Universidade Federal de Santa Catarina

Curso de Pós-Graduação em Engenharia Mecânica

UMA INTERFACE CAD/CAM PARA A PROGRAMAÇÃO FORA DE LINHA DE ROBÔS INDUSTRIAIS

Dissertação Submetida à Universidade Federal de Santa Catarina para Obtenção do Grau de Mestre em Engenharia Mecânica

ALUNO: LEONARDO BASTOS DE TOLEDO

ORIENTADOR: ALTAMIR DIAS, D. SC.

Florianópolis, Fevereiro de 2000.

Uma Interface CAD/CAM para a Programação
Fora de Linha de Robôs Industriais

Leonardo Bastos de Toledo

Esta Dissertação foi Julgada Adequada para a Obtenção do
Título de

Mestre em Engenharia

Especialidade Engenharia Mecânica,
Área de Concentração Projeto de Sistemas Mecânicos com
Ênfase em Sistemas de CAE/CAD/CAM, Robótica e Controle,
e Aprovada em sua Forma Final pelo
Curso de Pós-graduação em Engenharia Mecânica

Altamir Dias, D.Sc.
Orientador

Júlio César Passos, Dr.
Coordenador da Pós-graduação

Banca Examinadora

Raul Guenther, D.Sc.
Presidente

Edson de Pieri, Dr.
Membro

Marcelo Ricardo Stemmer, Dr.
Membro

"Muitos pensam que a pesquisa científica é uma atividade puramente racional, na qual o objetivismo lógico é o único mecanismo capaz de gerar conhecimento. Como resultado, os cientistas são vistos como insensíveis e limitados, um grupo de pessoas que corrompe a beleza da natureza ao analisá-la matematicamente. Essa generalização, como a maioria das generalizações, me parece profundamente injusta, já que ela não incorpora a motivação mais importante do cientista, o seu fascínio pela Natureza e seus mistérios. Que outro motivo justificaria a dedicação de toda uma vida ao estudo dos fenômenos naturais, senão uma profunda veneração pela sua beleza? A ciência vai muito além da sua mera prática. Por trás das fórmulas complicadas, das tabelas de dados experimentais e da linguagem técnica, encontra-se uma pessoa tentando transcender as barreiras imediatas da vida diária, guiada por um insaciável desejo de adquirir um nível mais profundo de conhecimento e realização própria. Sob esse prisma, o processo criativo científico não é assim tão diferente do processo criativo nas artes, isto é, um veículo de autodescoberta que se manifesta ao tentarmos capturar a nossa essência e lugar no Universo".

Marcelo Gleiser, A Dança do Universo – Dos mitos de Criação ao Big-Bang.

DEDICATÓRIA

Aos meus pais, Arnaldo e Tania,
pelo carinho e ensinamentos de vida a mim repassados.
Aos meus irmãos, Gabriela e Rodrigo,
pelo companheirismo.

AGRADECIMENTOS

- À Universidade Federal de Santa Catarina e ao Curso de Pós-graduação em Engenharia Mecânica, pela oportunidade e por proporcionar as condições para a realização deste trabalho.

- À CAPES, pela oportunidade e pelo apoio financeiro para realização do trabalho.

- Ao Prof. Altamir Dias, pela confiança depositada e pela orientação que se fez presente nos momentos decisivos do trabalho.

- Aos Professores Altamir Dias, Raul Guenther, Edison da Rosa, Victor De Negri, Arno Bollmann pelos conhecimentos transmitidos nas cadeiras de Projeto de Sistemas Mecânicos.

- Ao pessoal do GRANTE, pelas discussões, pela amizade e pela convivência agradável durante estes anos. Em especial aos amigos Ricardo, Michael, Alex, Mineiro, Vinadé e Valdir.

- Ao amigo Cláudio Menescal, *Menesca*, com quem pude compartilhar a experiência vivida durante toda a estada em Florianópolis.

- A cidade de Florianópolis, Linda e Maravilhosa, com suas belas praias e tranqüilidade de cidade do interior. Continue assim!!!

- À todos que direta ou indiretamente contribuíram para a realização deste trabalho.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

SIMBOLOGIA

Q	- Vetor de coordenadas das juntas de um manipulador.
$q_1 \ldots q_n$	- Vetores de posição das n juntas de um manipulador.
$\dot{q}_1 \ldots \dot{q}_n$	- Vetores de velocidade das n juntas de um manipulador.
$\ddot{q}_1 \ldots \ddot{q}_n$	- Vetores de aceleração das n juntas de um manipulador.
X	- Vetor de coordenadas operacionais do efetuador do manipulador.
x, y, z	- Vetores das coordenadas de posição de um manipulador.
ϕ, φ, ψ	- Vetores das coordenadas de orientação de um manipulador.
U	- Superfície planar pertencente ao espaço Euclidiano R^2
a, b, c, d	- Valores dos limites da superfície planar U.
S	- Superfície paramétrica pertencente ao espaço Euclidiano R^3.
g	- Função de modelagem de superfícies paramétricas.
u,v	- Parâmetros da superfície paramétrica.
γ	- Percurso de pontos nos parâmetros de uma superfície paramétrica.
α	- Percurso de pontos em coordenadas cartesianas.
s	- Parâmetro do percurso de pontos nos parâmetros de uma superfície paramétrica.
s_0	- Valor inicial do parâmetro s.
s_1	- Valor final do parâmetro s.
P	- Função de modelagem das curvas "B-Splines".
B_i	- Coeficientes da formulação de curvas "B-Splines".
$N_{i,k}$	- Funções de base da formulação de curvas "B-Splines".
x_i	- i-ésimo valor do vetor de referência da formulação de curvas "B-Splines"
t	- Parâmetro da formulação de curvas "B-Splines".
k	- Ordem da curva "B-Spline".
n	- Número de vértices do polígono de definição da formulação de curvas "B-Splines".
${}^{B}_{A}R$	Transformação de rotação de um sistema de coordenadas de A para B
${}^{C}_{B}T$	Transformação de translação de um sistema de coordenadas de B para C

$P^{/}$ - Derivada primeira da função de modelagem das curvas "B-Splines".

$P^{//}$ - Derivada segunda da função de modelagem das curvas "B-Splines".

$N^{/}_{i,k}$ - Derivada primeira da função de base da formulação de curvas "B-Splines".

$N^{//}_{i,k}$ - Derivada segunda da função de base da formulação de curvas "B-Splines".

-

$J_{n,i}$ - Funções de base da formulação de superfícies de Bézier na direção do

$K_{m,j}$ - Funções de base da formulação de superfícies de Bézier na direção do

$B_{i,j}$ - Rede poligonal de definição das superfícies de Bézier

RESUMO

Este trabalho trata do problema do planejamento e programação de trajetórias em manipuladores servo controlados, com ênfase no uso das técnicas de projeto e de manufatura auxiliadas por computador (CAD/CAM). A proposição consiste em usar a informação geométrica do contorno do produto, obtida a partir da modelagem de superfícies bi-paramétricas em sistemas CAD, em procedimentos de planejamento de trajetórias. Estes procedimentos, desenvolvidos e integrados, fazem uso de formulações de curvas e superfícies "B-splines". A meta principal do trabalho é efetuar a análise e a programação de atividades de robôs industriais em processos de fabricação que envolvam um movimento contínuo do efetuador, tais como a pintura, o polimento, o desbaste e a soldagem.

O estudo de programação de tarefas em manipuladores resultou no desenvolvimento de um programa computacional escrito na linguagem de programação do MatLab, para o sistema operacional Windows numa plataforma IBM-PC. A interface desenvolvida consiste em rotinas para entrada de dados, de cálculo inerentes à cinemática de robôs e de visualização de resultados.

Foram feitos testes para avaliar o comportamento do método quanto à precisão, tempo de processamento, nível de controle dos dados gerados. Também são incluídos estudos para verificar a aplicabilidade da metodologia proposta e do sistema implementado, através da programação de dois diferentes robôs industriais.

ABSTRACT

This work presents the trajectory planning and programming problem applied to servo-controlled manipulators. CAD techniques applied to product design and manufacturing are studied and used to task programming of robots. Geometric information is available when parts are designed in systems CAD/CAM. This geometric information is used to know what path the end-effector of manipulator has to follow. Procedures to program path planning include B-Splines curves and surfaces mathematics formulation. The main goal of this research is to cover manufacturing processes associated to painting, polishing and welding activities.

The results of this research are presented in form a software written using a MatLab programming language for Windows IBM-PC platform. The interface includes data generation from surfaces involved, kinematics calculation and data visualization.

Tests to evaluate the performance of the algorithms are done. To evaluate the software performance some examples are presented for two different kinds of laboratory robots.

CAPÍTULO 1

INTRODUÇÃO

Os avanços obtidos em tecnologia de processadores e a conseqüente geração de equipamentos e ferramentas para a fabricação trouxeram enormes mudanças à produção industrial nas duas últimas décadas. A utilização de sistemas integrados, através da aplicação de sistemas mecânicos, eletrônicos, e computacionais para operar e controlar a produção, de modo a realizarem os processos, as montagens, as manipulações de materiais, e as inspeções das atividades, com pequena ou nenhuma participação humana, tornaram a automação da produção industrial mais do que uma realidade. É possível afirmar que hoje isso se torna uma exigência de um mercado globalizado, altamente competitivo, onde as empresas necessitam produzir cada vez mais, com melhor qualidade e com custos reduzidos.

Os robôs industriais constituem um dos elementos principais na integração da automação industrial, onde sua maior atribuição consiste no fato de ser uma máquina programável que possui certas características antropomórficas. Nos robôs industriais, a característica antropomórfica mais óbvia é o braço mecânico do robô que é usado para executar tarefas industriais. Características humanas menos óbvias são as capacidades do robô para tomar decisões, responder à entrada de sensores e comunicar-se com outras máquinas. Dentre os

desenvolvimentos que foram obtidos em tecnologia de robô nas últimas décadas estão a maior sofisticação dos controladores de robô, as melhorias na precisão de posicionamento e movimentação e a adoção da tecnologia de sensores. Estes fatores tornaram possível o desenvolvimento de métodos para o planejamento e a programação dos equipamentos para diferentes tarefas do processo produtivo. A capacidade de reprogramação dos robôs industriais vem adicionando aos processos de produção uma maior flexibilidade e velocidade na obtenção de um determinado produto, além de precisão, confiabilidade e repetibilidade dos processos.

A tecnologia de computador teve também uma grande influência nas funções do ciclo de processamento de informação para o apoio à fabricação. O projeto e a fabricação auxiliados por computador (CAD/CAM) constituem técnicas para aperfeiçoar a eficiência das atividades das engenharias de projeto e de fabricação através do esforço de integração destas, ao invés de tratá-las como duas diferentes atividades. Inicialmente, a técnica estava limitada às empresas que podiam dispor de grandes e poderosos sistemas computacionais. Com a contínua redução dos preços dos computadores e o aparecimento dos microcomputadores, os sistemas CAD são acessíveis até às pequenas companhias, e, portanto, mais engenheiros puderam se beneficiar do acesso ao hardware e software para uso em aplicações de CAD.

Com a disponibilidade de computadores pessoais e de sistemas CAD/CAM, criou-se uma consciência crescente das vantagens do uso destes sistemas na programação dos robôs industriais. Com isso, os sistemas CAD/CAM vêem assumindo responsabilidade cada vez maior na criação das tarefas dos robôs industriais, conforme se desenvolvem para as aplicações destes equipamentos. Atualmente as indústrias de sistemas CAD/CAM desenvolvem ferramentas gráficas para a simulação de sistemas industriais flexíveis e, também para sistemas robóticos. A meta final é obter vantagens com o uso da programação fora de linha para os sistemas de robôs de maneira semelhante às ferramentas presentes para programar máquinas NC e CNC.

Como será visto ao longo do trabalho, a utilização de software de CAD/CAM e de estações de trabalho de engenharia para o planejamento e a programação das atividades dos robôs industriais fornece o potencial para ultrapassar muitas das dificuldades encontradas no procedimento de programação destes equipamentos. Os sistemas de programação com fundamentos gráficos possibilitam aperfeiçoar a célula de manufatura, com resultado direto na diminuição dos custos de utilização da célula, do tempo de fabricação e, ainda, uma melhoria da qualidade do produto.A possibilidade de comunicação entre os controladores dos robôs e os sistemas gráficos interativos de simulação e de programação fornece um importante potencial para a difusão do uso dos robôs industriais na automação industrial.

Tendo em vista a potencialidade dos sistemas CAD/CAM para o aumento da aplicabilidade dos robôs industriais nas tarefas dos processos industriais de produção, o objeto principal desta pesquisa consiste em apresentar um estudo dos conceitos envolvidos e as características dos métodos de utilização dos sistemas CAD/CAM para programação dos robôs industriais. A partir destes conceitos é apresentada uma proposta de metodologia, defendida em congressos Nacional e Internacional [53, 54,] para o uso desta técnica na programação de robôs industriais. O programa desenvolvido, também apresentado em congressos Nacional e Internacional [14, 15], com o uso do método em análise, é aplicável em certos tipos de atividades de fabricação dos robôs industriais, onde as tarefas estão relacionadas com o contorno do produto em questão. No trabalho é desenvolvida uma interface para a programação de robôs industriais diretamente a partir de modelos geométricos gerados em sistemas CAD. Realiza-se a interação de diversos softwares, num ambiente computacional, para planejar e definir as trajetórias cartesianas a partir dos contornos dos produtos envolvidos, processar e analisar os dados destas trajetórias no espaço de juntas de diferentes manipuladores, e preparar os dados a serem transcritos para uma linguagem de programação do controlador do robô industrial.

O trabalho foi elaborado em cinco partes, expostas ao longo dos capítulos desta pesquisa.

No capítulo 2 são apresentadas as características gerais dos sistemas de robôs industriais e a da sua programação através do estudo dos componentes destes sistemas e dos métodos de programação destes equipamentos.

No capítulo 3 é realizado um estudo sobre o desenvolvimento e a utilização de sistemas CAD/CAM para a programação de robôs industriais. Nesta etapa são revisados alguns conceitos básicos gerais importantes para o entendimento dos elementos necessários ao uso da técnica. Sendo assim, inicialmente é feita uma revisão das características gerais e dos conceitos básicos para o desenvolvimento dos sistemas gráficos interativos e, posteriormente, são analisados especificamente os sistemas CAD/CAM. Finalmente, é revisado, de modo geral, o estado da arte da aplicação da técnica na área de robótica, registrando os elementos envolvidos, as características de uso e seus benefícios, através de alguns sistemas existentes e publicações pertinentes.

No capítulo 4 é apresentada a proposição de metodologia de integração de sistemas CAD/CAM para o planejamento, a simulação cinemática e dinâmica, e a programação fora de linha de robôs industriais em atividades de fabricação. Nesta etapa é feita, paralelamente, a descrição da interface desenvolvida.

No capítulo 5, a utilização do programa desenvolvido e alguns testes da metodologia proposta são apresentados. Os testes visam demonstrar o desempenho de precisão e de qualidade dos dados obtidos, bem como as possibilidades de controle dos dados a serem gerados, e ainda, a possibilidade de programação de robôs industriais com estes dados gerados.

Finalmente, no capítulo 6 são feitas as conclusões sobre o estudo realizado, a metodologia proposta e o programa implementado. Algumas recomendações para a melhoria do programa e sugestões para trabalhos futuros envolvendo a programação fora de linha dos robôs industriais com o uso de recursos da computação gráfica são também apresentadas.

No Apêndice 1 é apresentado em mais detalhe o procedimento e as equações para a modelagem de curvas "B-Splines" utilizadas no programa desenvolvido para a programação fora de linha de robôs industriais.

O programa utiliza modelos de superfícies paramétricas cujos métodos de modelagem estão apresentados detalhadamente no Apêndice 2.

E ainda, considerando a importância do conhecimento já existente da técnica para a programação das aplicações das máquinas ferramentas numericamente controladas, é realizado, no Apêndice 3, um breve apanhado dos conceitos e das características importantes desta área de aplicação dos sistemas CAD/CAM, permitindo a identificação de semelhanças com a robótica.

Finalmente, no Apêndice 4, tem-se uma breve explanação das fases identificáveis para implementação de uma nova aplicação de robôs industrias e os passos necessários no planejamento de uma nova aplicação, permitindo vislumbrar detalhadamente o escopo do uso dos sistemas CAD/CAM em robótica.

CAPÍTULO 2

CONCEITOS E DEFINIÇÕES PARA O PLANEJAMENTO E A PROGRAMAÇÃO DE TAREFAS DOS ROBÔS INDUSTRIAIS

2.1 INTRODUÇÃO

Como enfatiza RANKY [41], para a programação dos robôs industriais, ou outra máquina controlada por computador, é sempre importante pensar em termos de um sistema, consistindo de vários subsistemas no qual o robô por si só é apenas um componente.

Sendo assim, é necessário então que se conheça a estrutura de um sistema robótico, suas características e interdependências. Neste intuito, no presente capítulo são apresentadas as características básicas que definem o mecanismo de um manipulador, o seu acionamento, o seu controle e a sua programação. Algumas definições envolvendo o papel dos sistemas robóticos nos processos de produção industrial também são apresentadas visando mostrar a sua relação com a programação.

2.2 A ESTRUTURA BÁSICA DE UM ROBÔ INDUSTRIAL

Segundo a definição adotada pela RIA[1] [50]:

"Um robô é um manipulador reprogramável, multifuncional, projetado para mover materiais, peças, ferramentas ou outros dispositivos especiais para a execução de uma variedade de tarefas".

A figura 2.1, a seguir, ilustra um diagrama representativo dos sistemas de robôs industriais. Assim como as máquinas ferramentas numericamente controladas, os atuais robôs industriais são sistemas avançados de automação, nos quais os movimentos dos eixos do dispositivo mecânico são produzidos por um sistema de acionamento e coordenados por um sistema de controle computadorizado, de modo a guiar uma ferramenta para execução de tarefas [30]. As tarefas são executadas de acordo com comandos preestabelecidos de movimento e de ações através de procedimentos de programação. Os procedimentos de programação estão ligados com o fornecimento dos dados necessários para descrever o ciclo de trabalho do robô para execução de uma tarefa desejada [19].

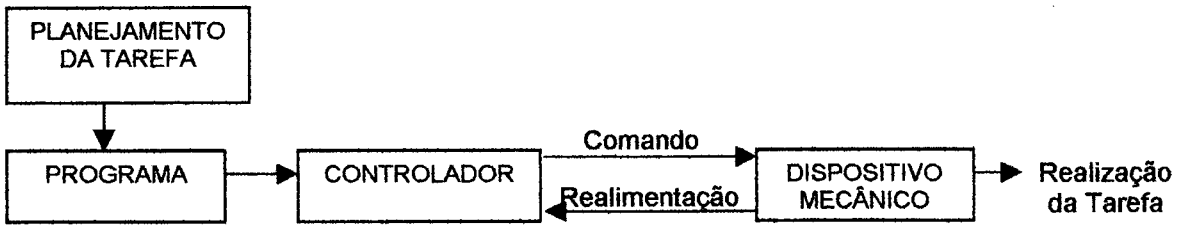

FIGURA 2.1: Diagrama Representativo dos Sistemas Robóticos Industriais [30]

2.2.1 O DISPOSITIVO MECÂNICO

Segundo CRAIG [11], um manipulador mecânico é composto por ligações ou elos rígidos, conectados por eixos de juntas que permitem o movimento relativo entre as ligações vizinhas. (Figura 2.2.) As juntas, de um modo geral, podem ser rotativas ou prismáticas, sendo os seus deslocamentos chamados de ângulos de junta ou translações de junta, respectivamente.

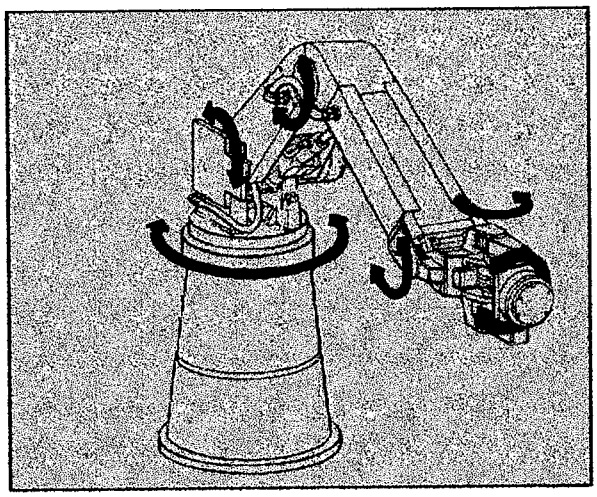

FIGURA 2.2: O Manipulador Mecânico.

A estrutura mecânica formada pelas ligações e juntas constitui uma cadeia cinemática simples e aberta. Simples, pois o número de graus de liberdade que o manipulador possui é igual ao número de variáveis de posição independentes, que devem ser especificadas para localizar todas as partes do mecanismo, e aberta, pois pelo menos um elo está conectado a apenas uma junta.

[1] Robot Institute of America

2.2.1.1 O ESPAÇO DE JUNTAS

A seqüência dos tipos de movimentos dos eixos das juntas e o número de graus de liberdade do manipulador determinam a configuração da cadeia cinemática do robô [04]. Esta é então identificada por um conjunto de variáveis denominadas de coordenadas de juntas ou coordenadas generalizadas. Cada coordenada corresponde a um grau de liberdade e no espaço de juntas define-se um vetor Q da seguinte forma:

$$Q = [q_1 \ q_2 \ \dots \ q_n]^T \tag{2.1}$$

onde n representa o número de graus de liberdade do robô.

Teoricamente, pode haver um grande número de configurações possíveis para um robô. Porém, a maioria dos robôs industriais são manipuladores de cadeia cinemática com seis graus de liberdade. Isto devido ao fato de que são necessários no mínimo seis graus de liberdade para determinar uma posição e uma orientação arbitrária no espaço cartesiano [03].

Também, podem ser identificados em sua estrutura o braço, o punho e o efetuador. O braço, fixado à uma base, contém os três primeiros graus de liberdade e é o responsável pelos movimentos mais amplos do efetuador. O punho, conectado à extremidade livre do braço, contém os três últimos graus de liberdade do manipulador, e é o responsável pelos movimentos locais do efetuador. Na extremidade livre da cadeia cinemática do manipulador está fixado o efetuador, que é o elemento ativo para o cumprimento das tarefas [16].

2.2.1.2 AS CONFIGURAÇÕES COMUNS DA CADEIA CINEMÁTICA

Sob um ponto de vista prático, a maioria dos robôs recaem em algumas categorias populares de configurações para o braço, sendo estas: a **esférica**, a **cilíndrica**, a **cartesiana** e a **articulada** [52]. (Figura 2.3.) As três primeiras configurações foram denominadas fazendo-se uma analogia com os respectivos sistemas de coordenadas que elas representam.

