramento no tempo dos valores relativos aos sensores de posição das juntas (*encoders*), conforme mostra a figura 11.45.



Figura 11.44: Telas Típicas de Status de Evolução da Trajetória.

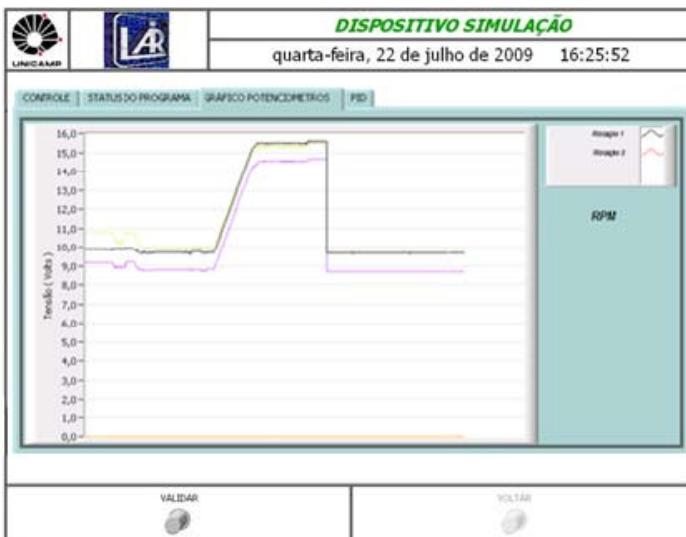


Figura 11.45: Tela de Monitoramento dos Sensores das Juntas.

11.8.4.7 - Tela de Ajuste de Parâmetros do Controlador de Posição PID

Esta tela permite o ajuste de parâmetros do controlador PID (Proporcional, Integral e Derivativo) e escolha da porcentagem de erro máximo (figura 11.46).

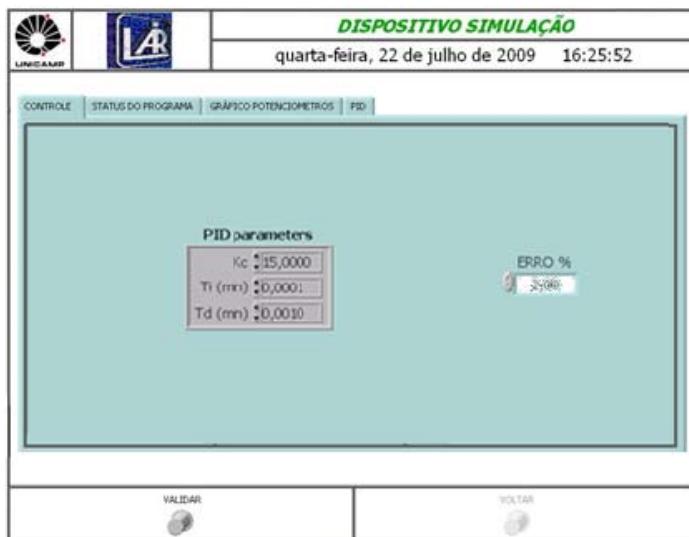


Figura 11.46: Tela de Ajuste de Parâmetros do Controlador PID.

11.8.4.8 - Tela de Modo de Segurança

Na implementação do programa final em LabViewTM foram contemplados aspectos relacionados à segurança de utilização do dispositivo. Os aspectos considerados foram: estabelecimento de limitação via *software* dos limites de final de curso de cada junta rotativa (informação fornecida durante a fase de calibração) e final de curso da base (informação fornecida pelos sensores indutivos dispostos na base). Essa proteção consiste basicamente na interrupção do programa (desliga o sistema de acionamento) e mensagem de alarme na tela ao usuário, conforme mostra a figura 11.47.



a) Limites de Segurança do Atuador Rotativo 1.



b) Limites de Segurança do Atuador Rotativo 2.

Figura 11.47: Tela de Alarme de Funcionamento (Modo Segurança).

11.8.4.9 - Geração de Arquivo de Banco de Dados

Após carregamento dos arquivos, um banco de dados, constituído de informações de funcionamento do arquivo é automaticamente gerado (figura 11.48).



Figura 11.48: Banco de Dados (Arquivos de Movimentação)

11.9 - Conclusão

Neste capítulo foi apresentado um exemplo prático de implementação de um dispositivo robótico (plataforma hidráulica) e sua modelagem matemática para que fosse implementado inicialmente dentro de um simulador virtual desenvolvido em ambiente Matlab/ Simulink®, com arquitetura modular e aberta. A concepção do simulador considerou apenas as duas juntas rotacionais da plataforma (base superior), mostrando algumas simulações possíveis. O simulador desenvolvido pode facilmente ser acrescido com outros módulos.