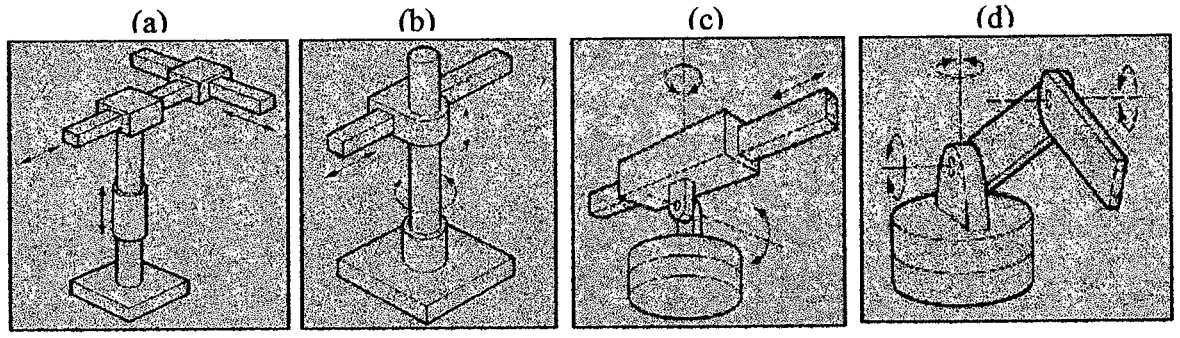

FIGURA 2.3: Configurações Comuns para o braço. (a) Cartesiano (b) Esférico (c) Cilíndrico (d) Articulado.

Cada configuração possui características próprias que a qualificam para determinadas tarefas [16]. Os robôs cartesianos, constituídos apenas por juntas prismáticas, são indicados para operações de montagens de precisão e operações de inspeção, devido à rigidez de sua estrutura. Os robôs cilíndricos, que diferem do anterior apenas por uma junta de rotação na base do manipulador, possuem rigidez estrutural e capacidade de carga similar aos cartesianos, sendo utilizados para a alimentação de máquinas e no transporte de peças. As configurações esféricas e articuladas possuem um alcance e uma mobilidade maior que as primeiras, sendo indicadas para operações mais complexas, entre estas a manipulação, a solda e a pintura.

2.2.1.3 O ESPAÇO OPERACIONAL DO MANIPULADOR

Na maioria dos robôs industriais, seu propósito é movimentar seu efetuador, pois este é o elemento que realiza a interface física com o ambiente [55]. O efetuador pode ser representado no espaço operacional por um vetor X, denominado coordenadas operacionais, dado por:

$$X = [\, x_1 \; x_2 \ldots \; x_m \,]^T, \tag{2.2}$$

onde m é o número de coordenadas necessárias para representar o efetuador no espaço. No caso mais genérico, adota-se m=6, como já mencionado, e podendo-se acessar um ponto com qualquer orientação.

Neste caso o vetor de coordenadas no espaço operacional compreende duas parcelas: as variáveis de posição e as de orientação.

$$X = [\, x \; y \; z \; / \; \phi \; \varphi \; \psi \,]^T \tag{2.3}$$

Usualmente, determina-se a orientação e o posicionamento de um sistema de coordenadas {C} fixado no efetuador do manipulador com relação a um sistema de coordenadas inerciais {A}, normalmente fixado na base do manipulador, através de uma transformação de rotação ${}^{B}_{A}R(\phi,\varphi,\psi)$ e uma transformação de translação ${}^{C}_{B}T(x,y,z)$, como ilustrado na figura 2.4. CRAIG [11] apresenta diversos métodos para representação da orientação: ângulos X-Y-Z fixos, ângulos de Euler Z-Y-X, ângulos de Euler Z-Y-Z. No método dos ângulos X-Y-Z fixos, também conhecidos como ângulos "roll, pitch & yaw", a representação da orientação é feita através de rotações consecutivas em torno dos eixos coordenadas X, Y e Z, também ilustrada na figura 2.4.

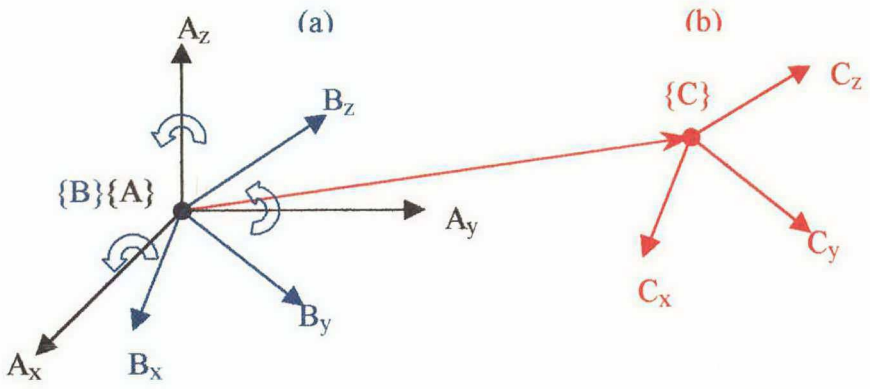

FIGURA 2.4: Representação da posição e orientação do sistema de coordenadas {C} definida através de (a) uma rotação R e (b) uma translação T com relação ao sistema de referência {A}.

De modo geral as aplicações de robôs industriais consistem em determinar as variações temporais das posições x, y e z e das orientações ϕ, φ e ψ do efetuador da cadeia cinemática para o cumprimento de determinada tarefa [18].

2.2.1.4 O ESPAÇO DE TRABALHO

O espaço de trabalho da cadeia cinemática é definido como o envelope ou espaço dentro do qual o robô pode manipular o efetuador, sendo determinado pelo número e tipos de juntas do manipulador, pelo tamanho físico das juntas e ligações, e pelos alcances das várias juntas [19]. A forma do volume de trabalho depende em grande parte do tipo de configuração do braço do robô. Uma configuração polar tende a ter uma esfera parcial como seu volume de trabalho; um robô cilíndrico tem um envelope de trabalho cilíndrico; e um robô cartesiano tem um espaço de trabalho retangular.

2.2.2 O SISTEMA DE ACIONAMENTO

Todos os eixos do manipulador possuem um sistema de acionamento que converte os sinais elétricos de comando em movimentos mecânicos [30].

São os elementos que, através dos sinais enviados pelo controlador, produzem fisicamente a movimentação das juntas agindo diretamente sobre estas ou através de elementos de transmissão [04].

Quanto à natureza de acionamento, os robôs podem ser classificados em pneumáticos, hidráulicos, elétricos, ou mistos [19].

Segundo [04], a característica mais distinta para descrever um robô industrial é o seu sistema de acionamento pois, normalmente, este determina o alcance das características de desempenho do robô e a viabilidade das várias aplicações.

Algumas características com respeito às diferentes naturezas de acionamento foram reunidas a partir das referências [04, 19, 30, 35]:

Os atuadores hidráulicos são recomendados quando se necessita mover grandes cargas, com velocidades altas e com uma razoável precisão, diminuindo ao máximo os efeitos de vibração. Entretanto, requerem alta potência para o seu funcionamento, e na ocorrência de vazamentos poluem o ambiente, não sendo aconselháveis em ambientes que exijam alto grau de limpeza.

Os elementos pneumáticos são mais limpos, de baixo custo, pois utilizam o ar comprimido como fonte de potência, e podem alcançar alta velocidade. Porém, devido à compressibilidade do fluído, seu controle e regulagem são mais complexos, além de ter carga limitada quando comparados com os hidráulicos.

Os motores elétricos sempre são uma opção válida para o controle de sistemas mecânicos que exigem alto grau de precisão e repetibilidade, apesar das suas limitações em termos de

potência e até mesmo em relação ao tempo de resposta quando comparados aos atuadores hidráulicos. Em certas aplicações a melhor escolha recai sobre os tradicionais atuadores elétricos. O sinal elétrico é limpo, seguro, fácil de controlar, manipular e transportar.

2.2.3. O SISTEMA DE CONTROLE

O controle é um ingrediente essencial em qualquer sistema automatizado de produção [27].

Trata-se do processo no qual faz-se uma variável de sistema aderir a um valor particular, chamado valor de referência [39].

De modo geral, dois tipos de controladores podem ser definidos: o controlador em malha aberta e o controlador em malha fechada [30].

A figura 2.5, a seguir, ilustra o diagrama de blocos de um típico controlador em malha aberta A figura 2.6 ilustra um diagrama de blocos típico do controle em malha fechada.[39]

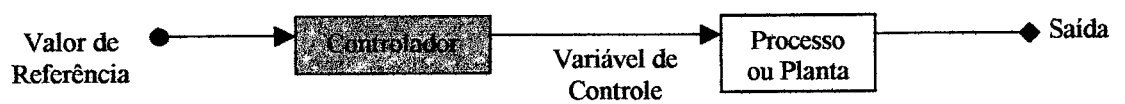

FIGURA 2.5: Sistema de Controle em Malha Aberta.

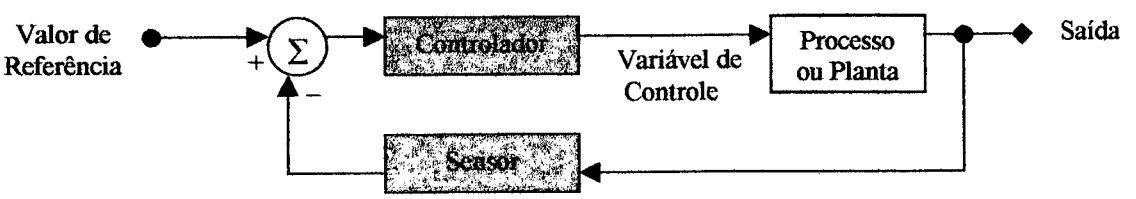

FIGURA 2.6: Sistema de Controle em Malha Fechada.

Nos sistemas de controle em malha fechada, a realimentação é usada para monitorar em tempo real as variáveis de saída e corrigir as discrepâncias com relação à saída desejada [32].

Segundo HOLLINGSHEAD [25], o controle do movimento do braço é o elemento mais crítico em um robô industrial, porque afeta o desempenho do sistema. Os dois fatores chave são a velocidade na qual o "hardware" ou "software" de servo controle pode ser atualizado e a precisão com que são calculados os comandos.

Segundo a RIA, de acordo com o nível de sofisticação do algoritmo de controle do controlador de robô, este pode ser classificado em uma das seguintes quatro categorias [19]: robôs de sequência limitada, de repetição ponto a ponto e de repetição com caminho contínuo, e inteligentes.

Os robôs de sequência limitada possuem o tipo de controle mais elementar e só podem ser utilizados durante ciclos de movimento simples, como as operações "pick-and-place". Normalmente são implementados fixando limites ou paradas mecânicas para cada articulação e sequenciando os atuadores das articulações para realizar o ciclo. A realimentação é usada apenas para indicar que a atuação de uma junta particular foi realizada, de forma que o próximo passo na seqüência possa ser iniciado. Porém, não há nenhum servo controle para realizar o posicionamento preciso da articulação. Muitos robôs pneumáticos são robôs de seqüência limitada.

Os robôs de repetição com controle ponto-a-ponto representam uma forma mais sofisticada de controle do que os robôs de seqüência limitada. Nestes sistemas, o controlador tem uma memória para não só registrar a seqüência dos movimentos em um determinado ciclo de trabalho, mas também as localizações que são associadas com cada elemento do ciclo de movimento. No controle ponto-a-ponto (PTP), são registradas posições individuais do braço de robô em memória. A realimentação é usada durante o ciclo de movimento para averiguar se as articulações individuais alcançaram as localizações desejadas definidas no programa.

Os robôs de repetição com controle de caminho contínuo têm a mesma capacidade de repetição como o tipo prévio; porém, o número de localizações individuais que podem ser registradas em memória é muito maior do que no de repetição ponto-a-ponto. Isto significa que os pontos que constituem o ciclo de movimento podem ser espaçados muito próximos, para realizar um movimento contínuo e suave. Em PTP, só a localização final dos elementos de movimento individuais é controlada; o caminho levado pelo braço para alcançar a localização final não é controlado. Em um caminho contínuo, o movimento do braço e do punho são controlados durante sua execução. É usado o servo controle para manter um controle contínuo sobre os da posição e da velocidade do manipulador.

Por último, a categoria dos robôs industriais inteligentes. Algumas das características que fazem um robô parecer inteligente são a capacidade para interagir com seu ambiente, tomar decisões diante de erros durante o ciclo de trabalho, comunicar com seres humanos, fazer computações durante o ciclo de movimento, e operar em resposta a entradas avançadas de sensores como a visão de máquina. Além disso, estes robôs possuem a capacidade de controle PTP e de caminho contínuo. Estas características requerem um nível relativamente alto de carga computacional e uma linguagem de programação avançada, para que se possa introduzir a lógica de decisão na fabricação, entre outras " inteligências " na memória.

Os tipos de sistemas de acionamento, de atuadores, de sensores e de controle com realimentação para as articulações determinam as características de resposta dinâmica do manipulador. A velocidade com que o robô pode alcançar uma posição programada .e a estabilidade de seu movimento são duas características importantes de resposta dinâmica em robótica. A velocidade de resposta é importante porque influencia o tempo de ciclo do robô. Isto determinará a taxa de produção na aplicação do robô. A estabilidade do robô se refere ao sobrepasso e oscilação que acontece no movimento do robô conforme este tenta alcançar uma

certa localização. A maior oscilação no movimento é uma indicação de menor estabilidade. Por sua vez, os robôs com maior estabilidade são inerentemente mais lentos na sua resposta. [19]

2.3 A PROGRAMAÇÃO DOS ROBÔS INDUSTRIAIS

Para fazer o trabalho útil o robô industrial deve ser programado para executar o seu ciclo de movimento. Numa perspectiva do robô, um programa de instruções pode ser definido como uma trajetória no espaço a ser seguida pelo manipulador, combinada com ações periféricas que apoiam o ciclo de movimento, conforme destaca GROOVER [19]. Trata-se de um conjunto de informações sobre trajetória, controle e sinais dos sensores que resultam em sinais enviados ao manipulador para executar determinada tarefa.

Em outras palavras, HERNANDEZ et al [24], a programação consiste de uma sequência de posições e ações de baixo nível que são especificadas ao controlador do sistema de uma maneira que o robô possa executar uma tarefa.

Sob uma perspectiva do programador a visão para a descrição da tarefa pode mudar. Segundo AMBLER [02], para a descrição das tarefas de robôs, o programador tem duas decisões básicas a fazer:

- O que ele quer descrever ? (Aplicação)
- Como ele vai fazer a descrição ? (Método de Programação)

Quanto à aplicação que o programador quer descrever, o domínio da tarefa de um robô é implicitamente determinado por suas capacidades mecânicas e sensoriais [31].

Durante os últimos vinte anos estas capacidades têm provido aplicabilidade numa extensa faixa de atividades. Em manufatura os robôs vêm sendo utilizados em diferentes aplicações como por exemplo solda, alimentação de máquinas, pintura, polimento e montagem [04].

A definição de um sistema para programação é então um problema difícil, pois se, por um lado objetiva-se a facilidade de uso do sistema para que este seja utilizado por usuários menos experientes, por outro, deseja-se que o sistema seja expressivo o bastante para que o usuário utilize-o na descrição de diferentes tipos de tarefas complexas [31].

Um indicador útil da sofisticação de um sistema de programação de robô é, portanto, o nível de descrição da tarefa do qual é capaz, o que, segundo YONG *et al* [56], pode ser classificado em quatro níveis, como segue:

1. *Nível de Juntas*: exige a programação individual de cada articulação da estrutura do robô para alcançar as posições globais exigidas.

2. *Nível de Manipulador:* envolve a especificação dos movimentos do robô em termos de posições no mundo da sua estrutura. São usadas técnicas matemáticas para determinar os valores das juntas individuais para estas posições.

3. *Nível de Objeto:* requer especificação da tarefa em termos dos movimentos e posicionamentos de objetos dentro da área de trabalho do robô. Isto implica na existência de um modelo da instalação do qual possa ser extraída a informação para determinar as posições das juntas do manipulador necessárias.

4. *Nível do Objetivo:* envolve a especificação da tarefa numa forma mais geral, como por exemplo, "pintar interior da porta do carro". Requer um banco de dados inclusivo que

contenha não só um "modelo mundo" mas também conhecimento das técnicas de aplicação, como os dados de condições e métodos ótimos de pintura. Seriam requeridos algoritmos "dotados de inteligência" para interpretar as instruções e aplicar a base de conhecimento, de modo a produzir programas de robô aperfeiçoados.

Quanto aos métodos para a descrição das tarefas está presente uma variedade na literatura, cada qual tendo suas vantagens e restrições, como descrito na seção seguinte.

2.3.1 Os Métodos de Programação

A descrição da tarefa a ser executada pêlos robôs industriais é atualmente realizada de dois modos: através da programação "on-line"[2] e através da programação "off-line"[3]. Na programação em linha o robô está presente nas diferentes fases do processo de descrição das tarefas. Já na programação fora de linha concentra-se na minimização do envolvimento do robô no processo de planejamento e depuração dos programas, estando o robô presente somente nas fases de desenvolvimento e teste.

A figura 2.7 apresenta um diagrama esquemático dos diferentes métodos utilizados atualmente para a programação dos robôs industriais. Basicamente, a descrição da tarefa pode ser realizada através da **programação gestual**, onde a presença do robô se faz essencial no procedimento de programação, através da **programação textual,** cujo envolvimento do robô é menor no planejamento mais essencial na depuração do programa, ou através da **programação gráfica**, que constitui o método mais recente e no qual se realiza os procedimentos de programação e depuração com o uso de sistemas gráficos interativos [02, 04, 19, 24, 28, 30, 31, 39].

[2] Em Linha: Tradução para o Português

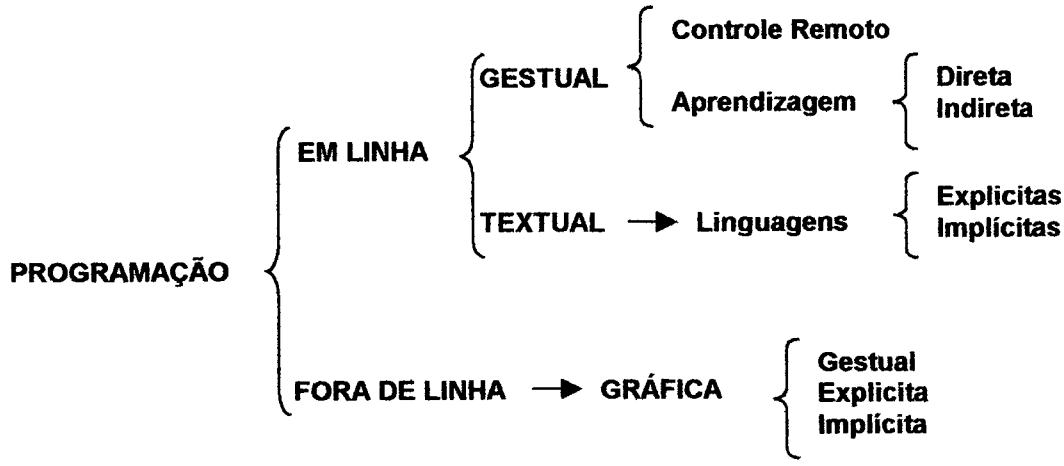

FIGURA 2.7: Métodos de Programação dos Robôs Industriais.

A seguir, temos uma breve explanação dos diferentes métodos citados acima para a descrição das tarefas de robôs industriais.

2.3.1.1 A PROGRAMAÇÃO GESTUAL

Os métodos de programação gestuais, que ainda são os mais usados e difundidos no ambiente industrial, estão concentrados na programação dos movimentos do robô. As tarefas são definidas pelo usuário através de um dispositivo de controle remoto ou através da aprendizagem, com a movimentação física do manipulador (Direta) ou com um dispositivo "mestre" de ensino (Indireta),e então armazenadas para posterior execuções [28].

Na programação gestual direta, as tarefas são criadas nas máquinas no chão de fábrica movendo o braço por um caminho de posições, e eventualmente velocidades e acelerações. A programação é realizada fisicamente, num movimento tipicamente de eixo (em coordenadas de juntas) ou de coordenada (em coordenadas operacionais), e entrando as funções de controle e

[3] Fora de Linha: Tradução para o Português

outros parâmetros ao longo do caminho. A depuração é realizada pela execução seletiva dos passos individuais da tarefa possibilitando ao operador percorrer a tarefa um passo desta de cada vez [25].

Como atrativos para o uso deste método podem ser citadas a falta de conhecimentos especializados do usuário e a possibilidade de analisar os resultados imediatamente após os procedimentos de programação, decorrente da geração automática de programas prontos para rodar [31].

Porém, estes métodos tradicionais de programação podem consumir um tempo excessivo. Freqüentemente, o tempo gasto aumenta de forma desproporcional ao crescimento de complexidade da tarefa. À medida que os robôs permanecem fora de produção, a programação "on-line" reduz a utilidade do robô, e por isso sua viabilidade econômica é substancialmente questionada. Para reduzir estes custos, é desejável reconfigurar células e reprogramar os robôs fora de linha [56].

O custo da reprogramação aumenta de acordo com o nível de complexidade da tarefa e com os diversos tipos de robôs utilizados. Sendo assim, a queda do tempo e do custo da reprogramação é um resultado do número de robôs envolvidos na aplicação e do grau de complexidade da tarefa [33].

Além do mais, é substancialmente mais efetivo gerar programas de robô antes que os protótipos fiquem disponíveis, sejam construídas as instalações, ou sejam instalados os robôs. O tempo típico para reprogramar uma aplicação de robô é de 1 a 2 semanas. Quando multiplicado por centenas de robôs por planta, porém, a programação on-line não fica prática, e para atingir metas, a programação do robô tem que começar antes que os protótipos se tornem disponíveis.

E ainda, dentre as desvantagens de sua utilização, pode-se citar que o procedimento necessita do robô para sua realização [28]; a dificuldade de interface com sensores [02]; e a não

compatibilidade com novas tecnologias baseadas em sistemas computacionais, tal como sistemas CAD/CAM e redes locais de computador [19].

2.3.1.2 A PROGRAMAÇÃO TEXTUAL

Aproximações posteriores realizaram os procedimentos de programação de robôs industriais através de programas escritos em linguagens de programação de robô, do mesmo modo como são escritas as linguagens de programação de computador. Por isso, são chamadas de linguagens de programação de robôs[11].

Nestes métodos, as instruções são escritas num programa linha orientado, numa determinada linguagem de programação de robô, e convertidas por um interpretador ou compilador. Num compilador, o programa é executado em alta velocidade e a detecção de erros é possível. Num interpretador, um ambiente interativo e de depuração deve ser provido.

A introdução desta técnica de programação para os robôs proveu a oportunidade para executar certas funções importantes que os métodos tradicionais de programação, citados anteriormente, não podiam realizar prontamente. Estas funções incluem [19]:

- aumentar as capacidades para sensores, inclusive o uso de entradas e saídas, tanto analógicas como também digitais;
- aperfeiçoamento das capacidades de saídas para o controle de equipamentos externos;
- controle de lógica programável;
- computações e processamentos de dados semelhantes aos das linguagens de programação de computadores;
- comunicações com outros sistemas computacionais;

- a realização fora de linha de parte do procedimento de programação.

2.3.1.2.1 AS LINGUAGENS EXPLÍCITAS DE PROGRAMAÇÃO

As linguagens explícitas de programação estão concentradas na geração de todos os dados necessários para mover o efetuador do robô ao longo de um percurso requisitado para o cumprimento de uma tarefa específica [30]. Por serem orientadas para a descrição explícita das operações dos robôs, estas linguagens estão no nível de manipulador, conforme a classificação citada no início desta seção. Na literatura podem ser encontrados muitos exemplos, mas segundo CRAIG [11] estas podem ser classificadas em três diferentes grupos:

Linguagens Especializadas de Programação: estas linguagens de programação de robôs foram construídas pelo desenvolvimento de uma linguagem completamente nova, onde ao mesmo tempo em que estão endereçadas às diferentes áreas específicas de robótica, podem ser ou não consideradas como uma linguagem de programação de computadores. Um exemplo é a linguagem VAL desenvolvida para controlar os robôs industriais da "Unimation Inc".

Bibliotecas de Robôs para uma Linguagem de Computador existente: estas linguagens de programação de robôs foram desenvolvidas partindo de uma linguagem popular de programação de computadores, como por exemplo o Pascal ou Basic, e adicionando uma biblioteca de subrotinas específicas de robôs. Assim, o usuário edita um programa fazendo uso de um pacote de subrotinas pré-definidas para as necessidades específicas dos robôs. Como exemplos de implementações em BASIC pode-se citar o AR-BASIC da "American Robot Corporation" e o ROBOT-BASIC da "Intelledex" e, em PASCAL, o JARS, desenvolvido pelo "NASA's Jet Propulsion Laboratory".

Bibliotecas de Robôs para uma Linguagem de Computador nova e de propósito geral: estas linguagens de programação de robôs foram desenvolvidas primeiro através da criação de uma nova linguagem de computador de propósito geral e, então, fornecendo uma biblioteca de subrotinas pré-definidas específicas para os robôs. Exemplos de tais linguagens de programação de robôs são o AML desenvolvido pela "IBM " e o RISE desenvolvido pela "SILMA Inc".

2.3.1.2.2 AS LINGUAGENS IMPLÍCITAS DE PROGRAMAÇÃO

Nas linguagens implícitas de programação o usuário deve especificar sub-metas ao invés de especificar com detalhes cada ação que o robô deve tomar. Estes sistemas de programação têm a habilidade de desenvolver o planejamento da tarefa automaticamente, permitindo que o usuário esteja apto a incluir instruções no programa aplicativo em alto nível [11].

O programador especifica com algum formalismo a tarefa que deseja resolver e, então, o sistema de programação, com alguns planos gerais e com a descrição cinemática e geométrica do robô, gera as instruções de movimento que resolvem a tarefa [24].

A descrição das tarefas nestes sistemas é "orientada a objeto", ao invés de "manipulador orientada". Para tal, o sistema de programação deve possibilitar estabelecer relações espaciais entre os objetos envolvidos na tarefa e identificar os modelos geométricos destes objetos, de modo a automaticamente interpretar ao nível de manipulador as descrições das sub-metas [31].

Como vantagem destas linguagens pode-se citar a possibilidade de gerar programas de robôs mais eficientes e de maneira relativamente simplificada [31].

2.3.1.3 A PROGRAMAÇÃO GRÁFICA

A programação gráfica dos robôs industriais, que é um método relativamente novo, requer a existência de um modelo teórico do robô e do seu ambiente. O objetivo é usar este modelo para simular o modo como o robô se comportaria na vida real. Usando o modelo, podem ser construídos programas que, depois de uma interface satisfatória, são usados para dirigir o robô.

De modo geral, existem dois métodos para realizar a programação gráfica dos robôs industriais. Em ambos os métodos, o processo de programação e a simulação da tarefa são exibidos numa tela usando sistemas de coordenadas nos modelos de objetos e de robôs com relação a um sistema de coordenadas inerciais [11]. Um ambiente computacional gráfico interativo para a definição do posicionamento dos modelos de objetos e de robô envolvidos na tarefa é fornecido, possibilitando ao usuário realizar o planejamento das tarefas do robô para posterior execução.

No primeiro método o usuário é provido da habilidade de movimentar o modelo do robô no seu ambiente, de modo a definir as movimentações deste para o cumprimento de uma tarefa específica. O procedimento é idêntico ao da programação gestual , mas é realizado num modelo virtual ao invés da instalação real [02].

No segundo método, relações espaciais entre os objetos envolvidos e o robô são identificadas para o planejamento de tarefas específicas. Este método pode ser dividido nos diferentes níveis para a descrição da tarefa, citados no início da seção 2.3. Permitindo classifica-los, assim como na programação textual, em métodos gráficos explícitos, através de procedimentos de programação no nível de juntas e de manipulador, e em métodos gráficos implícitos, através de procedimentos nos níveis de objeto e de objetivo.

Uma primeira vantagem visível deste método consiste no fato de que os objetos envolvidos na atividade são descritos e armazenados apenas uma vez, podendo assim ser recuperados em qualquer momento para o planejamento das tarefas [39].

Adicionalmente às vantagens providas pela programação textual, os sistemas de programação gráfica possuem ainda as seguintes vantagens [30]:

1. Depuração gráfica do programa de instruções, o que reduz o tempo e o esforço da depuração no robô real.
2. Podem-se experimentar os programas antes que o robô industrial seja comprado. Também podem ser testados diferentes modelos de robôs na tela e selecionado o mais apropriado para a tarefa planejada.
3. O ambiente do robô pode ser visualizado enquanto são exibidos os movimentos do robô, de modo a evitar colisões.
4. Bancos de dados de sistemas CAD/CAM podem ser utilizados no planejamento e simulação de diferentes atividades.

Diversos sistemas, [08, 12, 14, 24, 25, 27, 29, 31, 38, 40, 43, 46, 47, 53, 56], utilizam este último método para a programação das aplicações dos robôs.

O uso da depuração gráfica interativa permite realizar os procedimentos de programação de maneira amigável ao usuário, testando diferentes análises de projetos para a aplicação, permitindo a geração de programas otimizados e precisos, aumentando a produtividade e a qualidade nas aplicações de robôs industriais [30].

2.3.2 A PROGRAMAÇÃO FORA DE LINHA

Como define GROOVER [19], "na automação programável, diferindo da automação rígida, o equipamento de produção é projetado com a capacidade para mudar a seqüência de operações para acomodar as diferentes configurações de produto. A seqüência de operação é controlada por um programa que é um conjunto de instruções codificadas de forma que o sistema possa ler e interpreta-las. Os programas novos podem ser preparados e introduzidos no equipamento para produzir produtos novos. Para produzir cada lote novo de um produto diferente, o sistema deve ser reprogramado com o conjunto de instruções de máquina que correspondem ao novo produto. Este procedimento de comutação leva tempo. Por conseguinte, o ciclo típico para um determinado produto inclui um período durante qual a reprogramação e os ajustes são feitos, seguido por um período no qual o produto é produzido". Exemplos de automação programável incluem as máquinas ferramentas numericamente controladas e os robôs industriais.

A programação fora de linha pode ser considerada como o processo pelo qual os programas de robôs são desenvolvidos, parcialmente ou completamente, sem requerer o uso do próprio robô [56].

Muitas aplicações atuais de robô envolvem processos de produção em massa, como por exemplo, a atividade de solda em linhas de produção de automóveis onde o tempo de reprogramação deve ser mínimo. No entanto, tem havido um largo crescimento no número de robôs em atividades de fabricação e, como ressaltam HAUTAU et al [23], para ser possível a produção no campo de séries pequenas e médias, onde os tempos de reprogramação podem ser significativos, um sistema de programação fora de linha é essencial para as aplicações de robôs.

Como sugerido por MORRIS [33], este campo foi previamente considerado bastante não prático, em face das estruturas de "hardware" de computadores existentes e da falta de pacotes de "software" comercialmente produzidos para o engenheiro de controle.

Atualmente a técnica permite que o programa de instruções seja preparado num terminal de computador distante e carregado ao controlador de robô para a execução. Esta é a vantagem significante da verdadeira programação fora de linha, onde o tempo de manutenção para reprogramar seria minimizado, sendo permitido, assim, que o robô continue em produção ininterrupta. [19].

O crescente nível de complexidade das aplicações de robô torna ainda mais atraentes as vantagens associadas à programação fora de linha.

Estas foram sumarizadas a partir das referências [06, 08, 12, 27, 28, 29, 32, 33, 34, 38, 43, 47, 50, 56] , como se segue:

1. ***Redução do tempo de robô fora de produção***: o robô ainda pode estar em produção enquanto sua próxima tarefa está sendo programada. Isto habilita que a flexibilidade do robô seja utilizada mais efetivamente. O tempo de planta é de robô é muito valioso para programar robôs no chão da fábrica. Por exemplo, as plantas de automóveis simplesmente não podem dispor do tempo necessário para reprogramar 200 ou mais robôs toda vez que uma mudança de engenharia é realizada.

2. ***Remoção do programador de ambientes potencialmente perigosos e insalubres***: quanto mais o desenvolvimento de programa é realizado longe do robô, mais se reduz o tempo durante o qual o programador está em risco de comportamento aberrante do robô. Ainda, a programação na planta de fábrica simplesmente não é

conveniente para produção dos programas até mesmo para uma aplicação simples. É muito complexo se programar em um ambiente onde há muitas distrações e onde a concentração é dificultada ao máximo.

3. ***Sistema de programação simples:*** o sistema fora de linha pode ser usado para programar uma variedade de robôs sem a necessidade de saber as idiossincrasias de cada controlador de robô. Estas são levadas aos cuidados de pós-processadores apropriados o que minimiza a quantia de treinamento necessário para os programadores de robôs.

4. ***Integração com sistemas CAD/CAM existentes***: isto permite à interface ter acesso ao banco de dados de peças padrão, limitando a quantia de captura de dados necessária ao sistema fora de linha. A centralização do programa de robô dentro dos sistemas de CAD/CAM habilita o acesso a outras funções industriais como o planejamento e o controle.

5. ***Simplificação de tarefas complexas:*** a utilização de uma linguagem de computador de alto nível para o sistema fora de linha facilita a programação de tarefas mais complexas em robótica.

6. ***Verificação de programas de robô:*** os sistemas CAD/CAM existentes ou o próprio sistema fora de linha podem ser usados para produzir um modelo sólido real do robô e da instalação. O "software" de simulação pode ser usado então para provar, por exemplo, tarefas livres de colisão antes da geração do programa de robô.

Em essência, a programação fora de linha constitui um vínculo essencial para o CAD/CAM. O sucesso em seu desenvolvimento resultaria em um uso mais difundido de robôs industriais e também aceleraria a implementação de sistemas flexíveis de manufatura (FMS) na indústria [27].

Apesar dos diversos sistemas existentes empregarem métodos de programação diferentes, eles contêm certas características comuns essenciais para a programação fora de linha. A lista a seguir fornece as exigências que foram identificadas por YONG et al [56] como importantes em um sistema fora de linha para a programação de robôs industriais:

1. **Conhecimento do processo** ou da tarefa a ser programada;

2. **Modelo tridimensional**, isto é, dados em descrições geométricas de componentes e as relações entre estes dentro do espaço de trabalho;

3. **Conhecimento** da **geometria**, da **cinemática** (inclusive as restrições de articulação e perfis de velocidade), e da **dinâmica** do robô;

4. Um **sistema de computação** com um **método para programação** de robôs que utiliza dados de (1), (2), e (3). Tal sistema poderia ser com fundamentos gráficos ou textuais;

5. **Verificações de programas** produzidos por (4). Por exemplo, conferindo violações de restrições das juntas de robô e descoberta de colisão dentro do espaço de trabalho;

6. **Interface apropriada** de modo a permitir a **comunicação dos dados** gerados no sistema de controle do sistema fora de linha para vários controladores de robô. A escolha

de um robô com um controlador satisfatório (i.e., que possa aceitar dados gerados fora de linha) facilitará a interface;

7. **Efetiva interface homem/máquina**. Implícita na programação fora de linha está a remoção do programador do robô. Para permitir a transferência efetiva das habilidades dele a um sistema fora de linha, é crucial que uma interface de programação usuário amigável seja incorporada.

Cada exigência tem suas próprias dificuldades inerentes, mas os problemas principais surgem quando são feitas tentativas para generalizar as características funcionais. Embora a generalização (para atender diferentes tipos de robôs e aplicações) seja necessária para fazer o sistema mais efetivo, também é importante assegurar que os aumentos correspondentes em complexidade não inibam o uso funcional do sistema [56].

CAPÍTULO 3

SISTEMAS CAD/CAM E A PROGRAMAÇÃO DE ROBÔS INDUSTRIAIS

3.1 INTRODUÇÃO

O objetivo principal do capítulo é realizar um levantamento do estado da arte do desenvolvimento de sistemas gráficos interativos que fazem uso das técnicas de integração das fases de projeto e manufatura especificamente para a programação de aplicações de sistemas robóticos.

De modo a fundamentar o estudo da integração de sistemas CAD/CAM para a programação de tarefas de robôs industriais, é realizada inicialmente, na seção 3.2, uma revisão genérica sobre os sistemas gráficos interativos. Neste sentido, serão apresentadas algumas definições, conceitos e características importantes no uso e no desenvolvimento de programas aplicativos gráficos interativos para gerar as interfaces de comunicação entre o usuário e o computador.

Na seção 3.3 é feita uma breve abordagem de algumas definições e características dos programas aplicativos gráficos interativos que fazem uso das técnicas de projeto e fabricação

com o auxílio do computador. O interesse no estudo destas ferramentas é o desenvolvimento de técnicas para a automatização do planejamento e da programação dos processos produtivos aplicados aos robôs industriais.

Finalmente, realiza-se uma revisão, na seção 3.4, dos conceitos e características destas ferramentas tecnológicas para o planejamento, simulação e programação de robôs industriais considerando sua importância para o desenvolvimento e integração de automação da produção industrial.

3.2 A Computação Gráfica Interativa

A computação gráfica foi iniciada logo após a introdução dos computadores, através da exibição de dados em cópias impressas e em telas com tubos de raios catódicos. Hoje em dia, esta é usada em diferentes áreas da indústria, da educação, do entretenimento, do governo e do comércio. A lista de aplicações é enorme e está crescendo rapidamente conforme os computadores com capacidades gráficas tornam-se disponíveis como um produto acessível no mercado.

Atualmente, como FOLEY [17] caracteriza em seu livro, a computação gráfica interativa cresceu para incluir: "a criação, o armazenamento e a manipulação de modelos e imagens de objetos, preocupando-se com a síntese da imagem destes objetos imaginários ou reais a partir de modelos baseados em computador". Estes modelos vêm de diversos e crescentes campos, e incluem modelos físicos, matemáticos, de engenharia e até mesmo estruturas conceituais (abstratas) e fenômenos naturais.

A computação gráfica atualmente é muito interativa, podendo o usuário controlar o conteúdo, a estrutura e a aparência dos objetos e de sua exibição através de dispositivos de entrada, tal como o teclado e o mouse. Com a revolução, nas duas últimas décadas, da "cultura

de computador", reluzida pela Apple Macintosh e o IBM PC e seus clones, até mesmo as crianças da pré-escola estão confortáveis com as técnicas de interfaces gráficas interativas, como a metáfora da "desktop" [1], ilustrada na figura 3.1, para manipulação de janelas e seleção de menus e ícones com o mouse. Deste modo, a interação gráfica vem substituindo a maioria das interações textuais com os terminais alfanuméricos.

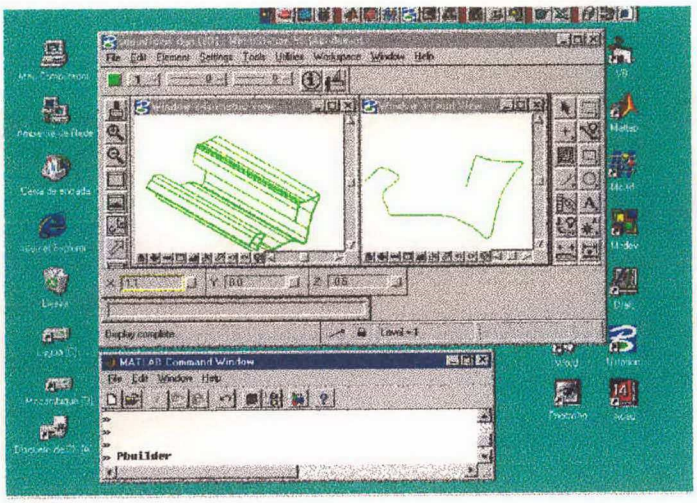

FIGURA 3.1: "Desktop" do Sistema Operacional Windows 95.

Ao mesmo tempo em que a computação gráfica interativa tornou-se comum nas interfaces com os usuários e na visualização de dados e objetos, outro avanço de extremo valor consiste na visualização de objetos 3D, o que tornou as imagens consideravelmente mais realistas (Figura 3.2, a seguir). As técnicas que eram experimentais no início dos anos oitenta são agora práticas padrão. Os mais simples tipos de pseudo-realismo, que por imagem tomavam horas do tempo de computador nos anos oitenta, são agora realizados rotineiramente, com taxas de animação em computadores pessoais.

O fator determinante para o uso de programas aplicativos gráficos interativos se deve ao fato de que estes, como ressalta *GROOVER* et al [21], "proporcionam uma efetiva comunicação

[1] O conceito de "desktop" tornou-se uma metáfora popular para a organização do espaço de tela. Através do sentido de um gerenciador de "janelas", o usuário pode criar, posicionar, e redimensionar áreas retangulares de tela que atuam como terminais gráficos virtuais, cada uma rodando uma aplicação e permitindo a troca entre diversos aplicativos.

homem/máquina, sendo o principal facilitador desta interação". A visualização científica tornou-se um campo importante nos anos oitenta, quando cientistas e engenheiros concluíram que não podiam interpretar a extraordinária quantidade de dados, produzidos na execução de programas aplicativos nos computadores, sem sumarizar os dados e salientar as tendências e fenômenos em diversos tipos de representação gráfica.

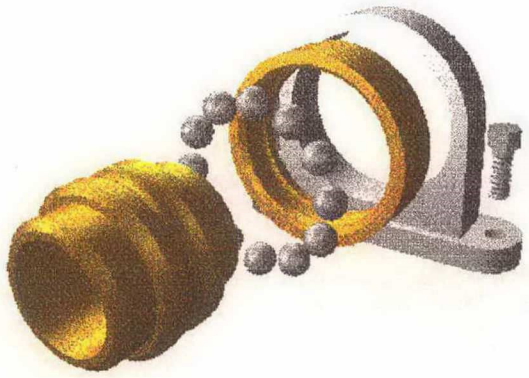

FIGURA 3.2: Exemplo da capacidade de visualização dos sistemas de computação gráfica (Imagem obtida da biblioteca do Microstation95).

FOLEY [17] faz algumas considerações importantes à respeito do uso dos gráficos interativos. Segundo ele, os gráficos provêem um dos meios mais naturais de comunicação com o computador, tanto que em muitos processos a informação que uma figura pode dar é virtualmente indispensável. O autor considera a computação gráfica interativa o mais importante meio para produção de imagens desde a invenção da fotografia e da televisão, e ainda ressalta a vantagem de poderem ser feitas imagens não apenas de objetos concretos, do "mundo real", mas também abstratos, tal como superfícies matemáticas em 4D, e dados que não têm nenhuma geometria inerente. O autor também constata que a computação gráfica não está mais confinada às imagens estáticas. Apesar das imagens estáticas serem um bom meio de comunicar informação, a variação dinâmica de imagens é freqüentemente melhor. Esta constatação é especialmente válida para fenômenos variáveis no tempo, tanto reais quanto abstratos. Segundo

o mesmo autor, este é o mais importante e crescente movimento na computação gráfica, e diz respeito à modelagem de objetos não apenas para a obtenção de suas figuras. O interesse está crescendo na descrição da variação temporal da geometria e do comportamento de objetos 3D. A computação gráfica está cada vez mais ligada à modelagem, à simulação e à animação, de modo a criar objetos que pareçam e se comportem da maneira mais realista possível.

Principalmente nos últimos anos, este campo tem sido beneficiado pela constante redução na relação preço/performance de hardware e do desenvolvimento de pacotes gráficos de alto-nível, independentes dos dispositivos de exibição, o que ajuda a fazer a programação gráfica racional e direta. Hoje, a computação gráfica permite uma intensa interação com uma larga faixa de usuários de computadores, acentuando significativamente a habilidade de entender os dados, perceber as tendências, e visualizar objetos reais e imaginários com a intenção de criar "mundos virtuais" que possam ser explorados com pontos de vista diversos. A comunicação mais eficiente, através das interfaces gráficas interativas e amigáveis aos usuários, possibilita obter resultados e produtos com maior qualidade e precisão, maior produtividade, e custos de projeto e análise reduzidos.

FOLEY [17] realiza uma abordagem completa do campo, reunindo aproximadamente 700 referências relativas à área, combinando conceitos atuais e aplicações práticas da computação gráfica, e explorando as perspectivas do usuário, do programador e do projetista de hardware. A partir desta revisão bibliográfica realizada pelo autor, e com o intuito de revisar alguns conceitos básicos envolvidos no uso e desenvolvimento das interfaces gráficas interativas, foram reunidas nas próximas subseções informações referentes à estrutura das interfaces gráficas interativas (3.2.1), bem como os dispositivos físicos envolvidos (3.2.2) e algumas características do seu projeto e implementação (3.2.3).

3.2.1 Estrutura Conceitual das Interfaces Gráficas Interativas

Segundo FOLEY [17], a estrutura conceitual mostrada na Figura 3.3 pode ser usada para descrever basicamente quase todas as interfaces gráficas interativas.

No nível de hardware (não demonstrado explicitamente no diagrama, porém comentado na seção 3.2.3), o computador recebe entradas de dispositivos de interação, e gera imagens de saída para o dispositivo de exibição.

O software têm três componentes básicos. O primeiro é o ***programa aplicativo***: este cria, armazena e recupera o segundo componente, o ***modelo aplicativo***, onde estão representados os dados ou objetos para serem retratados na tela de exibição. No modelo aplicativo têm-se todos os dados, os objetos e as relações entre estes que são relevantes para as rotinas de exibição e interação com o programa aplicativo, ou qualquer outro módulo não gráfico de pós-processamento.

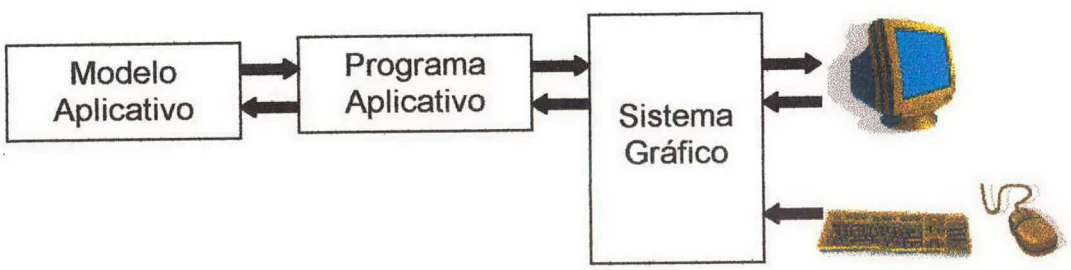

FIGURA 3.3: Estrutura Conceitual das Interfaces Gráficas Interativas.