Como uma alternativa para a solução ao integrar e implementar uma programação executa-se as instruções

programadas por uma simples lógica de comunicação via porta serial E/S. Essa solução não atende de forma geral a alguns tipos de processos, principalmente em atividades que se exigem respostas rápidas em relação aos movimentos dos braços robóticos, bem como na sincronia de movimentos. Para validação deste trabalho de pesquisa foram implementados dois estudos:

- a) Plataforma *WEB* colaborativa em Automação: integração de dois robôs industriais junto com dispositivo robótico de 3 GL para realização de tarefas colaborativas automatizadas através de *WEB*;
- b) Implementação de Sistema de Supervisão e Controle de um dispositivo robótico com 3 GL, com ênfase na modelagem cinemática e controle de posicionamento.

11.10 - Referências Bibliográficas

Asfahl, C. Ray. Building Blocks of Automation in: **Robots and Manufacturing Automation**, 2nd ed. New York: John Wiley & Sons, 1992 p 23-53.

Brumson, B. **Emerging Trends in Robotic Integration**. Robotics online e-news. Michigan, Nov.2001. Disponível em: <<http://www.roboticsonline.com/public/articles/archivedetails.cfm?id=577>> Acesso 19 jan. 2004.

Camelot, R., **Robot Off-line Programming and Simulation system for industrial robots**. Disponível em: <<http://www.camelot.dk/english/offvson.htm>> Acesso 26 jun. 2003.

Campbell, J.E., **Robot Motion Control: How Special is it?** Robotics online e-news. Michigan, Set.2003, Disponível em: <<http://www.roboticsonline.com/public/articles/archivedetails.cfm?id=1174>>, Acesso 19 jan. 2004.

Easy-Rob, Easy-rob, **The Company for 3D Robot Simulation**, Disponível em: <<http://www.easy-rob.com>> Acesso 03 Set. 2003.

Fiedler, P. J., Schilb, **C. J. Open Architecture Robot Controllers and Workcell Integration**. Robotics online e-news. Technical Papers, Michigan. Disponível em: <http://www.roboticsonline.com/public/articles/Open_Architecture_Genesis.PDF> Acesso 17 set. 2003.

Fiedler, P. J., Dehof, M.K. **Workcell Communications: Connectivity, Man-Machine interfaces, and Multi System Management**. Robotics online e-news. Technical Papers, Michigan. Disponível em: <http://www.roboticsonline.com/public/articles/Workcell_Communications_ANS98_Genesis.PDF> Acesso 17 Set. 2003.

Flow Software Technologies, **Workspace5 - Robotic simulation and off-line programming software**. Disponível em: <<http://www.workspace5.com>> Acesso 03 Set. 2003.

Fu, K.S., González, R.C., Lee, C.S.G. **Robotics: Control, Sensing, Vision, and Intelligence**, McGraw-Hill International, 1987, 580p

Groover, Mikell P. Automation in: Dorf, Richard C. (Ed), Nof, Shimon Y. **International Encyclopedia of Robotics: Applications and Automation**, New York: John Wiley & Sons, 1988 p 136-151

Hardin, W., **Robotic Integrators take their Specialties Seriously. Robotics online e-news. Michigan, Abr.2002**

Nagle, B. Your First Robot. **Robotics Industry Directory - 2002-03 Edition**, A publication of robotic Industries Association, p 8.

Rosário, J.M. Modelagem e Controle de Robôs in: Romano, V.F. (Ed) **Robótica Industrial: Aplicação na Indústria de Manufatura e Processos**, São Paulo: Edgard Blucher Ltda, 2002 p 20-46

Roos, E., Behrens, A., Anton, S., RDS – Realistic Dynamic Simulation of Robots. Disponível em < <http://www.easy-rob.com/data/rds.pdf>>. Acesso 29 mar. 2003

Roos, E., Behrens, A. **Off-line Programming of industrial robots** – Adaptation of simulated user programs to the real environment. Computers in industry, 1997 33(1): p 139-150.

Salminen, V: Ten Years of Mechatronics Research and Industrial Applications in Finland, IEEE-ASME Transactions on Mechatronics, vol. 1, no. 2, p. 103105, 1996.

Trostman, E., Conrad, F., Trostman, S., Nielsen L.F. **Robot Off-line programming and simulation as a true CIME subsystem.** In: IEEE International Conference on Robotics and Automation. 1992.