O programa aplicativo controla as entradas do usuário. Este produz exibições que são enviadas ao terceiro componente, o sistema gráfico, através de uma série de comandos de saídas gráficas que contêm uma completa descrição geométrica do que deve ser visto e os atributos de descrição de como os objetos devem ser vistos. O ***sistema gráfico*** é responsável pela produção

da imagem a partir desta descrição e pela passagem das entradas do usuário ao programa aplicativo para o processamento [17].

FOLEY [17] define que a relação entre o modelo aplicativo, o programa aplicativo e o sistema gráfico pode ser ilustrada em diagrama, como na figura 3.4, a seguir, ilustrando os componentes lógicos, mas não necessariamente a estrutura de componentes do programa. Neste diagrama o programa aplicativo está subdividido em cinco subsistemas, rotulados de (a) a (e), e tendo estes as funções como detalhadas na legenda.

(a) Cria, modifica, e mantém o modelo aplicativo através da adição, exclusão ou substituição de informações contidas neste.
(b) Varre o modelo aplicativo para extrair informações para exibição.
(c) Varre o modelo aplicativo para extrair informações usadas na análise de comportamento/performance do modelo
(d) Exibição dos dados de um modelo ou de uma análise, e das ferramentas de interface com o usuário (menus, caixas de diálogo).
(e) Execução de tarefas não necessariamente envolvendo o modelo aplicativo ou a exibição deste.

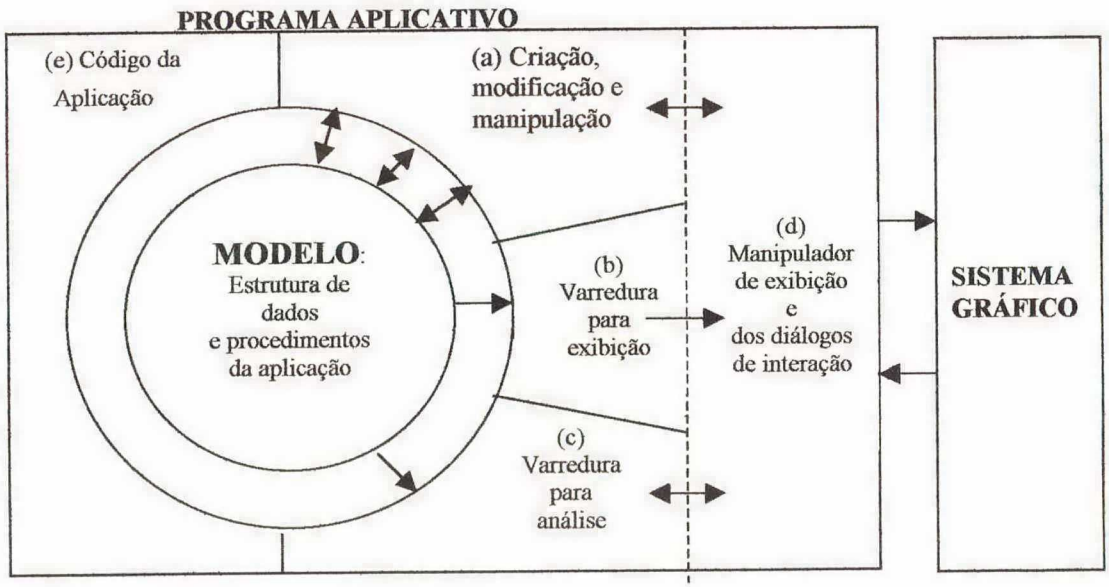

FIGURA 3.4: O Programa Aplicativo e seus Subsistemas.

HORDESKI [26] defende que o desenvolvimento das linguagens de *alto nível* possui um grande potencial quando comparado às de ***baixo nível***, uma vez que estas possibilitam ao programador utilizar tipos simples de instruções que envolvem grande quantidade de instruções de códigos de máquina. Segundo este, os programas aplicativos são escritos para fazer algum tipo de tarefa útil, tal como a visualização de dados ou a execução de cálculos, e devem ser

traduzidos em linguagem de máquina para controlar o circuito do computador. Esta tradução é realizada utilizando um compilador ou interpretador[2].

Segundo FOLEY [17], a principal tarefa do projetista de um programa aplicativo gráfico interativo consiste na especificação das classes de itens de dados ou objetos que devem ser gerados e representados, e em como o usuário e o programa aplicativo devem interagir de modo a criar e modificar o modelo e sua representação visual. A grande parte das tarefas dos programadores consiste, então, na criação do modelo, edição do modelo e controle das interações do usuário.Quanto aos dados do modelo, o autor constata que estes podem ser rudimentares como um vetor de pontos ou complexo como uma "lista de ligações" representando uma estrutura de dados em rede ou um banco de dados relacional. E ainda, os objetos armazenados no modelo podem diferir muito na quantidade de geometria intrínseca necessária para descrevê-lo.

Segundo HORDESKI [26], em muitas ocasiões é mais fácil, barato e seguro realizar experimentos com um modelo do que com uma entidade real. Em muitas situações de engenharia, a modelagem e a simulação constituem-se no único método viável para projetar e instruir um sistema. Os modelos são uma descrição dos componentes e dos processos que juntos especificam a estrutura e o comportamento de um sistema. Segundo este, a principal preocupação na modelagem consiste no projeto e implementação de um modelo que, utilizando quantidades precisas, reflita adequadamente as propriedades do sistema. Se possível, o modelo deve ser verificado com respeito ao sistema real para que o projetista tente ajustar o modelo para aperfeiçoar sua exatidão e utilidade, através das variáveis de entrada e de saída, de parâmetros ajustáveis, e funções especiais definidas pelo usuário para permitir que o modelo tenha características orientadas a um propósito especial em particular.

---

[2] O compilador ou interpretador opera na linguagem de alto nível e produz a versão do programa chamada "programa objeto". O programa objeto é construído em código de máquina de modo a ser executado diretamente pelo computador.

3.2.2 Dispositivos Físicos dos Sistemas Gráficos Interativos

Nesta seção descrevemos os dispositivos físicos principais num sistema. Diversas publicações, por exemplo [17], [26] e [32], apresentam um hardware típico de sistema gráfico como sendo constituído de três componentes principais, como os ilustrados na Figura 3.5. O computador, que consiste no processador e na memória associada, os dispositivos de entrada e os dispositivos de saída. O processador ou unidade central de processamento (CPU) executa os cálculos matemáticos exigidos para as funções de cálculo e processamento de dados. Conectado ao processador está a memória do computador, aonde são armazenados o programa aplicativo (software) e os modelos aplicativos.

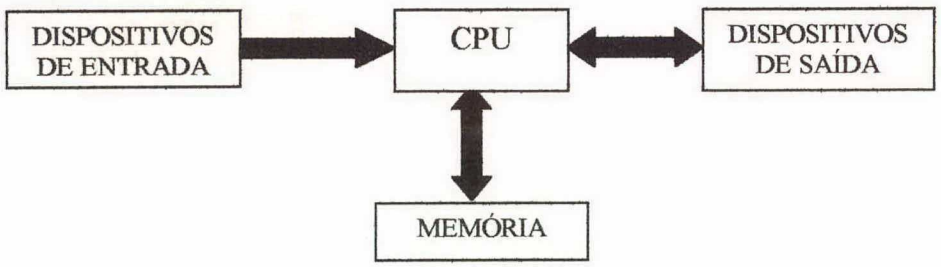

FIGURA 3.5: Componentes Principais do Hardware de Sistemas Gráficos.

Os periféricos de interação dos sistemas gráficos podem ser classificados em dispositivos de entrada e dispositivos de saída. Os dispositivos de entrada são utilizados para entrar com as informações no computador, podendo ser uma caneta de luz, um teclado alfanumérico, uma mesa digitalizadora, um "mouse", uma digitalizadora tridimensional, ou ainda, como na robótica, alguns sensores. Já os dispositivos de saída são usados para apresentar imagens gráficas a partir do computador, e podem ser divididos em dispositivos para a representação temporária de imagens, como os terminais gráficos, e dispositivos para a representação permanente de imagens, como as plotadoras e impressoras.

Segundo FOLEY [17], a avaliação das vantagens e desvantagens dos vários dispositivos de interação pode ser feita em três níveis: dispositivo, tarefa e diálogo. O nível de dispositivo está concentrado nas características do dispositivo físico *"per se"*, e não nas características de uso do dispositivo controlado pelo software. Neste nível, por exemplo, nota-se que a forma de um mouse pode ser mais confortável de segurar do que a de um outro, ou ainda, que uma mesa digitalizadora ocupa mais espaço do que um mouse. No nível de tarefa, deve-se comparar as técnicas de interação usando diferentes dispositivos para realizar uma mesma tarefa. Por exemplo, um usuário mais experiente pode preferir utilizar o teclado ao invés de menu. Finalmente, no nível de diálogo, devem ser consideradas não apenas as tarefas de interação em individual, mas também sequências destas tarefas. Por exemplo, os movimentos da mão podem ser mais ligeiros para o posicionamento do cursor com o "mouse" do que com um "joystick".

3.2.3 Projeto e Implementação de Interfaces Gráficas Interativas

O interesse na qualidade da interface usuário/computador é recente no estudo formal dos computadores. A ênfase até os anos oitenta era concentrada na otimização de dois escassos recursos de hardware, o tempo de computador e a memória. A eficiência do computador e do programa era o maior objetivo. Hoje em dia, com a diminuição dos preços de hardware e o aumento da potencialidade dos ambientes gráfico-orientados em computadores, pode-se dispor da otimização da eficiência do usuário ao invés da exclusiva eficiência do computador.

A qualidade da interface com o usuário determina, de certo modo, se o usuário vai desfrutar ou desprezar um sistema e, conseqüentemente, se este vai ter sucesso ou fracassar no mercado. Decerto, em aplicações de risco como o monitoramento do controle de trafego aéreo ou de uma planta de energia nuclear, uma interface pobre pode contribuir para, ou até causar, um desastre de proporções catastróficas.

A metáfora da "desktop" para a interface com usuário, com suas "janelas", ícones e menus, é popular porque é de fácil aprendizado e requer pouca habilidade e experiência. A maioria dos usuários não são programadores e não têm nenhuma afinidade com o antigo, e de difícil aprendizado, estilo de interface teclado/orientada.

Na revisão realizada por FOLEY [17], o autor faz algumas apreciações com relação ao projeto de interfaces gráficas interativas, relevando alguns objetivos chaves e considerações importantes, e expondo características dos diálogos de interação e dos estilos de interfaces. Algumas destas serão apreciadas a seguir.

Inicialmente, FOLEY [17] ressalta a importância de o projetista de uma aplicação gráfica interativa ter sensibilidade ao desejo dos usuários de utilizar interfaces de fácil aprendizado, porém funcionais. Embora seja uma declaração que pareça trivial, segundo este "o primeiro passo no projeto de uma interface consiste na ***definição da sua finalidade***". A definição insatisfatória das necessidades pode significar a ruína de um projeto de uma interface ainda nos estágios iniciais. O autor defende que o entendimento das ***necessidades do usuário*** pode ser proporcionado, por exemplo, através do estudo do método atual de solução do problema em consideração, isto é, como as tarefas em questão são executadas. O objetivo é entender ***o que*** os prováveis usuários atualmente fazem, e o ***porquê***, de modo a permitir desenvolver novas e melhores ferramentas, sendo uma típica estratégia realizar primeiramente a mímica de métodos existentes e tornar acessíveis novos métodos de modo que os usuários possam ser treinados ao longo do tempo com as novas capacidades prevenindo o treinamento e a resistência moral das forças de trabalho. As ***características do usuário*** também devem ser identificadas. Perguntas devem ser realizadas, como por exemplo: o usuário utilizará o sistema esporadicamente ou freqüentemente, em tempo integral ou não? O usuário será animado ou relutante para aprender a utilização do sistema? É importante que o projetista de um sistema lembre que o que ele gosta ou

quer não é necessariamente a mesma coisa que aqueles para os quais o sistema esta sendo desenhado desejam.

FOLEY [17] define alguns objetivos chaves no projeto de uma interface com o usuário: o aumento na velocidade de aprendizagem e na velocidade de uso, a redução na taxa de erro, o encorajamento da chamada de ajuda para o uso da interface, e o aumento de atrativos para os potenciais usuários e compradores.

O autor [17] também faz algumas considerações importantes que visam assegurar bons fatores humanos no projeto de interfaces com usuários. Segundo ele, as interfaces devem ***Ser Consistentes***, tendo como propósito básico prover ao usuário a generalização do conhecimento de um aspecto do sistema para outros aspectos do sistema, e devem ***Prover Realimentação***, característica essencial na interação do usuário com o computador, assim como é na interação entre dois indivíduos. E ainda, *Minimizar a* ***Possibilidade de Erro***, por exemplo, dificultando ou impossibilitando ao usuário realizar ações que não são permissíveis no contexto. O autor [17] também enfatiza a necessidade de ***Permitir a Reparação de Erro***, uma vez que existem amplas evidências experimentais de que as pessoas são mais produtivas se os seus erros podem ser corrigidos, devido ao fato de o usuário sentir mais a vontade para explorar o sistema sem medo de cometer falhas, encorajando o aprendizado das características do sistema. Segundo este, também é essencial ***Acomodar Níveis Múltiplos de Habilidades***, permitindo variar desde um usuário inexperiente ao que já trabalhou diversas horas com o sistema, e ainda ***Minimizar a Memorização***, invocando, sempre que possível, o reconhecimento do usuário de modo que não ocorra a memorização desnecessária.

O autor [17] define o diálogo entre o usuário e o computador como ponto chave central no projeto de sistemas interativos. Este ressalta que, "a computação gráfica habilita aos usuários de computador a utilização das imagens como uma modalidade de comunicação adicional. Entretanto, muitos dos atributos dos diálogos entre as pessoas devem ser preservados nos

diálogos entre o usuário e o computador. As pessoas que efetivamente se comunicam compartilham um conhecimento comum e um conjunto de presunções. As presunções e os conhecimentos devem ser do usuário e não do projetista de uma sofisticada interface entre o usuário e o computador".

Segundo FOLEY [17], através de uma ***tarefa básica de interação*** (BIT – Basic Interaction Tasks)[3], o usuário de um sistema interativo descreve uma unidade de informação significativa no contexto da aplicação. Já as ***tarefas compostas de interação*** (CITs – Composite Interaction Tasks)[4] são construídas a partir das BITs. O autor apresenta uma lista de diferentes técnicas de interação para cada tipo de tarefa básica, estando estas em constante ampliação.

Como constata FOLEY [17], três estilos de interfaces entre o usuário e o computador são comumente utilizados, sendo estes: o ***WYSIWYG***[5], onde a representação com a qual o usuário interage no dispositivo de exibição é essencialmente a mesma imagem criada pela aplicação. (exemplo típico de interface WYSIWYG são os processadores de texto); o de ***manipulação direta***, onde os objetos, os atributos e as relações que podem ser operadas são representados visualmente e invocados através de ações executadas nas representações visuais (tipicamente utilizando um mouse), estando o comando implícito à ação na representação visual; e o de ***ícones***, onde estes são representações pictóricas de um objeto, de uma ação, de uma propriedade ou outro conceito. Outras formas de diálogo também citadas, não intrinsecamente gráficas, mas que podem ser utilizadas em aplicações gráficas, são: os ***menus de seleção***, onde os usuários usam a memória de reconhecimento para associarem imagens visuais (textual ou icônica) com palavras e significados realmente familiares; as ***linguagens de comando***, tradicionalmente usadas na interface com o computador; a ***linguagem natural de diálogo***, que visa fazer com que

[3] Um conjunto completo de tarefas básicas de interação com os gráficos interativos engloba: o posicionamento, a seleção, a entrada de textos e a entrada de quantidades numéricas.[17]

[4] Atualmente existem três principais formas de CITs: as caixas de diálogos, utilizadas para especificar múltiplas unidades de informação; as de construção, utilizadas para criar objetos; e as de manipulação, usadas para remodelar objetos geométricos existentes.[17]

os computadores entendam comandos falados ou teclados numa língua como o inglês; e a de ***diálogo pergunta e reposta***, que é iniciada pelo computador, e o usuário deve responder dentro de uma gama restrita de possíveis respostas.

A tabela 3.1 demonstra uma comparação realizada por FOLEY [17] entre os estilos de interface com o usuário perante alguns requisitos importantes. Discussões mais aprofundadas podem ser encontradas nas referências pesquisadas por este autor.

	WYSI-WYG	*Manipulação Direta*	*Menu de Seleção*	*Linguagem de Comando*	*Linguagem Natural*	*Dialogo P/R*
Tempo de Aprendizado	Baixo	Baixo	Médio	Alto	Baixo	Baixo
Velocidade de Uso		Média	Média	Alta	Média	Baixa
Propensão ao Erro	Baixa	Baixa	Baixa	Alta	Alta	Baixa
Extensibilidade	Baixa	Baixa	Média	Alta	Alta	Alta
Destreza Datilográfica necessária		Nenhuma	Nenhuma	Alta	Alta	Alta

TABELA 3.1: Estilos de Interfaces e Requisitos para Avaliação.

Finalmente, FOLEY [17] ressalta a importância do projeto visual da interface gráfica, constituindo um importante elemento no desenvolvimento de sistemas gráficos interativos. Segundo ele, o projeto visual abrange todos os elementos gráficos de uma interface, incluindo, por exemplo, o planejamento da tela de exibição, o projeto de menus e o uso de cores. Para o autor, no projeto visual o empenho é realizado visando à obtenção da ***consistência*** dos códigos das aplicações, através de regras de organização visual das informações, e da ***clareza*** da combinação dos elementos visuais em objetos gráficos e ícones de alto nível, de modo a reforçar e enfatizar a organização lógica subjacente, tornando o significado de uma imagem facilmente aparente ao observador.

[5] WYSIWYG: abreviação de "what you see is what you get" , em inglês.

3.3 FERRAMENTAS DE AUXÍLIO AO PROJETO E À FABRICAÇÃO EM CAD/CAM

Como todo programa aplicativo gráfico interativo, os ingredientes básicos de um sistema CAD estão na forma de hardware e de software. Em geral, os programas de CAD exigem um sistema de computador potente o bastante para a criação e a manipulação de imagens 3D realísticas. O estado da arte dos softwares de CAD inclue o desenho e o detalhamento 2D e 3D, a modelagem sólida com operações "booleanas", a remoção de linhas escondidas, a modelagem de curvas e superfícies e a possibilidade da troca de dados entre sistemas em diferentes padrões.

O hardware exigido para o uso dos sistemas CAD costumava ser os dispendiosos sistemas "mainframe". Com o avanço da tecnologia de computador, os minicomputadores ficaram mais poderosos e baratos, tornando estes uma das principais plataformas para a maioria das aplicações de sistemas CAD. Os sistemas CAD baseados nas plataformas PC, em especial, tornaram-se uma escolha viável para o desenvolvimento de aplicações de CAD, principalmente devido ao seu baixo custo e alta performance, à disponibilidade de vendedores destes sistemas, com uma grande faixa de capacidades de performance e preço, numa arquitetura aberta que permite fácil manutenção, uma pequena ou inexistente necessidade de treinamento no uso do computador e do sistema operacional e, à quase inexistente necessidade de um administrador de sistema.

O uso do CAD vem sendo cada vez mais difundido nos recentes anos e, com rápidos avanços, o CAD hoje constitui uma ferramenta viável em engenharia, desde o projeto conceitual ao produto final. Com a aceleração da automação industrial, o CAD não está mais limitado à produção de sofisticados modelos 2D e 3D. O termo "sistema CAD/CAM" é usado se este suporta tanto as aplicações de fabricação como as de projeto.

Neste sentido, em um sistema CAD/CAM ideal, é possível adquirir as especificações de projeto do produto que residem no banco de dados do sistema CAD e convertê-las num plano de

processo para a fabricação do produto, conversão esta realizada automaticamente nos sistemas CAD/CAM.

GROOVER [19] apresenta uma introdução sucinta aos conceitos gerais envolvidos. Este autor define de maneira geral os conceitos básicos da idéia de integração de sistemas de projeto e fabricação. Como constata o autor, "o método de manufatura de um produto é uma função direta do seu projeto, sendo assim, na interface entre sistemas CAD e CAM, uma ligação direta é estabelecida entre as engenharias de projeto do produto e de fabricação tendo como objetivo não apenas a automatização de certas fases de projeto e de manufatura, mas também a transição do projeto para a manufatura". Para um melhor entendimento da aplicação da técnica iremos buscar alguns conceitos referentes à produção industrial visando o entendimento da área de aplicação destes sistemas.

GROOVER [19] define que em qualquer tipo de produção industrial, seja esta em massa ou não, algumas funções básicas devem ser executadas para converter materiais brutos em produtos acabados. Segundo o autor, numa firma engajada em fazer produtos discretos, estas funções são (Figura 3.6): o processamento, a montagem, o armazenamento e manipulação de material, a inspeção e teste, e o controle.

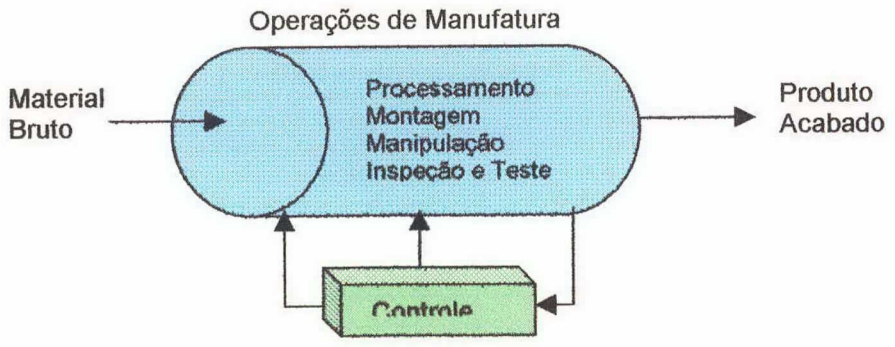

FIGURA 3.6: Modelo de Fábrica mostrando as cinco funções de manufatura.

Dentre as operações de manufatura, temos o controle, que é requerido para coordenar e regularizar as atividades físicas de fabricação, e as operações que entram em "contato direto"

com o produto, no caso as quatro primeiras funções. As empresas devem se organizar para cumprirem estas cinco funções, mais detalhadamente descritas no livro do autor em análise [19].

A partir deste modelo de fábrica, GROOVER [19] apresenta um modelo conceitual de manufatura onde as atividades físicas de fabricação relacionadas à produção numa fábrica, operações nas quais se tem um contato direto com o produto durante a manufatura, podem ser discriminadas e relacionadas às atividades de processamento de informações (Figura 3.7, a seguir).

As atividades físicas ocorrem dentro da fábrica tornando materiais brutos em produtos acabados, e as atividades de processamento de informação formam um anel ao redor destas, fornecendo os dados e o conhecimento necessários para produzir o produto satisfatoriamente, formando um ciclo de eventos que devem acompanhar as atividades físicas de produção, mas não necessariamente em contato direto com o produto.

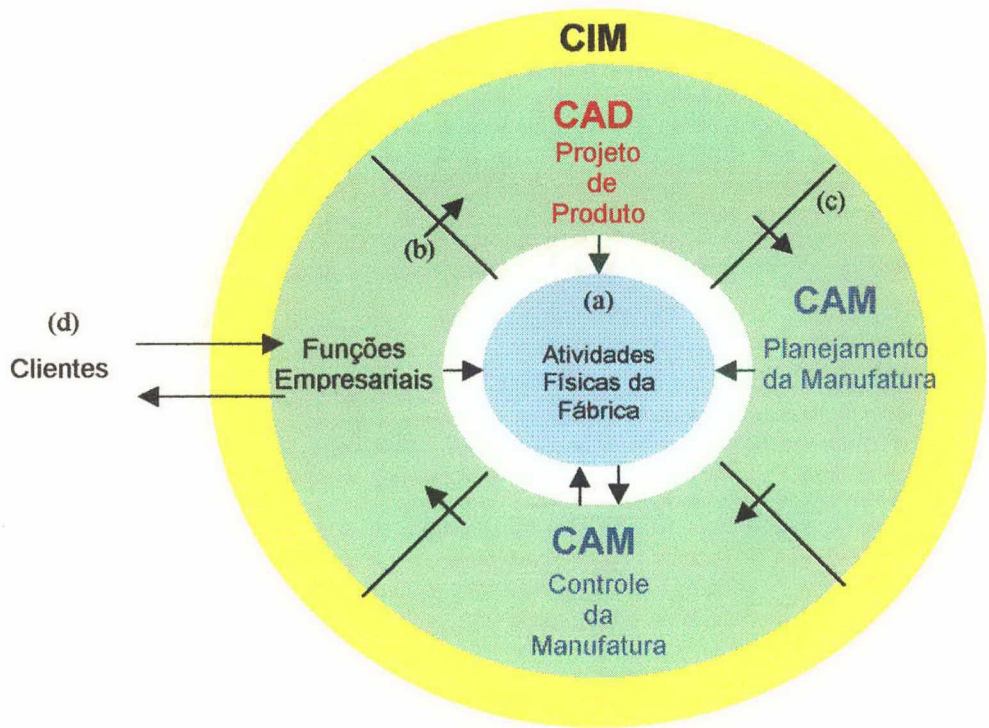

FIGURA 3.7: Um Modelo Conceitual de Manufatura: (a) a fábrica como um núcleo onde as atividades físicas de manufatura são executadas (Figura 3.6), (b) o fluxo das atividades de processamento de informação formando um ciclo em torno do núcleo, (c) escopo dos sistemas CAD e CAM, e (d) as trocas externas com clientes e fornecedores.

A tarefa de coordenação de todas as atividades individuais requeridas para produzir o produto, realizar a sua montagem, e entregá-lo ao cliente é um problema de processamento de informação. Como ilustrado, as funções de processamento de informação incluem as atividades ***Empresariais***, as atividades de ***Projeto de Produto***, as atividades de ***Planejamento da Manufatura***, e as atividades de ***Controle da Manufatura***.

As atividades empresariais constituem o principal meio de comunicação com os clientes. Estas estão no início e no fim do ciclo de processamento de informação. Inclusas nesta categoria estão as vendas e a publicidade, as ordens de entrada, as notas fiscais aos clientes, entre outras. O termo CIM ("Computer Integrated Manufacturing") atualmente é empregado se estiverem integradas ao sistema CAD/CAM as atividades empresariais.

As funções de projeto de produto, de planejamento e de controle da manufatura serão brevemente revisadas nas próximas seções.

3.3.1 A FUNÇÃO DE PROJETO DE PRODUTO

Segundo ***SHIGLEY*** [48], um processo geral de projeto pode ser caracterizado como um processo iterativo, ilustrado na figura 3.8, a seguir, e que consiste nas seguintes seis fases:

O ***reconhecimento das necessidades*** envolve a realização da existência do problema e de que uma ação corretiva pode ser tomada na forma de uma solução de projeto, o que constitui uma ação criativa.

A ***definição do problema*** envolve a completa especificação do item a ser projetado incluindo as características físicas, a função, o custo, a qualidade, e a performance de operação.

A ***síntese e a análise*** estão muito relacionadas e são altamente iterativas no processo de projeto. Cada subsistema do produto deve ser conceituado pelo projetista, analisado, aperfeiçoado através de procedimentos de análise, projetado novamente, analisado de novo e

assim por diante. O processo deve ser repetido até que o projeto esteja otimizado com as restrições impostas pelo projetista.

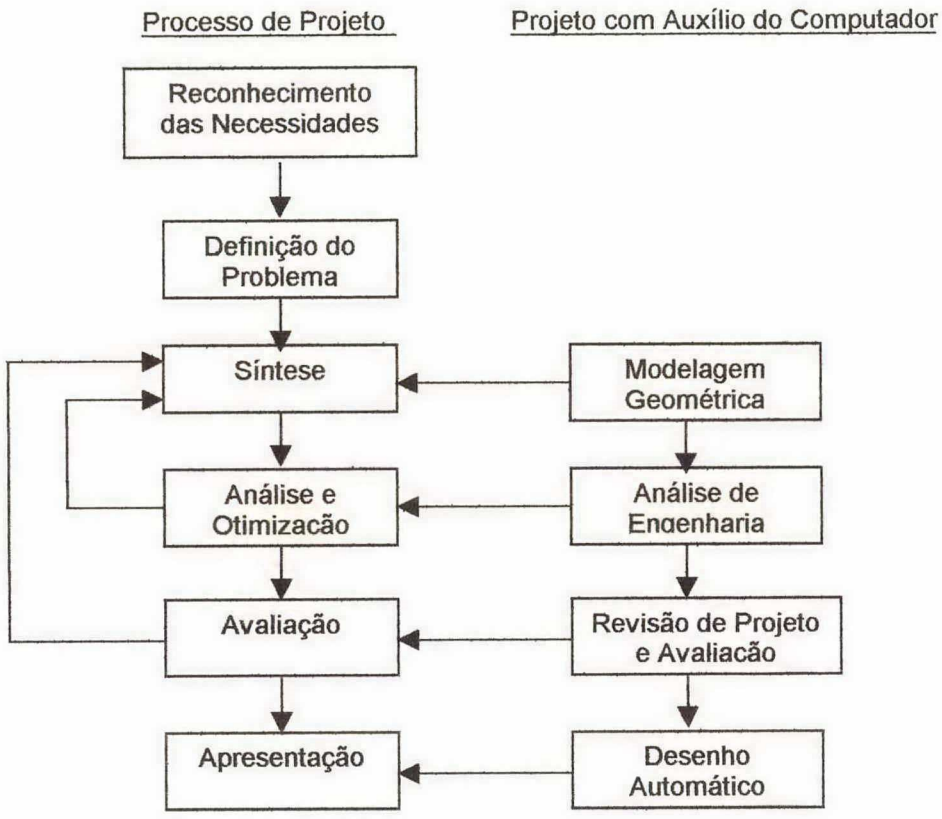

FIGURA 3.8: Processo de Projeto e o CAD.

A ***avaliação*** está concentrada na medição do projeto de acordo com as especificações estabelecidas na fase de definição do problema. Esta avaliação necessita, em alguns casos, da fabricação e teste de um modelo protótipo para acessar a performance em operação, a qualidade, a confiabilidade, e outros critérios.

Finalmente, a fase final, quando se dá a apresentação do projeto. A ***apresentação*** está concentrada na documentação do projeto por meio de desenhos, especificações de materiais, listas de montagens, e etc. Em essência, a documentação significa que o banco de dados de projeto foi criado.

Segundo ***GROOVER*** [21], as atividades realizadas nos atuais sistemas de computador para o apoio ao projeto (CAD), também ilustradas na figura 3.8, podem ser benéficas nas seguintes quatro fases do processo de projeto.

A ***modelagem geométrica*** está envolvida no uso dos sistemas CAD para o desenvolvimento de uma descrição matemática da geometria de um objeto. A descrição matemática, chamada de modelo, está contida na memória do computador e pode ser bidimensional ou tridimensional. Isto permite ao usuário do sistema CAD exibir a imagem do objeto num terminal gráfico e executar certas operações no modelo. Estas operações incluem a criação de novos modelos, a manipulação da imagem na tela, a possibilidade de se aproximar certas características da imagem e outras operações. Estas capacidades permitem ao projetista construir o modelo de um novo produto ou modificar um modelo existente.

Os desenhos técnicos 2D são construídos através da combinação de entidades geométricas 2D básicas, como os pontos, as linhas, os arcos, os círculos e as curvas. Essas entidades geométricas básicas são referidas como primitivas e são essenciais na atividade de síntese de projeto com auxílio dos sistemas CAD.

Quanto ao espaço 3D, em geral, existem atualmente três tipos básicos de modelagem geométrica utilizados nos sistemas CAD. [17, 44]

Os ***Modelos "Wireframe"*** representam um objeto definindo as suas arestas, que são linhas que conectam seus vértices. A maioria dos sistemas CAD comerciais usam linhas, arcos e círculos para representar as arestas.

A ***Modelagem por Superfícies*** é uma técnica de modelagem gráfica utilizada para definir e descrever superfícies. Uma variedade de modelos de produtos pode ser desenvolvida utilizando esta técnica de modelagem, que tem como principal vantagem o fato de definir perfeitamente o contorno dos produtos.

A ***Modelagem Sólida*** constitui uma técnica mais recente de modelagem geométrica, onde se tem uma definição não apenas da parte externa do produto mas também da interna, possibilitando a obtenção de propriedades destes, como por exemplo, sua massa, momento de inércia, centro de gravidade e volume. Neste tipo de modelagem as superfícies são utilizadas para definir os contornos dos objetos. Por exemplo, o contorno de um cubo ou de uma esfera, pode ser representado através de superfícies planas ou de revolução, respectivamente.

A modelagem "wireframe" é um método útil para descrever um objeto e é uma aproximação comum em projeto. Comparados aos modelos de superfícies e aos modelos sólidos, os modelos "wireframe" são uma representação geométrica 3D simples, que utiliza pouco tempo de computador para processar a imagem. Todavia, estes modelos não contem informações completas das superfícies e, ainda, podem ser ambíguos na representação de geometrias complexas.

Na década de 80 os sistemas CAD eram essencialmente modeladores geométricos, isto é, permitiam que peças ou conjunto de peças fossem representados tridimensionalmente através das técnicas acima citadas. Porém, em projeto e fabricação, uma peça ou conjunto devem ser representados em um sistema CAD com muito mais informações do que aquelas necessárias para sua definição geométrica. Num mundo real trabalhamos com diferentes tipos de materiais, tolerâncias, acabamentos superficiais, entre muitas outras características.

Como constata PEDRA [37], atualmente temos uma nova geração de sistemas CAD, denominados "Modeladores Tecnológicos", que permitem a associação das informações acima descritas aos modelos, trazendo ganhos de produtividade antes não imaginados, pois etapas posteriores como análises de processos ou estruturais, processos de usinagem, geração de listas de materiais e muitas outras têm uma fonte de alimentação de dados única e previamente definida. As "características" ("features") geométricas tornaram-se tecnológicas, agregando mais informações aos produtos. Um furo rosqueado terá, além de seus parâmetros geométricos, uma

ferramenta e um processo de usinagem associados, representando mais um ganho de produtividade. Os sistemas com múltiplos "Solucionadores" geométricos, como por exemplo, o paramétrico, o variacional e o adaptativo, que atuam de forma integrada e inteligente, se encarregam de ativar o melhor para uma dada situação de projeto. As distinções atuais entre sólidos e superfícies, que hoje em dia significam conversões problemáticas entre modelos gerados por uma ou outra técnica, deverão ser eliminadas. Os sistemas CAD de nova geração deverão tratar indistintamente a geometria criada, seja ela "manifold" ou "non-manifold".

Depois que a síntese de projeto é desenvolvida, alguma forma de ***análise de engenharia*** deve ser executada como parte do processo de projeto. A análise pode tomar a forma de cálculos de tensão e de resistência, análise de transferência de calor ou simulação dinâmica. As computações requeridas são quase sempre complexas e consomem tempo, sendo que antes do advento do projeto com auxílio do computador, estas eram simplificadas ou omitidas dos projetos. A disponibilidade de software CAE ("Computer Aided Engineering") para auxílio na análise de engenharia nos sistemas CAD incrementa sensivelmente a disposição e a habilidade para a execução mais completa de um projeto proposto.

Os procedimentos de ***avaliação e revisão de projeto*** podem ser beneficiados através do uso de um sistema de projeto com auxílio do computador. Algumas das capacidades dos sistemas CAD que auxiliam na avaliação e revisão de um projeto proposto incluem: as rotinas automáticas de ***Dimensionamento***, que determinam medições precisas de distâncias; as rotinas de teste de ***Interferência***, que identificam se dois objetos ocupam o mesmo espaço; e as rotinas C***inemáticas e Dinâmicas***, usadas para testar a operação de mecanismos.

Finalmente, a quarta área em que o CAD é útil no processo de projeto é a de ***apresentação e documentação***. Os sistemas CAD podem ser utilizados para preparar desenhos de engenharia de maneira rápida e acurada, incrementando sensivelmente a produtividade do projetista quando comparado à preparação manual.

Com isso, destacam GROOVER [21] e LIN [32], atualmente os sistemas CAD contribuem principalmente para as atividades de projeto devido ao fato de: ***Aumentar a Produtividade do Projetista,*** com a ajuda ao projetista para conceituar o produto e os seus componentes; ***Aumentar a Qualidade do Produto,*** com o uso de um sistema CAD com capacidades de hardware e software apropriadas, permitindo ao projetista realizar uma análise de engenharia mais completa, e considerar uma maior variedade e número de alternativas de projeto, melhorando assim a qualidade e precisão dos resultados de projeto; ***Melhorar a Documentação do Projeto,*** uma vez que a saída gráfica de um sistema CAD resulta numa melhor documentação do projeto do produto quando comparada às técnicas manuais, já que uma padronização pode ser aplicada aos desenhos, com menor quantidade de erros, maior legibilidade e com economia de espaço podendo os arquivos serem mais facilmente acessados, reproduzidos e revisados; e finalmente, ***Criar um Banco de Dados para Manufatura,*** uma vez que no processo de criação da documentação do projeto do produto (especificações geométricas, dimensões dos componentes, especificações de materiais, custo dos materiais, etc...), grande parte do banco de dados requerido para manufaturar o produto também já é criada.

3.3.2 As Funções de Planejamento e Controle da Manufatura

GROOVER [19] define os sistemas CAM como o uso efetivo da tecnologia de computador no planejamento e no controle das funções de manufatura. Como visto na figura 3.6, suas aplicações podem ser divididas em duas categorias: o planejamento da manufatura e o controle da manufatura. Nesta fase as informações e a documentação que constituem o projeto do produto fluem para a função de planejamento da manufatura.

No ***Planejamento da Manufatura*** o computador é usado indiretamente para suportar as funções de produção, mas não há contato direto entre o computador e o processo de manufatura.

O computador é usado "off-line", provendo as informações para o efetivo planejamento e gerência das atividades de produção. Hoje em dia, diferentes aplicações importantes de sistemas computacionais para o auxílio à manufatura podem ser listadas, dentre estas: as de ***estimativa de custos***, através do processamento computacional dos principais passos necessários para preparar uma estimativa; as de ***planejamento de processo***, onde os sistemas CAPP ("Computer Aided Process Planning") possibilitam a preparação de planilhas que listam as seqüências de operações e os centros de fabricação necessários para produzir o produto e seus componentes; e as de ***programação de máquinas ferramentas NC*** e ***robôs industrias***, através de sistemas que em muitos casos representam um método mais eficiente para a geração das instruções de controle destes equipamentos do que com os métodos manuais.

A segunda categoria das aplicações de sistemas CAM está concentrada no desenvolvimento de sistemas de computador para a implementação da função de ***controle da manufatura***, ou seja, o gerenciamento e o controle das atividades físicas da fábrica. O ***controle de processo***, o ***controle de qualidade***, o ***controle de chão de fábrica*** e o ***monitoramento de processo*** estão inclusos no escopo desta função.

3.3.3 A Integração de Sistemas CAD e CAM

No fluxo do ciclo de informações de projeto e manufatura, os arquivos precisam ser transferidos de uma organização, usando um tipo de sistema, para outra, utilizando outro tipo de sistema. Entretanto, os modelos criados em um sistema CAD/CAM não podem ser usados diretamente em um outro sistema CAD/CAM. É o que se chama de problema de compatibilidade de bancos de dados, e caracteriza um dos maiores problemas na integração de sistemas CAD/CAM.

LIN [32] destaca dois principais fatores para esta incompatibilidade. Primeiro, a maioria dos modelos em bancos de dados dos sistemas CAD/CAM comerciais são escritos em seu próprio formato compacto, de modo a minimizar o espaço em disco e o tempo de acesso aos dados. Em segundo, esses modelos internos aos bancos de dados são considerados propriedade da empresa e, conseqüentemente, não são públicos.

Atualmente, esta incompatibilidade existe entre diferentes versões de um mesmo programa aplicativo obrigando os vendedores a alterar o banco de dados periodicamente, quando novas características e entidades são adicionadas ao programa. Esta incompatibilidade dos formatos dos arquivos tornou-se a maior restrição à transferência de dados para a integração entre os diferentes pacotes de programas.

De modo a integrar os diferentes sistemas CAD/CAM, a necessidade de padronização na troca de dados tornou-se um sentimento crescente entre os responsáveis pelo desenvolvimento de sistemas CAD/CAM. Agências governamentais, acadêmicos de pesquisa, industriais e vendedores de software vêem estudando este problema.

Hoje em dia dois métodos são utilizados para resolver os problemas de interface de dados: a comunicação direta e a comunicação indireta [32].

3.3.3.1. A Comunicação Direta

A comunicação direta é comumente usada para a troca de dados entre pacotes de programas aplicativos específicos. Este método requer um tradutor customizado para a troca de dados entre os sistemas. A figura 3.9 ilustra um diagrama da comunicação direta entre sistemas específicos de CAD/CAM.

A comunicação direta tem como vantagem o fato de ser uma maneira eficiente e efetiva de troca de dados, além de minimizar tempo e espaço em disco. Como desvantagem, cita-se o

fato de necessitar de atualizações no tradutor a cada versão nova dos sistemas, podendo ter assim um custo alto de manutenção.

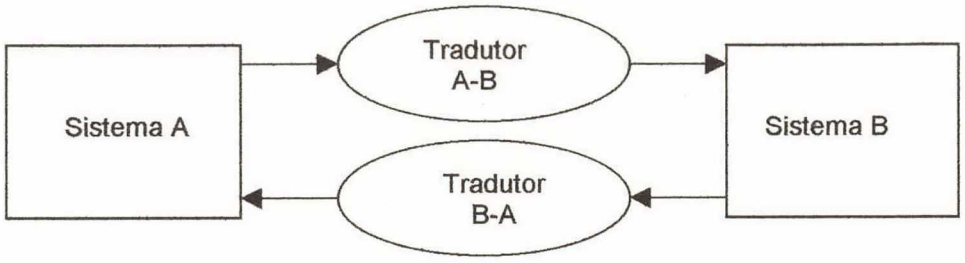

FIGURA 3.9: Comunicação Direta.

3.3.3.2. A COMUNICAÇÃO INDIRETA

A comunicação indireta utiliza processadores padronizados, implementados para a troca de dados em formato neutro de arquivo. A figura 3.10 ilustra um diagrama esquemático para implementação da comunicação indireta.

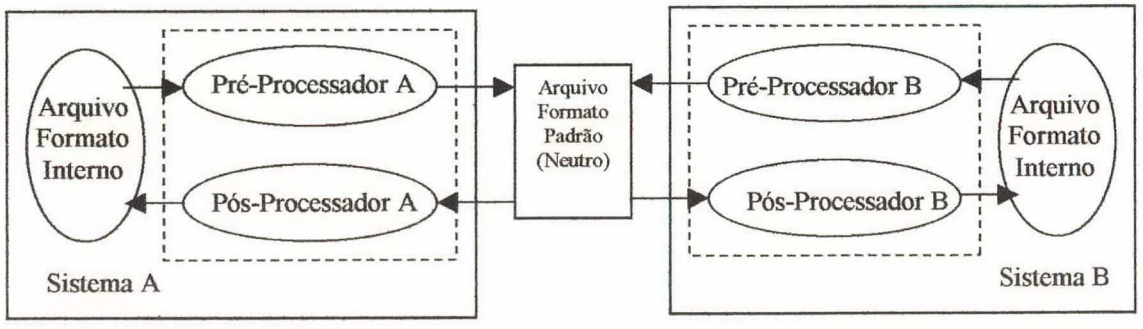

FIGURA 3.10: Comunicação Indireta.

O primeiro tradutor, que processa o formato interno e exporta os dados no formato padronizado, é chamado de pré-processador. O segundo tradutor, que importa o arquivo em formato neutro e converte o formato padrão num formato interno ao sistema, é chamado de pós-processador.

Lin [36] sugere como vantagens do uso da comunicação indireta a facilidade para os desenvolvedores de software em manusear as utilidades do tradutor com um formato de arquivo

padronizado, bem como, a diminuição do tempo de desenvolvimento e o acesso a um canal de comunicação aberto e fidedigno entre os sistemas CAD/CAM.

Por outro lado, como desvantagens o autor [36] sugere que em geral apenas tipos básicos de entidades geométricas estão inclusos, o que acarreta problemas durante a troca de dados de um sistema para outro. E ainda, que os desenvolvedores de software podem ficar com um potencial limitado para a elaboração de um pacote mais complexo se o padrão não suporta estes desenvolvimentos. Conseqüentemente, o padrão impede os desenvolvimentos de software.

Apesar destas desvantagens, a troca indireta de dados baseada em formatos neutros tem ganhado popularidade nos últimos anos. Muitos vendedores de software CAD/CAM vêem desenvolvendo e incluindo utilidades de troca de dados nos seus pacotes. Com intuito de fazer uma breve explanação sobre o estado atual do desenvolvimento de padrões de troca de dados, os parágrafos seguintes ilustram as características de alguns destes padrões populares através da comunicação indireta.

A) DXF

O padrão DXF ("Data Exchange Format") foi desenvolvido pela "AutoDESK" e usado no software "AutoCAD". Desde a sua inserção, o DXF vem sendo um padrão popular na indústria para a troca de dados de arquivos de desenhos da maioria dos sistemas CAD/CAM baseados em microcomputadores. Os principais vendedores de software CAD/CAM adotaram o DXF no suporte de troca de dados.

Os arquivos DXF são arquivos texto com padrão ASCII que contêm todas as informações do banco de dados para o desenho. Uma troca de dados em DXF é um processo de dois passos: o sistema de origem usa o pré-processador DXF-out para traduzir as entidades do banco de dados para o formato DXF. O sistema de recebimento utiliza o pós-processador DXF-in para trazer o arquivo formatado em DXF para o formato nativo do seu banco de dados.

B) IGES

O padrão IGES ("Initial Graphics Exchange Specification") iniciou-se por volta de 1970 pela Força Aérea Americana e foi suportado pelo departamento de defesa, usuários da indústria e vendedores de sistemas CAD/CAM deste país. Este padrão foi adotado pela ANSI ("American National Standards Institute") e tornou-se um padrão nacional em 1981. Desde então, o padrão IGES teve diversas revisões para ampliar sua capacidade e eficiência.

Assim como o padrão DXF, o padrão IGES ganhou o suporte de vendedores e usuários de sistemas CAD/CAM. Similar ao padrão DXF, os arquivos IGES são arquivos de texto padrão ASCII, mas possuem diferenças significantes em termos da estrutura de dados e da definição de entidades.

C) PDES

O padrão PDES ("Product Data Exchange Specification") iniciou-se em 1985 por um constituído de agências governamentais, pesquisadores e companhias privadas. A tarefa constituía do desenvolvimento de um padrão nacional com ênfase na representação e troca de informações completas de um produto manufaturado, e não apenas no seu desenho. Esta informação inclui a geometria do produto, as propriedades dos materiais, as tolerâncias, os acabamentos de superfícies e outras especificações de fabricação. O objetivo final do padrão PDES é prover a informação completa do produto de modo que este possa ser usado por programas aplicativos como os de projeto de produto, os de modelagem geométrica, de análise de elementos finitos, de planejamento de processo, de controle de inventário, e de geração de programas de controle numérico, todos estes com um mínimo de reinterpretação humana. A implementação do padrão PDES efetivamente auxilia as indústrias de manufatura na troca de informações completas dos produtos.

D) STEP

Outros países industrializados reconheceram a necessidade da padronização na troca de dados de desenhos, e foram feitos esforços para estabelecer seus próprios padrões. Entre estes padrões têm-se o SET, padrão para troca de dados da França, e oVDA/FS, um padrão alemão para a troca de dados de superfícies 3D.

Os esforços para definir uma padronização na troca de dados devem ser unificados internacionalmente. Esta meta é liderada pela ISO ("International Standard Organization"). O primeiro padrão internacional para a troca de dados, STEP ("Standard for the Transfer and Exchange of Product Model Data"), é um projeto em andamento e iniciado pela ISO em 1984. Este padrão permite a troca de dados de um modelo computacional do produto, com toda a sua informação num formato neutro pré-determinado. O desenvolvimento deste padrão mundial é influenciado por muitos padrões existentes de diferentes países industrializados, e tem com alvo produzir um padrão global num futuro próximo.

Naturalmente, o progresso que acompanharemos durante os próximos anos não ficará restrito aos exemplos aqui mencionados: tecnologia WEB, Internet, Intranet, VRML e HTML, cada vez mais se consolidarão no vocabulário dos sistemas CAD/CAM e também contribuirão para a conformação dos sistemas atuais aos novos moldes tecnológicos.

3.4 Ferramentas Computacionais para o Planejamento, Simulação e Programação de Tarefas dos Robôs Industriais

A pesquisa realizada em recentes anos no uso do projeto (CAD) e da fabricação (CAM) auxiliados por computador para aplicações de robótica, e as resultantes aplicações industriais desenvolvidas neste campo decorrem de diferentes imperativos e exigências. As aplicações dos robôs industriais envolvem, freqüentemente, problemas relacionados ao movimento de uma

estrutura articulada em um espaço de trabalho com uma visão para realizar uma determinada tarefa no qual há obstáculos presentes. A variedade de tipos de robôs industriais e a extensa e indeterminada diversidade de tarefas, como comentado no capítulo 2, formam um contexto complexo para a aplicação destes sistemas. Com isso, como será visto, muitas ferramentas gráficas de auxílio ao planejamento e programação de robôs industriais estão em uso ou em desenvolvimento, buscando o planejamento de aplicações tanto generalizadas como para casos específicos das atividades de fabricação.

Já são amplamente usados os sistemas CAD/CAM em processos industriais, constituindo uma importante ferramenta para melhorar a produtividade. Deste modo, seu uso deve ser considerado na estrutura mais geral de projeto e de técnicas de produção para permitir que sua importância em robótica seja avaliada. As diversas tarefas que são ou serão automatizadas através do uso dos robôs industriais, como a pintura, a soldagem, o transporte de materiais e até mesmo as atividades de montagem, podem se beneficiar da utilização dos sistemas CAD/CAM para a sua programação fora de linha, através da integração com as demais funções de projeto e fabricação.

Esta tendência pode ser notada na área de robótica através do desenvolvimento de sistemas gráficos interativos que auxiliam o planejamento, a análise e a programação de tarefas de robôs industriais, sendo este o objeto do estudo principal deste capítulo. O apêndice 4 apresenta uma breve revisão das fases identificáveis para implementação de uma aplicação de robótica, em especial a fase de planejamento, e que podem ser auxiliadas por estes sistemas computacionais. Assim como controle numérico, brevemente revisado no Apêndice 3, a pesquisa atual e as aplicações industriais estão tentando usar os sistemas CAD/CAM no processamento de informações para qualificar os dados gerados para programação das tarefas dos robôs industriais, minimizando o envolvimento do próprio.

Tais ferramentas de programação permitem realizar as fases de planejamento, desenvolvimento, teste e programação de tarefas de robôs industriais sem a utilização dos mesmos. A idéia consiste em verdadeiramente auxiliar computacionalmente as três primeiras fases de implementação (Apêndice 4), simplificando a instalação e o início da produção de uma nova aplicação de robótica.

Na robótica, a simulação é importante porque pode ajudar a resolver os problemas de otimização, implantação e de colisão. A simulação também pode ajudar a resolver problemas do tipo teórico, como determinar trajetórias ótimas. Também pode reduzir os custos, o estudo piloto e o tempo de instalação. Todos os usuários de robô concordam que a simulação de tarefas de robô é uma ajuda inestimável à programação. Com a variedade de tipos de robô (dispositivos de manipulação simples, robôs de pintura com sete eixos cartesianos etc.), o usuário de robôs apreciará a ajuda de uma ferramenta de simulação que lhe permitirá evitar trabalhar cegamente em um robô sobre o qual ele sabe muito pouco.

BIEN [06] destaca que a atual sofisticada tecnologia de software de simulação habilita a modelagem geométrica precisa e acurada de robôs, efetuadores finais e dispositivos de proteção dentro da célula robótica de fabricação. Além disso, podem ser incorporados modelos humanos nas células para estudos ergonômicos e análise da tarefa. Empregando atributos atuais de robôs, como o planejamento do movimento, cinemática, dinâmica e lógica binária, é possível simular virtualmente qualquer aplicação de robótica.

A simulação permite aos engenheiros avaliar a célula por inteiro sem qualquer obstrução visual para selecionar os robôs, os dispositivos de proteção e os equipamentos subordinados apropriados e, então determinar a colocação ótima de cada componente. Igualmente, podem ser restaurados modelos precisos de robôs industriais de uma biblioteca inclusiva para avaliar certas características, como o máximo volume de trabalho de cada robô. Também, podem ser colocados robôs em localizações ótimas baseando-se em acessibilidade, tolerâncias, colisão e limitações de

articulação. Estes fatores podem ser calculados automaticamente ou podem ser definidos por engenheiros para assegurar o trabalho próspero para o robô, antes da instalação atual.

BIEN [06] defende que reunindo as várias formas de informação em um único ponto de referência, torna-se viável que níveis múltiplos de decisão, como os operadores de máquina, engenheiros de manutenção, engenheiros de fatores humanos, engenheiros de projeto e engenheiros de processo participem coletivamente no projeto inteiro da célula de manufatura. E ainda, ressalta o autor, devido a sua natureza visual e interativa, a simulação é uma ferramenta para eliminação das "ilhas de dados" de CAD e quebra das "barreiras de comunicação", tornando-se uma ferramenta de escolha para condução da análise de projeto, da avaliação de riscos, da programação fora de linha e do treinamento.

Com isso, a simulação oferece uma vantagem significante sobre os métodos tradicionais com protótipos, no que diz respeito à sua habilidade para criar, armazenar e restaurar bibliotecas completas de robôs, dispositivos periféricos, equipamentos subordinados, modelos humanos, dispositivos de proteção, e planos inteiros de células de fabricação. Esta funcionalidade permite aos engenheiros referenciar células de fabricação previamente modeladas para encurtar e dar forma ao processo de projeto mais eficientemente. Além disso, enquanto são projetadas as células de fabricação, os engenheiros podem explorar planos alternativos para alcançar uma solução ótima.

E ainda, segundo BIEN [06], em análise de projeto, a simulação é uma ferramenta versátil para estudar a natureza complexa de cada célula de fabricação. A flexibilidade inerente de robôs e a sua habilidade para utilizar a célula por inteiro criam variáveis de segurança adicionais que não existiriam em centros convencionais de fabricação. De fato, depois da instalação da célula, vários problemas e violações dos padrões de segurança que tinham sido negligenciados são descobertos, sendo deixadas aos engenheiros a árdua e cara tarefa de modificar a célula implementada. Hoje, utilizando a simulação anteriormente às instalações, os

engenheiros podem aderir à prática de bom projeto obedecendo aos padrões de segurança da ANSI/Robotic Industries Association.

Uma das vantagens de fácil observação desta aproximação é a segurança obtida pelo comprador do robô, pois é possível saber antecipadamente que este pode fazer o seu trabalho, evitando-se surpresas de última hora, como por exemplo, um espaço de trabalho muito pequeno. Certamente o problema do planejamento e da escolha da melhor instalação de robô pelo usuário para o trabalho é freqüentemente muito complexo para ser resolvido só pelo comprador de um robô industrial. Neste momento aparece a primeira vantagem principal de usar o CAD/CAM.

PARENT [36], defende que o CAD/CAM também permite antecipar o desenvolvimento, reduzir o tempo e os custos de desenvolvimento, minimizar os riscos durante a programação, e ainda, evitar a dependência da produção de um protótipo.

Algumas atividades devem ser coordenadas para a realização do planejamento e posterior programação. Em geral, o processo consiste na modelagem geométrica, cinemática e dinâmica do robô industrial, dos objetos envolvidos na tarefa e do seu ambiente. O projeto conceitual da célula de manufatura envolve o projeto e a seleção dos elementos para realização da tarefa, e na descrição desta. Os elementos são combinados para formar o plano que especifica em detalhes as atividades e interações dos elementos da célula de manufatura. A partir deste planejamento, e utilizando de métodos de programação, tem-se o desenvolvimento e validação dos programas de tarefas de acordo com o projeto e a funcionalidade da célula de manufatura. Estes programas são então convertidos em linguagens específicas de robôs e carregados no controlador do robô para realização das operações de fabricação.

A seguir temos uma breve descrição das atividades que podem ser geralmente identificadas para realização do planejamento, simulação e programação de tarefas dos robôs industriais.

3.4.1 A MODELAGEM DOS ELEMENTOS ENVOLVIDOS NO PLANEJAMENTO

É sabido, como identifica SOLAND [49], que uma condição prévia para simular os movimentos de um robô é o software para o projeto e a representação do robô e seu ambiente em um terminal gráfico. Ao envolver a simulação dos movimentos de robô ou a obtenção da informação gráfica para uso na programação formal, é vital obter um modelo geométrico que possa ser usado facilmente pelo robô. Esta definição relacionará os objetos envolvidos nas tarefas a serem executadas pelo robô, como também o ambiente no qual se dará a produção.

Segundo PARENT [36], nesta fase preliminar de modelagem, tem-se uma definição dos elementos envolvidos: robô, objetos e ambiente. Sendo assim, um sistema de programação fora de linha deve poder modelar as propriedades de mecanismos articulados.

YONG et al. [56] destacam que há vários níveis nos quais isto pode ser tentado. O primeiro nível é o desenvolvimento de um sistema para um ***robô específico***, o que simplifica muito a implementação porém limita o âmbito de aplicação do sistema. Uma segunda aproximação é generalizar a uma ***classe de estruturas*** limitadas. Por exemplo, a maioria dos robôs comerciais consiste em um arranjo hierárquico de articulações controladas de forma independente e que permitem movimentos de rotação ou translação. Até mesmo a este nível, não há nenhuma solução geral que cubra todas as possibilidades, sendo necessário classificar as estruturas em grupos para os quais possam ser desenvolvidos algoritmos de controle apropriados. O terceiro nível é tentar dirigir ***estruturas complexas*** de manipulador, com articulações interconectadas, matematicamente dependentes, que não são compreendidas em qualquer senso geral, embora possam ser analisados exemplos particulares.

Quanto à sua representação gráfica para visualização, PARENT *et al* [36] enfatizam que esta também pode ser realizada em diferentes níveis: "ao nível de projeto elementar, um robô pode ser representado como um sistema articulado, formado por várias linhas unidas por eixos de

translação ou de rotação. Esta representação contém os elementos essenciais de definição do modelo: distâncias entre eixos e distâncias ao longo de ou sobre as quais os movimentos são realizados. Um método mais preciso de modelagem geométrica consiste em prender volumes ao esqueleto definido no modelo, para obter uma idéia do real tamanho do robô. Usando volumes geométricos simples (cones, cilindros, prismas etc.) interligados, é possível desenvolver um modelo de robô que considera o envelope de trabalho", como ilustrado na Figura 3.11.

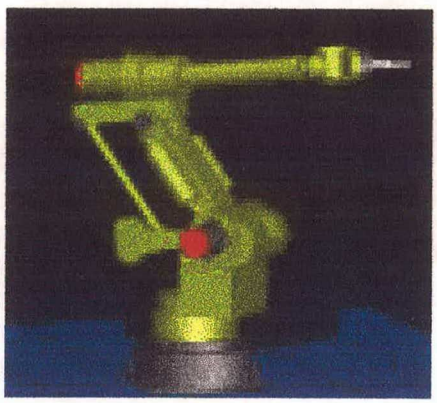

FIGURA 3.11: Visualização através da estruturação de volumes para composição do modelo geométrico do robô industrial.

A partir das diversas referências [24, 25, 27, 29, 31, 40], é possível definir que o software usado para definição do modelo do robô pode prover as informações para definição da sua ***Geometria***, *Cinemática e Dinâmica*, definição das ***Restrições de Articulações*** (Posições, Velocidades e Acelerações) e definição do ***Espaço de Trabalho***.

Com isso, nesta fase inicial é possível realizar a comparação do volume do espaço de trabalho para robôs diferentes (por exemplo, sobrepondo-os), a designação preliminar do modelo de robô e efetuador mais bem adaptado para o propósito intencional e a comparação preliminar entre os métodos de produção para dois robôs do mesmo tipo básico.

De fato, como ressalta PARENT [36], a definição de um objeto em um sistema CAD é extremamente simples, quando comparada com os problemas que podem surgir quando a modelagem é tentada em sistemas padrão de computador que usam métodos sintáticos (descrição

textual e analítica). A facilidade oferecida pelos sistemas CAD para o mesmo propósito pode apresentar uma tremenda vantagem. Se os objetos que são manipulados ou estão envolvidos na tarefa do robô, forem projetados já usando um sistema CAD, o problema de modelar não surge.

Posteriormente, estes objetos, embora já modelados, devem estar preparados para permitir formar o programa desejado. O problema mais importante nesta fase de formação do modelo é a posição relativa do conjunto de eixos de coordenadas para a definição de ambos, o robô e os objetos envolvidos. A figura 3.12, a seguir, visa ilustrar a possibilidade de modelagem numa interface gráfica interativa de um sistema em que uma célula de trabalho complexa pode ser descrita e a interação entre os elementos pode ser estabelecida.

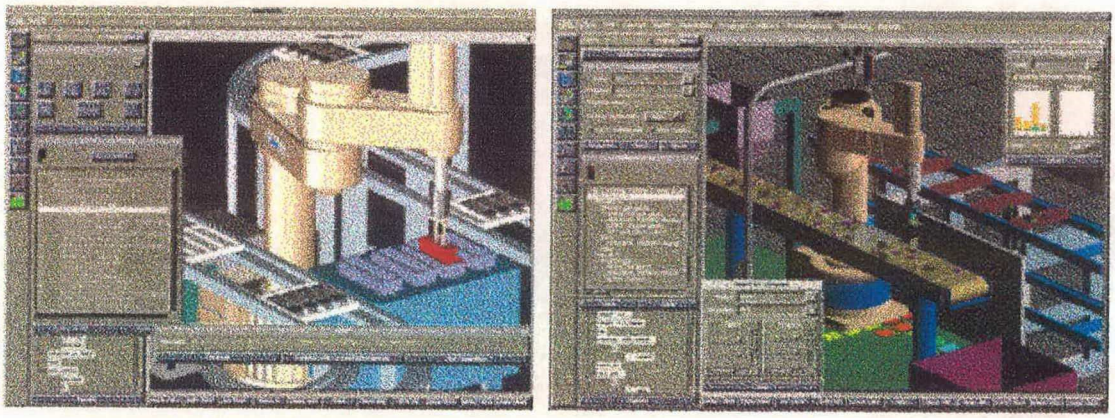

FIGURA 3.12: Modeladores de Estações de Trabalho. (http\\www.Silma.com)

3.4.2 O PROJETO FUNCIONAL DA ESTAÇÃO

Não apenas devem ser modelados os elementos constituintes da célula, mas também os tipos e a possibilidade de interação entre estes de modo a formar um plano funcional de operação.

Segundo KATAJAMAKI [29], dentre os objetivos desta fase de projeto funcional, tem-se: a descoberta do melhor arranjo de plano para a célula de produção, a seleção de máquinas e

equipamento mais satisfatórios para o propósito intencional, o projeto de especificação da tarefa e percursos e a garantia de que todas as tarefas estão contidas no espaço de trabalho do robô.

KATAJAMAKI [29] define que o projeto da planta começa pela localização dos modelos 3D das construções fixas e equipamentos em um sistema CAD. Como vimos, atualmente, vários formatos externos de dados de CAD pode ser virtualmente importado em um sistema de software para simulação. Isto permite aos engenheiros capturar os dados de sistemas CAD incompatíveis. Depois disto, a localização preliminar para o robô é buscada, de modo a obter o arranjo espacial inicial para posterior funcionamento.

Segundo PARENT [36], em um estudo preliminar de viabilidade, o sistema CAD permite que os objetos, ou como é o caso mais freqüente, o robô em relação aos objetos, seja posicionado com uma função de acessibilidade destes objetos.

KATAJAMAKI [29] ressalta que o espaço de trabalho do robô não é, porém, um local de conceito único no espaço, uma vez que este muda conforme a ferramenta a ser usada, devendo ser averiguado se o robô pode atingir o ponto na orientação exigida. Segundo ele, o método de simulação baseado em CAD é mais simples e preciso para se realizar esta tarefa, e, além disso, o tempo de performance das modificações necessárias é reduzido.

Com isso, como destaca PARENT [36], a vantagem de usar o CAD decorre da possibilidade de definir com precisão os elementos geométricos, e uma vez satisfeitas as restrições de acessibilidade, a real instalação destes elementos, pode ser determinada. Sem a ajuda do CAD o trabalho pode ser longo e difícil para o projetista.

Como um resumo dos benefícios do uso de sistemas CAD para o projeto da célula de manufatura e de sua funcionalidade, pode ser mencionado: a utilização de um sistema computacional para teste ou avaliação preliminar, o robô pode ser selecionado de acordo com a tarefa, há a possibilidade de experimentação (dentro das limitações de horário e orçamento) com várias soluções alternativas, e ainda, pode-se assegurar que o planejamento do controle de

ferramentas e equipamentos periféricos procede ajustadamente com a operação da célula e do robô.

3.4.3 O MÉTODO DE PROGRAMAÇÃO

O projeto do processo de produção continua, então, para a definição e programação das tarefas. Segundo YONG [56], os modeladores geométricos e de robô provêem a capacidade de controlar estruturas de robô dentro de um sistema de coordenada de modelagem "mundo". Um método de programação é exigido para habilitar seqüências de controle, quer dizer, seqüências de movimento de robô, a serem definidas e armazenadas de uma maneira lógica. O método deve permitir comandos de robô, funções de robô, e uma lógica de ciclo a ser incorporada dentro destas seqüências para habilitar as especificações de programas completos para os robôs.

Segundo o autor [29], a realização desta fase inclui a definição dos comandos de posicionamento, comandos de movimento, métodos de inspeção dos limites de juntas para conferir os ângulos críticos de junta, e eliminar os sobrepassos, e ainda, realizar o cálculo dos tempos de ciclo para percursos individuais considerando as transformações da ferramenta.

A obtenção dos objetivos traçados aqui requer, além de pensamento intuitivo e planejamento, muita experiência e testes. No método tradicional as fases de trabalho são freqüentemente executadas montando-se uma célula de teste na qual são construções aproximadas para testar as várias alternativas da operação de robô e são comumente dependentes do robô pré-designado. No sistema CAD/CAM não há tal limitação. Se o simulador de robô foi desenvolvido com métodos de programação e ensinamento independente do robô, e o sistema suporta várias alternativas de robôs, a célula de produção pode ser testada com vários tipos diferentes de robô. Por outro lado, um dos fatores claramente desvantajosos é a dificuldade de formular um mundo tridimensional numa exibição bi-dimensional do monitor.

KATAJAMAKI [29] ressalta que o detalhamento do planejamento e da programação de funções de robô começa depois que o projeto inicial da planta foi completado, o efetuador final e os equipamentos periféricos foram selecionados e as operações principais da célula são conhecidas. Segundo este, o objetivo desta fase de trabalho consiste na finalização das tarefas e caminhos de robô, assegurando-se que todas as localizações podem ser alcançadas na devida orientação, os caminhos individuais não incluem mudanças de configuração, as localizações intermediárias das interpolações não incluem no movimento localizações inatingíveis, e o robô não interfere com equipamentos periféricos ou peças de trabalho. Em seguida, devem ser criados programas aplicativos prontos para rodar, que considerem as trocas das ferramentas e da garra, estando presentes as operações condicionais e ramificações de programa, e realizando o processamento de entradas e saídas binárias, o processamento de entradas e saídas analógicas, e situações de emergência e anormais. E com isso, garantir da correta operação de robô e que o desempenho do robô e os programas estão dentro dos limites de exigência de produção.

KATAJAMAKI [29], defende as exigências para simulação das operações da célula de manufatura não são extensas. Deve-se assegurar, com a simulação, que os movimentos de robô em cada tarefa individual seja prosperamente carregado e que os tempos de ciclo correspondentes possam ser conferidos. Por isto os comandos do simulador podem cobrir somente comandos de movimento e não, por exemplo, comandos condicionais ou comandos relacionados ao controle.

Segundo YONG [56], estas exigências posteriores causam complicações quando o sistema é aplicado às diferentes áreas de aplicação. Por exemplo, as exigências funcionais e de técnicas de robô para a soldagem a arco são significativamente diferentes das envolvidas em pintura com jato. A modularização em áreas de aplicação do método de programação deveria aliviar estas complicações e produzir um sistema global mais eficiente.

De fato, o único jeito prático para assegurar a correta operação do robô é através de um teste, com um sistema de simulação ou com o próprio robô. Entretanto, vale ressaltar, que para que um sistema de simulação possa ser substituído por um teste com o robô, este deve poder testar todas as suas funções. Seria adicionalmente vantajoso se tais funções, como por exemplo, situações de emergência, pudessem ser testadas durante a simulação, pois nem sempre podem ser testadas no mundo real por causa de riscos econômicos.

Portanto, os sistemas de CAD/CAM, embora não sejam projetados especificamente para os usuários de robôs, podem oferecer soluções para os problemas da programação fora de linha, com várias vantagens. Um banco de dados geométrico, que define as configurações de robô, pode ser gerado rapidamente e precisamente, e ser facilmente modificado. Podem ser determinadas configurações atingíveis, e assim, o programa e a sua implantação serem modificados ou corrigidos. E ainda, podem ser resolvidos mais prontamente os problemas de colisão.

De fato, uma variedade de métodos de programação podem ser propostos para os mais diferentes tipos de atividades dos robôs industriais. Por exemplo, diagramas de ligação podem descrever uma atividade de montagem [08], ou ainda o próprio ambiente 3D pode permitir que o usuário defina as movimentações como numa programação "on-line" [47], só que num ambiente virtual. Mas, independente da complexidade do esquema para o planejamento e simulação da célula robótica, de maneira geral, estes sistemas se caracterizam como geradores de trajetórias, que são curvas no tempo de posição e orientação do efetuador do manipulador no espaço cartesiano ou de posições das juntas do manipulador no espaço de juntas. Na seção seguinte será abordado brevemente este problema.

3.4.4 A GERAÇÃO DE TRAJETÓRIAS

A geração de trajetória de um manipulador consiste no estudo de métodos computacionais para encontrar uma trajetória que represente o caminho desejado a ser descrito pelo efetuador-final para execução de uma tarefa

A trajetória pode ser definida como a história temporal da posição, velocidade e aceleração em cada um dos graus de liberdade do robô. Assim, deseja-se encontrar uma seqüência temporal de pontos que descrevam o movimento do manipulador quanto à sua posição, velocidade e aceleração, definindo totalmente a sua configuração a cada instante de tempo. Esta seqüência de pontos é posteriormente enviada ao sistema de controle do robô. Este sistema é responsável pelo acionamento dos atuadores que movimentam o manipulador.

Em geral, os procedimentos de geração de trajetórias podem ser classificados em métodos de planejamento no espaço cartesiano e no espaço de juntas do manipulador [11, 45].

Diversos métodos podem ser propostos para o planejamento de trajetórias no espaço cartesiano e no espaço de juntas do manipulador, como é revisado detalhadamente nos trabalhos de ROSA [45] e CRAIG[11]. No esquema de espaço de juntas, a forma espacial do caminho percorrido pelo efetuador-final é complicada posto que depende da cinemática do manipulador que está sendo usado. Usando o método de geração de trajetórias no espaço cartesiano é possível especificar a forma do caminho entre os pontos escolhidos, ou seja, se o caminho é uma linha reta, um círculo, uma senóide, entre outros. O problema é que caminhos definidos em espaço cartesiano são mais custosos de serem executados em tempo-real devido à necessidade de se calcular a transformação de valores cartesianos em valores no espaço de junta durante a execução.

Uma possível escolha para representar o caminho é a interpolação linear dos pontos que o definem. O problema é que este tipo de interpolação faz a velocidade ser descontínua no início e fim do movimento. Para criar um movimento com posição e velocidade contínuas, é possível começar com uma função linear, e no final da reta adicionar uma região curva parabólica. Durante a parte curva da trajetória uma aceleração constante é usada para mudar a velocidade. Esta é uma boa escolha, pela simplicidade. Também pelo fato de que outras trajetórias mais complexas poderem ser formadas pela justaposição de segmentos de reta. Outra razão é o fato de as trajetórias retilíneas são mais simples para visualização, além de apresentarem uma ampla aplicação em processos industriais, tais como operações de montagem, realização de cordão de solda entre duas chapas, etc.

No que diz respeito à exigência de desempenho do manipulador, é possível classificar os métodos de geração de trajetórias em duas categorias [45]:

"As ***trajetórias de transporte ou movimentação***, o manipulador realiza o movimento entre dois pontos definidos no espaço, sendo livre a trajetória seguida entre estes dois pontos. Aplicações típicas são a pega de peças em um ponto e descarga em outro, montagens de produtos e operações de transferência. Uma variante desta trajetória é a trajetória de movimentação com restrições, onde são adicionados pontos intermediários, pelos quais a trajetória deve passar, de modo a permitir um melhor controle sobre esta, como se requer no desvio de obstáculos.

A segunda classificação é relativa às ***trajetórias de precisão***, onde o importante passa a ser a trajetória propriamente dita e não os pontos de início e fim. É necessário que esta respeite o máximo possível a trajetória idealizada, de modo a cumprir adequadamente a tarefa. Aplicações típicas neste caso são as de pintura, de desbaste, de solda elétrica a arco ou a ponto, onde existe uma trajetória ideal que deva ser seguida tanto quanto possível".

CRAIG [11] ressalta que: "com relação ao usuário, deve-se levar em consideração o problema da interface humana no processo de planejamento da trajetória. O objetivo é fazer uma descrição do movimento do manipulador, que seja o mais simples possível para a interpretação de um usuário humano, de modo que não lhe seja exigido escrever funções complicadas para especificar a tarefa desejada. Deve-se, portanto, permitir a especificação de trajetórias com simples descrições e deixar o sistema calcular os detalhes. Por exemplo, o usuário pode especificar a posição e orientação desejadas e deixar o sistema decidir a forma exata do caminho, a duração, o perfil de velocidade e aceleração, entre outros detalhes".

CRAIG [11] também ressalta que é preciso levar em consideração que a trajetória é calculada em computadores digitais que possuem uma certa taxa típica de amostragem. Esta taxa deve ser respeitada para que o movimento seja suave. Desta forma, a cada instante de tempo os novos valores de posição, velocidade e aceleração devem estar disponíveis para serem enviados ao sistema de controle do robô. Este é mais um fator que contribui para que se trabalhe com trajetórias definidas em espaço de juntas, já que, geralmente, tem-se uma elevada freqüência de amostragem e a cinemática inversa exige muito esforço - e tempo - computacional.

ROSA [45] identifica vários critérios que podem ser utilizados para comparar e avaliar as trajetórias e os sistemas de planejamento: "em primeiro lugar, as trajetórias devem ser eficientes, tanto para o cálculo como para a execução. Em segundo lugar, as trajetórias devem ser previsíveis e precisas, não devendo degenerar de forma inaceitável próximo a singularidades, o que pode ocorrer para um planejamento no espaço cartesiano. Em terceiro lugar, a posição, velocidade e aceleração devem ser funções suaves no tempo".

Usualmente é desejado que o movimento do manipulador ocorra de maneira suave. Para isto, pode ser definida uma função suave de modo que ela e sua primeira derivada sejam contínuas. Também pode ser exigido, dependendo da aplicação, que sua segunda derivada seja

contínua para evitar que ocorra um aumento no desgaste dos órgãos mecânicos do robô, e também para evitar excitações de altas freqüências em seu mecanismo articulado. Finalmente, deve-se ter a preocupação de avaliar a trajetória proposta, verificando se esta leva o braço do manipulador para fora da área de trabalho, ou, então, se os limites de velocidade ou de aceleração são respeitados.

O apêndice 1 acrescenta alguns aspectos relativos aos métodos de geração de curvas cartesianas 3D, bem como do método utilizado na metodologia proposta.

3.4.5 A INTERFACE COM OS ROBÔS INDUSTRIAIS

Um sistema de programação fora de linha define e armazena a descrição de um programa de robô em um formato interno específico. Em geral, o formato é significativamente diferente do formato empregado por um controlador de robô para o programa equivalente. Conseqüentemente, é necessário ter uma forma de interface para converter a descrição do programa de um sistema fora de linha para um formato de controlador. Um dos problemas principais é a existência de um grande número de controladores diferentes junto com uma variedade de sistemas de programação, cada um empregando formatos diversos de descrição de programa.

Para evitar uma multiplicidade de interfaces entre sistemas e controladores específicos, precisam ser definidos e adotados padrões. YONG [56] sugere que esta padronização poderia ser empregada em uma ou mais das seguintes áreas:

1. O *Sistema de Programação*. A adoção de um sistema padrão para a programação fora de linha reduziria consideravelmente os esforços de interface.

2. O *Sistema de Controle*. Um sistema controlador de robô unificado teria efeitos benéficos semelhantes à unificação dos sistemas de programação. Considerações e práticas comerciais fazem desta aproximação uma ocorrência improvável.

3. O *Formato de programa*. A definição de um formato padrão para as descrições do programa de robô reduziria também os problemas de interface. Tal formato iria ser independente dos sistemas de programação e de controladores.

3.4.6 A IMPORTÂNCIA DA CALIBRAÇÃO

Segundo CRAIG [10], apesar de alguns resultados impressionantes, a maioria destes sistemas com habilidade para simular e programar robôs industriais fora de linha é ainda usada em estudos de planejamento de células de manufatura e sua simulação, com apenas uma minoria de sistemas usada para verdadeira geração de programas fora de linha. Sem a implementação de um pacote de calibração, não podem ser gerados programas de robô fora de linha de forma confiável, posto que a célula simulada não representa com precisão a cinemática nominal (definida pelo controlador de robô) e a cinemática atual de robô. As discrepâncias decorrentes das folgas, desgaste e uso dos eixos, ou até mesmo do processo de manufatura, podem não ser consideradas para a simulação do robô. Erros secundários na cinemática do robô simulado podem conduzir a erros maiores no chão de fábrica. Claramente, as características do dispositivo podem diferir de robô para robô por causa das diferenças na fabricação do robô, ou no seu uso.

Sendo assim, devido às diferenças implícitas entre um modelo teórico idealizado e as variações inerentes do mundo real, geralmente as seqüências simuladas não podem alcançar o objetivo de condução do robô sem erros. Na prática, o robô não entra para o lugar dito previamente pelo modelo, ou a peça de trabalho não está precisamente na localização definida no modelo.

YONG *et al.*[56] ressaltam que estas discrepâncias podem ser atribuídas aos seguintes componentes:

- *O ROBÔ* -

(a) Tolerâncias insuficientemente justas, usadas na fabricação das articulações de robôs, dão lugar a variações nas juntas. Erros pequenos na estrutura podem levar a erros bastante grandes à ferramenta.

(b) Falta de rigidez da estrutura de robô. Isto pode causar erros sérios em condições de carregamento pesado.

(c) Incompatibilidade entre robôs. Nenhum robô de fabricação e modelo idênticos executará o mesmo programa fora de linha sem pequenas divergências. Isto é causado pela combinação da calibração do sistema de controle e dos problemas de tolerâncias acima comentados.

- *O CONTROLADOR DE ROBÔ* -

(a) Resolução insuficiente do controlador. A resolução especifica o menor incremento de movimento realizável pelo controlador.

(b) Precisão numérica do controlador. Isto é influenciado pelo microprocessador e pela eficiência dos algoritmos usados para os propósitos de controle.

- *O ESPAÇO DE TRABALHO* -

(a) A dificuldade na determinação de localizações precisas de objetos (robôs, máquinas, peças de trabalho) com relação a uma dada referência dentro do espaço de trabalho.

(b) Efeitos ambientais, como temperatura, podem afetar adversamente o desempenho do robô.

- *O SISTEMA DE MODELAGEM E DE PROGRAMAÇÃO* -

(a) A precisão numérica do sistema computacional de programação.

(b) A qualidade dos dados do modelo mundo real. Isto determina a precisão final do programa fora de linha.

MORRIS [33] ressalta alguns pontos importantes a serem considerados neste tipo de problema. Para que um sistema de programação fora de linha possa ser utilizado com eficiência ótima em um ambiente industrial, o robô tem que ter um ponto de origem conhecido. E ainda, tem que ter todos os dispositivos mecânicos, de entrada e saída, acoplados com os equipamentos do processo e colocados em uma localização registrada, conhecida.

CRAIG [10] acrescenta que para calibração são necessárias medidas atuais do robô para dirigir o algoritmo de calibração. Os parâmetros cinemáticos que devem ser ajustados pelo algoritmo de calibração incluem compensações de ferramenta, da localização zero da articulação, dos parâmetros dos atuadores, e dos elos.

YONG [56] é otimista, e comenta que conforme a programação fora de linha se torne uma ferramenta prática, a condução das discrepâncias de magnitude significante na composição dos efeitos dos erros, através de todo o sistema de programação fora de linha, deve ser reduzida a um nível em que possam ser automaticamente realizados ajustes dos posicionamentos finais.

3.4.7 UMA BREVE DISCUSSÃO

Como visto, algumas precauções devem ser tomadas com a programação fora de linha embora esta seja de grande utilidade no espaço de trabalho industrial moderno.

A necessidade de definir um robô em termos de sua repetibilidade e precisão. É essencial que uma distinção seja feita entre estas duas características. A repetibilidade implica na habilidade de um robô ou outro dispositivo de automatização para retornar a um ponto ensinado

no espaço depois de programado. A tolerância com que eles retornam ao ponto ensinado é definida como a repetibilidade para aquela unidade. Precisão implica na habilidade do robô para mover uma distância escalar em uma direção vetorial. A diferença torna-se extremamente importante quando é discutido a programação fora de linha. Se um sistema robótico não tem a capacidade de movimentar-se com a precisão e a repetibilidade exigidas pela aplicação, uma discussão de programação fora de linha para aquela unidade é irrelevante.

A maioria dos fabricantes de produtos dentro da indústria não tem conhecimento aplicável para adotar a programação fora de linha. Enquanto a programação fora de linha é aceita amplamente nas indústrias para a fabricação com metais, como o CNC nos trabalhos de desbaste e polimento, a vasta maioria dos interessados nas industrias não estão familiarizados para este tipo de pensamento. Os fabricantes que produzem bens de consumos leves, tecidos, ou componentes de baixo valor agregado têm pequeno ou nenhum conhecimento no campo de programação fora de linha. Então, as dificuldades encontradas na implementação de um sistema robótico, podem estar fazendo com que os fabricantes hesitem em sua utilização. Neste sentido, o ensino nas faculdades e o treinamento na industria para familiarização com as atividades de sistemas robóticos são de extrema necessidade.

Alguns autores, como SOLAND [49], ressaltam; "um sistema de CAD/CAM pode ser uma valiosa ferramenta para o projetista de uma estação de trabalho de robô, mas não é nada mais que uma ferramenta. ***O projetista deve ter mais que um conhecimento básico de robôs***. Dentro das faixas de alcance de uma aplicação de robôs, uma seleção rígida tem que ser feita com respeito à capacidade de carga, repetibilidade, opções do software etc., antes que qualquer simulação útil em um sistema de CAD possa ser iniciada. O projetista também deveria estar familiarizado com o processo específico em que o robô será usado; i.e, o projetista precisa saber quais as velocidades e torques permitidos, velocidades de corte, etc. Sem tal informação, a simulação em um sistema de CAD/CAM pode ser inútil".

Finalmente, pode-se resumir que a implementação da programação fora de linha encontra problemas em três áreas principais. Primeiro há ***dificuldades de desenvolvimento de um sistema de programação generalizado*** que seja independente de robôs e das aplicações de robôs. Em segundo, para reduzir a ***incompatibilidade entre os robôs e os sistemas de programação***, padrões precisam ser definidos para as interfaces. Em terceiro, os programas fora de linha têm que responder por ***erros e inexatidões que existem no mundo real***.

CAPÍTULO 4

CAD/CAM E A PROGRAMAÇÃO DE TAREFAS DOS ROBÔS

4.1. INTRODUÇÃO

Os desenvolvimentos obtidos no uso de sistemas CAD/CAM para a programação dos robôs industriais certamente constituirão uma ferramenta importante para a ampliação da adaptabilidade e da versatilidade do uso destes equipamentos em atividades de fabricação. Como comentado no capítulo anterior, pode-se observar que o campo de aplicação destes sistemas em robótica é muito vasto. Verifica-se uma grande preocupação na procura de novos métodos, mais adequados ao planejamento e programação das tarefas dos robôs industriais, que atendam às necessidades de generalidade, eficiência, rapidez e qualidade dos dados gerados.

Este trabalho visa especificamente dar início ao desenvolvimento de uma proposição de sistemática [53, 54] para o planejamento e a programação de tarefas dos robôs industriais com o uso das técnicas de CAD e CAM. Tem como finalidade estabelecer uma metodologia balizada na idéia principal, que consiste em utilizar os dados das superficies de contorno do modelo geométrico do produto na atividade de planejamento de trajetórias de manipuladores. É um

"plano de processo" que possibilita a geração e análise dos dados das trajetórias, e que será usado para a programação fora de linha dos robôs industriais. O método é estabelecido de forma generalizada, permitindo que o programa desenvolvido [13, 14] sirva para fins didáticos, pesquisa e industriais. Na indústria pode ser utilizado para definir tarefas de operações de fabricação diversificadas, como por exemplo: solda, pintura, polimento e mesmo usinagem. Através da integração dos dados das superfícies de contorno do produto, do "plano de processo", e do uso de formulações de curvas, o movimento do efetuador do manipulador é definido em uma, ou mais, trajetórias contínuas.

A seguir, temos uma descrição inicial da proposição, e o detalhamento das atividades da metodologia aplicada e do sistema desenvolvido para a programação fora de linha dos robôs industriais.

4.2. DESCRIÇÃO BÁSICA DA METODOLOGIA

A proposta de interface para programação fora de linha é constituída, de maneira simplificada, por três blocos de atividades. A figura 4.1, a seguir, apresenta um diagrama esquemático destes blocos. As atividades de projeto do produto, de planejamento, de programação, e de execução das tarefas devem ser dadas. Cada bloco consiste em uma série de atividades e decisões não mostradas no presente diagrama, mas que serão descritas ao longo deste capítulo.

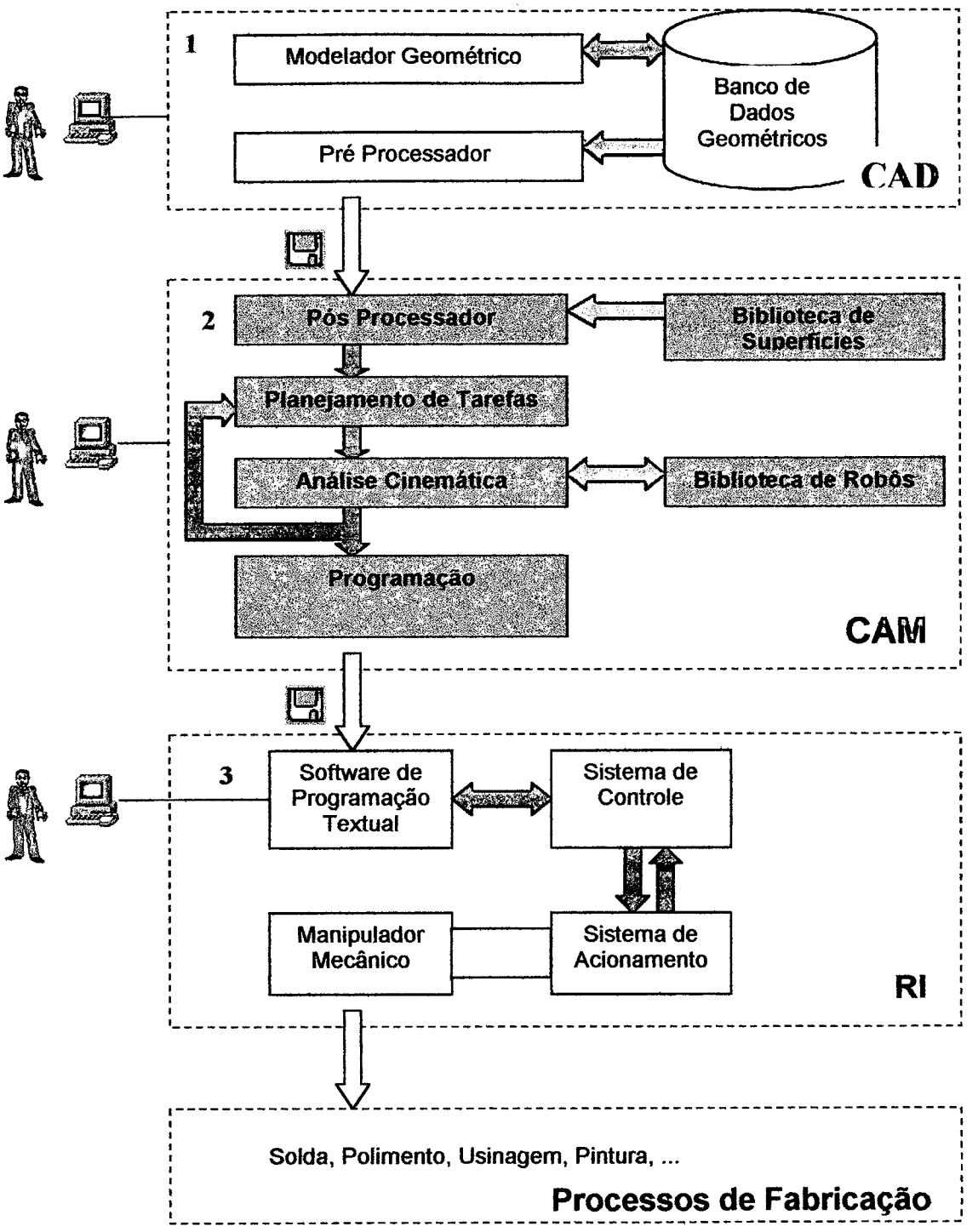

FIGURA 4.1: Diagrama Esquemático da Arquitetura da Interface CAD/CAM para o Planejamento e Programação de Tarefas dos Robôs Industriais.

No primeiro bloco a atividade é realizada no sistema CAD, onde a síntese de projeto dos produtos a serem manipulados e fabricados é concebida e armazenada. A partir desta, são identificados os dados e as informações geométricas do objeto, para serem usados nos procedimentos de programação. No projeto do produto, o projetista já antevê e separa as informações essenciais para realizar a troca de dados com o sistema de programação.

No bloco seguinte está a fase na qual ocorre o planejamento, a análise e a programação fora de linha dos movimentos dos manipuladores, a fim de que uma tarefa sobre o contorno de um objeto seja efetuada. Os dados descritivos do objeto, extraídos e identificados no bloco anterior, são usados com um plano de movimentação para realizar a definição da trajetória cartesiana a ser percorrida pelo efetuador do manipulador. Em seguida, as trajetórias são usadas como dados de entrada para a análise do comportamento cinemático de um modelo de manipulador selecionado para executar a tarefa. Com isto, determinam-se as posições, as velocidades e as acelerações cartesianas e de juntas correspondentes às configurações do sistema de coordenadas do efetuador, em cada ponto da trajetória, permitindo realizar a análise do comportamento cinemático do manipulador para o cumprimento da tarefa.

Finalmente, as trajetórias de juntas ou cartesianas escolhidas para a consecução das tarefas são transcritas através de um arquivo de programação textual específico do sistema de programação do robô industrial. Para que finalmente o programa possa ser carregado na memória do controlador e então ser executado.

4.3 A Aquisição de Dados a partir do Software de CAD

Como visto no capítulo anterior, os sistemas de CAD possuem várias possibilidades de definição de objetos. Uma vez que o modelo do produto foi gerado, esta informação pode ser aproveitada eficientemente nos procedimentos de programação dos robôs industriais,

possibilitando obter como benefício direto a redução dos ciclos de tempo e de custo da programação das atividades de fabricação do produto e, ainda, a melhoria da sua qualidade. Estes sistemas computacionais, que permitem ao usuário interagir com uma interface gráfica interativa amigável, possibilitam ao projetista realizar a definição, o armazenamento e a manipulação do modelo do produto através da sua modelagem geométrica ou mesmo a modelagem tecnológica. A principio o sistema utiliza apenas os dados geométricos.

De modo geral, como visto na seção 3.3.1, os modelos geométricos básicos de representação de dados para a tarefa de programação podem advir de modelos "wireframe", de superfícies ou mesmo de modelamento sólido. Nesta etapa o objetivo é estabelecer as informações da estrutura para a representação e comunicação digital dos dados de definição dos produtos[1].

Na metodologia aplicada nesta pesquisa, podem ser utilizados sistemas CAD que apresentam os dois últimos tipos de representação do produto. Com eles fica mais fácil fazer, a partir do arquivo de dados, a identificação dos elementos que compõem o contorno do produto. Esta identificação é dada pela extração de informações sobre as superfícies que definem o contorno do objeto. O processo de aquisição de dados pode ser feito a partir de sistemas CAD, tais como o "Microstation", o "Autocad", o "Ideas" e outros.

A forma de importação e exportação de arquivos pode ser dada através da comunicação direta ou indireta., como mencionado na seção 3.3.3. Neste trabalho, opta-se pela comunicação indireta e pelo uso de um arquivo em formato neutro. Com o uso de um interpretador de arquivos, recuperam-se somente as informações necessárias à manipulação da geometria do contorno do objeto.

[1] Dados de Definição do Produto: é o conjunto de elementos de dados que definem completamente o produto. Neste conjunto incluem-se a geometria, a topologia, as características, as tolerâncias e as relações necessárias para definir completamente um componente ou grupo de componentes.

Considera-se que os arquivos advindos dos sistemas CAD devem estar no formato padrão IGES. A partir dele, são obtidas então as informações geométricas para a recomposição do contorno da peça, onde será executado o planejamento de trajetórias. Outros formatos são considerados, mas não estão inseridos no presente desenvolvimento.

4.3.1 O PADRÃO IGES

Esta especificação define um formato neutro padronizado para troca de informações digitais entre sistemas CAD e também outras aplicações gráficas com natureza de representação vetorial [58]. O Padrão é desenvolvido e mantido pela "IGES/PDES Organization" (IPO), que é uma área da "U.S. Product Association". Constantemente são lançadas novas versões devido a propostas de seus usuários para incrementar sua funcionalidade.

A definição desta especificação possibilita a troca entre bancos de dados de sistemas CAD de diferentes softwares de CAD/CAM, cuja estrutura é incompatível. As aplicações suportadas pelo IGES incluem desenhos tradicionais de engenharia, modelos para análise e também funções para manufatura de produtos. Adicionalmente à especificação geral, a especificação IGES também inclui protocolos no qual o padrão é interpretado para exigências específicas de certas disciplinas [61].

Os protocolos aplicativos são sistemas de informação do ponto de vista de engenharia de produtos específicos e, representam atividades necessárias para o projeto e fabricação de produtos. Os seus dados são padronizados de modo a transferir as informações necessárias para suportar estas atividades e a construção específica do produto. Trata-se de um método para realizar uma troca consistente e determinística de dados de produtos específicos entre sistemas CAD [61].

A especificação IGES tem como objetivo a compatibilização das necessidades e capacidades dos métodos de desenvolvimento dos produtos CAD/CAM. Para integrar efetivamente as tecnologias de sistemas CAD heterogêneas, a indústria necessita de mecanismos de troca que sejam amplos e dignos de confiança. As implementações de tradutores de IGES para diferentes sistemas CAD continuam sendo irregulares em sua qualidade e capacidade. Adicionalmente, a maioria das industrias não adotou um nível de controle de informações de configuração necessário para o sucesso da troca de informações em sistemas CAD, usando o IGES. Os protocolos aplicativos são desenvolvidos como um mecanismo para a resolução dos problemas envolvendo a troca de dados entre sistemas CAD [61].

Durante os últimos anos foram adicionadas sofisticadas capacidades à especificação IGES. As capacidades adicionadas têm o objetivo de acomodar mais aplicações à especificação, tornando-a mais acessível e maleável. Algumas destas mudanças aumentaram sua complexidade e ambigüidade, trazendo maior dificuldade na utilização desta especificação [61].

Atualmente nenhum sistema CAD/CAM suporta a especificação IGES por inteiro no seu processador, implementa-se apenas um subconjunto da especificação. Sendo assim, tem-se apenas limitada correspondência de entidades entre os processadores de sistemas CAD não similares e uma não conformidade com a especificação IGES completa. Não se tem um critério generalizado aceitável para a identificação do quão completo é um processador da especificação IGES, ou sua conformidade com a especificação. Esta situação forçou os usuários a concentrar a utilização de sistemas CAD de modo que as entidades resultantes da especificação sejam de um mesmo subconjunto do IGES em ambos os processadores de troca.

O uso desta especificação estabelecida permite a troca compatível de dados entre vários sistemas de CAD/CAM. Nesta, um arquivo estruturado é especificado num determinado formato de linguagem para a representação da definição do produto por dados geométricos, topológicos e não geométricos. Os dados para a representação do produto neste formato podem ser trocados

entre diversos aplicativos computacionais. Esta metodologia de representação de produtos de dados[2] é extensa e independente do método de modelagem geométrica[58].

Os dados requeridos para a descrição e comunicação das características essenciais em engenharia de objetos físicos como produtos manufaturados são tratados. Estes produtos são descritos em termos de sua forma física, dimensões e informações adicionais que descrevem ou explicam o produto. Processos típicos que utilizam produtos de dados incluem projeto, análise de engenharia, planejamento de produtos e fabricação [58].

O formato do arquivo nesta especificação trata o produto como um arquivo de entidades independentes que mapeiam a representação do produto. Sendo assim, as entidades são a unidade fundamental de informação, podendo ser categorizadas em geométricas e não geométricas. O arquivo para exportação de dados entre sistemas CAD/CAM deve no mínimo suportar a comunicação de dados geométricos, anotações e os dados da organização.

As entidades geométricas representam a definição da forma física e incluem pontos, curvas, superfícies e relações entre coleções de entidades estruturadas similares.

As entidades não geométricas habitualmente servem para enriquecer o modelo com vistas em perspectiva na qual o desenho planar deve ser composto e com anotações e dimensionamento apropriados do desenho. Servem também para especificar características e atributos dos grupos de entidades.

4.3.1.1 O ARQUIVO ESTRUTURADO NO FORMATO ASCII

Dois diferentes formatos são definidos para a representação de dados em IGES. São estes o ASCII e o Binário [58]. O formato ASCII utiliza uma estrutura de caracteres orientados e o formato binário utiliza uma estrutura de bytes orientados. A utilização no formato ASCII é de

[2] Dados de Produto: é o conjunto de elementos de dados necessário para o suporte completo de um produto. Este

mais fácil implementação, porém produz excessivo volume de dados, enquanto que o formato binário é mais complexo, mas oferece uma redução no volume de dados. No formato binário a descrição é tratada como uma corrente contínua de blocos de bits entre os sistemas de envio e de recebimento, já o arquivo no formato de conjunto de caracteres apresenta 80 colunas num registro de dados.

O arquivo no formato ASCII permite a observação e o entendimento dos tipos de dados que são trocados entre os sistemas de envio e recebimento. A figura 4.2 apresenta um arquivo no formato ASCII da especificação IGES gerado no pré-processador "Microstation IGES 95" da "Bentley Inc".

O arquivo é composto de cinco seções: início, condições globais, entrada de diretórios (DE), parâmetros de dados (PD), e término, separadas por linhas verdes na figura. O arquivo pode conter qualquer número de entidades de modo a representar o produto definido, neste caso, tem-se três entidades do tipo linha. Cada ocorrência de entidade consiste em um registro de diretório de dados e um registro de parâmetros de dados. O registro de diretório provê um índice e inclui atributos descritivos sobre o dado, enquanto que um parâmetro de dados provê definições específicas da entidade, variáveis em comprimento e formato. Os diretórios de dados são organizados com campos fixos e consistentes para todas as entidades de modo a facilitar o acesso aos dados descritivos destas. Os diretórios e os parâmetros de dados de todas as entidades do arquivo são organizados em seções separadas com ponteiros bidimensionais, conforme ilustrado na figura com linhas vermelhas e azuis, realizando a ligação entre o registro de diretório e os parâmetros de cada entidade.

---

conjunto inclue dados de definição do produto e outros dados necessários para a operação e manutenção do produto.

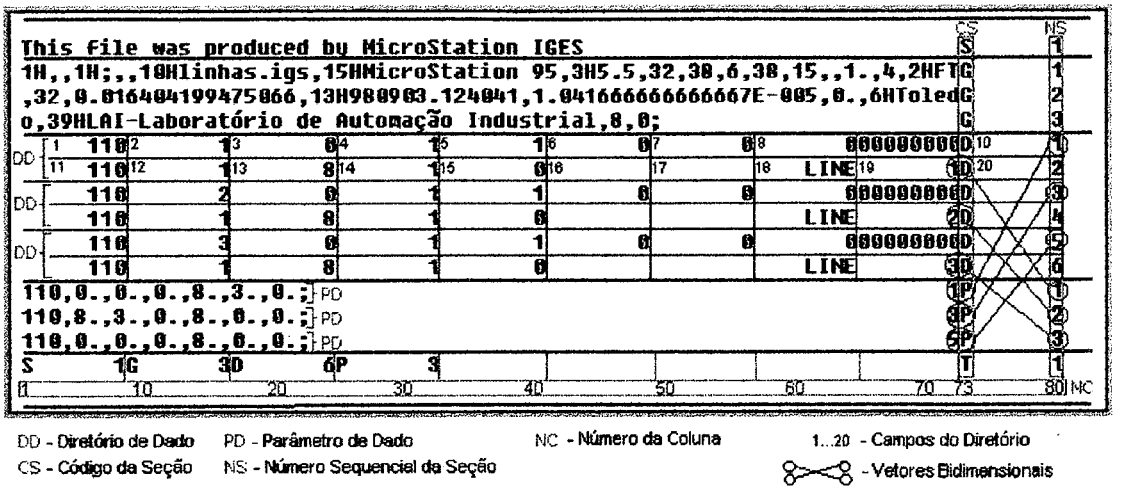

FIGURA 4.2 : Exemplo de Arquivos IGES gerado no Microstation IGES 95 na troca de dados de três entidades do tipo linha (LINE).

As letras de código da seção (CS) são inseridas na coluna 73, como a seguir:

• Seção de Início	(Start Section)	S
• Seção Global	(Global Section)	G
• Seção de Entrada de Diretório	(Directory Entry Section)	D
• Seção de Parâmetros de Dados	(Parameter Data Section)	P
• Seção de Término	(Terminate Section)	T

Os dados das seções do arquivo são entrados em formato livre, como descrito abaixo [58]:

- Espaços em branco são ignorados, a menos em strings
- São utilizadas vírgulas como delimitadoras de campo, separando os parâmetros
- Ponto e vírgula é um delimitador de registro, sendo utilizado para terminar a lista de parâmetros
- Duas vírgulas adjacentes significam que o parâmetro não foi especificado no arquivo e então este toma um valor "default".
- Ponto e vírgula aparecendo antes do fim da lista de parâmetros implica em valores "default" para o restante dos parâmetros

A seção de início deve conter uma introdução ao arquivo. Todos os registros nesta seção, bem como nas outras seções, devem ter uma seqüência de números (NS) nas colunas 74 à 80 identificando um ponteiro. A informação das colunas 1 à 72 não tem formatação especial, exceto quanto a que caracteres ASCII devem ser usados.

A seção global do arquivo contém informações descrevendo o pré-processador e informações necessárias para o pós-processador, de modo que o arquivo possa ser manuseado. Dentre estes dados tem-se: do tipo "string", que identifica, por exemplo, o sistema de proveniência, o nome do arquivo IGES, a versão do tradutor; do tipo inteiro, indicando o número de bits para representação de números inteiros, o número de bits do expoente de pontos flutuantes de simples precisão e o número de bits da mantissa de pontos flutuantes de simples precisão; do tipo ponto flutuante, que identifica o número máximo de graduações do peso de linhas e a aproximação do máximo valor de coordenada ocorrendo no modelo.

A seção de entrada de diretórios (DE) contém uma entrada para cada entidade no arquivo. Estas entradas, para cada entidade, têm tamanho fixo e contêm vinte campos de oito caracteres. O propósito desta seção é indexar o arquivo contendo informações de atributos para cada entidade. O número do tipo de entidade identifica o tipo de entidade, o número da versão indica como interpretar os parâmetros desta entidade e o número da forma da entidade identifica a forma da entidade para os casos em que estas podem ter diferentes interpretações e o rótulo da entidade é um identificador ou nome da entidade em alpha numérico.

A seção de parâmetros de dados contém os dados dos parâmetros associados a cada entidade. Os parâmetros têm formato livre até a coluna 64, com o primeiro campo contendo o número da entidade. O tipo da entidade e um delimitador de campo precedem o primeiro parâmetro de cada entidade. As colunas 66 até 72, em todos os registros de parâmetros, contêm o número seqüencial do primeiro registro do diretório de entrada da entidade à qual o parâmetro se refere. Comentários podem ser adicionados após o delimitador do registro.

A seção de término do arquivo possui apenas um registro com dez campos de oito colunas cada. É o último registro do arquivo e possui um número seqüencial com valor 1 nas colunas 74 à 80. Os campos do registro terminal possuem um caractere representando o tipo de seção e o último número seqüencial desta.

4.3.1.2 AS ENTIDADES GEOMÉTRICAS DE INTERESSE

Existem diversos tipos de entidades geométricas disponíveis para serem utilizadas neste arquivo de definição de produtos por entidades. Os números de tipos de entidades de 100 à 199 são reservados para entidades geométricas [58].

A seguir estão descritas algumas das entidades geométricas que são de interesse para a formulação desenvolvida, bem como os números de tipo de entidade determinados:

(112) ***Curva "Spline" Paramétrica*** – são seqüências de segmentos polinomiais parametrizados com grau 1, 2 ou 3. Esta entidade também representa as várias "splines" existentes em sistemas CAD: lineares, quadráticas, cúbicas, Wilson-Fowier, Wilson-Fowier modificada e "B-splines". Se a curva for planar, ele deve ser parametrizada em termos de x e y apenas. Adicionalmente tem-se o parâmetro H que especifica o grau de continuidade nos pontos extremos dos segmentos.

(114) ***Superfície "Spline" Paramétrica*** - são definidas por um rede de segmentos polinomiais parametrizados. Devido à sua generalidade, esta entidade representa diversas superfícies utilizadas em sistemas CAD, entre estas, superfícies de Bézier e "B-Splines".

(118) ***Superfícies de Translação*** - são definidas por um eixo de translação, que deve ser uma entidade do tipo linha, e por uma geratriz. A superfície é gerada pela translação da geratriz ao longo do eixo de translação. (Figura 4.4)

(120) ***Superfícies de Revolução*** - são definidas por um eixo de rotação, que deve ser uma entidade do tipo linha, por uma geratriz e ângulos de rotação inicial e final. A superfície é gerada pela rotação da geratriz em torno do eixo de rotação desde o ângulo inicial até o final. (Figura 4.5)

(122) ***Cilindro*** - é uma superfície formada pela translação paralela de um segmento de linha, chamado de geratriz, ao longo de uma curva, chamada de diretriz. Esta curva pode ser uma linha, um arco circular, um arco cônico, uma curva "spline" parametrizada ou uma curva composta. (Figura 4.6)

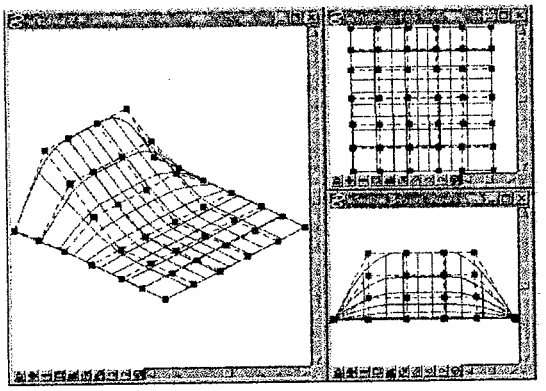

FIGURA 4.3: Superfície Livre gerada no Microstation 95.

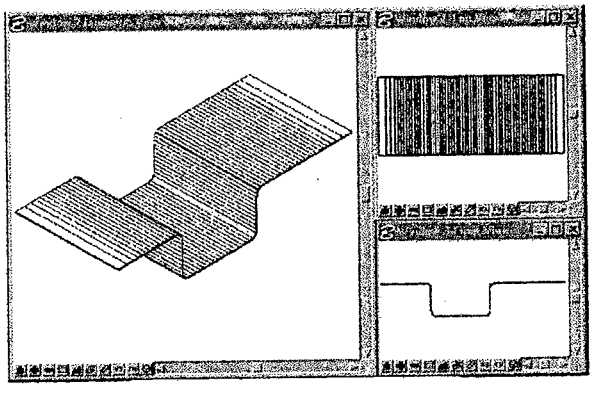

FIGURA 4.4: Superfície de Translação gerada no Microstation 95.

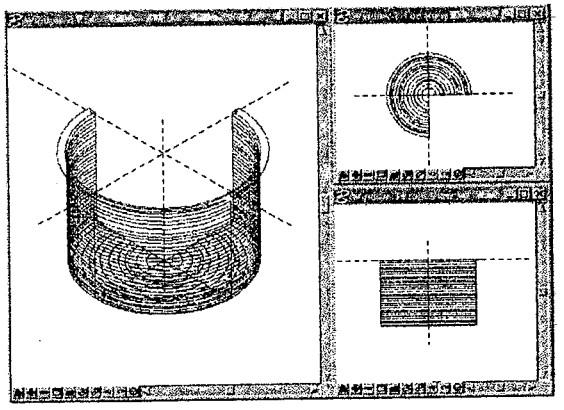

FIGURA 4.5: Superfície de Revolução gerada no Microstation 95.

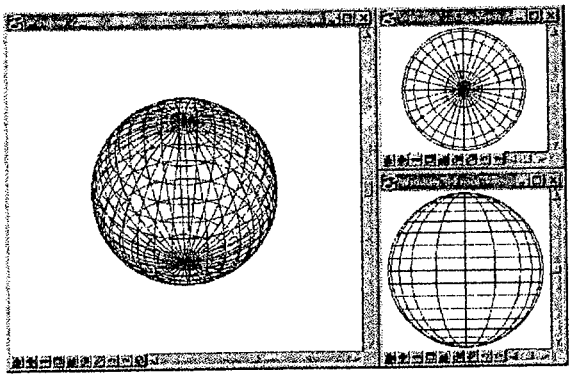

FIGURA 4.6: Superfície esférica gerada no Microstation 95.

(126) Curvas "B-Splines" Racionais - representam curvas analíticas. A entrada de diretório "Número de Forma" deve possibilitar a comunicação de informações importantes para a especificação da curva no sistema de envio e de recebimento, mas ambos os sistemas devem operar diretamente em "B-Splines" Racionais.

(128) Superfícies "B-Splines" Racionais - representam superfícies analíticas de grande interesse. O conhecimento desta metodologia é importante para ambos os sistemas, o de envio e o de recebimento. Assim como para curvas, a entrada de "Número de Forma" provê a comunicação das informações necessárias para a geração de "B-Splines" Racionais.(Figura 4.3)

4.4 O MÉTODO DE PLANEJAMENTO DE TAREFAS DE FABRICAÇÃO NO ESPAÇO CARTESIANO

O módulo de planejamento de tarefas tem como função principal permitir ao usuário estabelecer, com certa habilidade e versatilidade, trajetórias contínuas no espaço cartesiano para que, em seguida, possa ser realizada a análise destas no espaço de juntas de um manipulador específico, através do uso da cinemática inversa, e conseqüente verificação e validação da movimentação. Visa, com isto, fornecer os valores de referencia mais apurados para o controle da tarefa do robô industrial.

Embasado nas técnicas consagradas de desenvolvimento de sistemas CAD/CAM, o procedimento de definição da tarefa possibilita integrar num ambiente amigável gráfico interativo os dados do contorno de um produto ou objeto, extraídos do banco de dados geométrico de sistemas atuais de CAD, para serem interpretados e identificados com um conjunto de modelos de superfícies que são manipuláveis pelo sistema. Assim é possível o desenvolvimento de algoritmos para realizar diretamente da superfície do objeto o planejamento das trajetórias do efetuador final do manipulador para diversificadas atividades de fabricação,

objetivando-se deste modo, automatizar e qualificar a geração de dados nos procedimentos de programação dos robôs industriais.

Nas seções seguintes serão detalhados os passos do procedimento de determinação de tarefas. A especificação de uma tarefa pode ser dividida em dois estágios: a seleção dos pontos sobre o contorno do objeto e a geração da trajetória entre estes pontos. A finalidade do planejamento de tarefas é gerar os dados da trajetória cartesiana do efetuador para execução da tarefa.

4.4.1 O MÉTODO DE SELEÇÃO DOS PONTOS

O conjunto de superfícies de interesse para o procedimento, comentado na seção 4.2, são as superfícies paramétricas, detalhadas no Apêndice 2, que permitem o desenvolvimento da técnica detalhada a seguir para a definição dos pontos cartesianos do percurso da tarefa.

Detalhando matematicamente, é possível representar analiticamente uma superfície $S \in R^3$, via uma parametrização. Em outras palavras, podemos obter uma parametrização do tipo

$$\begin{aligned} &U = [a\ ,\ b] \times [c\ ,\ d] \\ &g : U \rightarrow S \\ &g(u,v) = [x(u,v)\ ,\ y(u,v)\ ,\ z(u,v)] \end{aligned} \qquad (4.1)$$

para S e representar todos os percursos sobre $S \in R^3$ pela sua correspondente em $U \in R^2$. Isto é, se $\gamma : [s_0, s_1] \rightarrow S$ é um percurso possível, existe $\alpha : [s_0, s_1] \rightarrow U$ com a mesma regularidade tal que

$$\gamma(s) = g(\alpha(s)) = g(u(s), v(s)) \qquad (4.2)$$

Com isto, a determinação das atividades de fabricação sob o contorno do objeto pode ser realizada partindo de diversas propostas de percursos $\alpha(s)$ para seleção dos pontos a serem utilizados na geração de trajetórias. A figura 4.8 ilustra uma possível proposição para geração de um percurso de pontos nos parâmetros de uma superfície paramétrica qualquer.

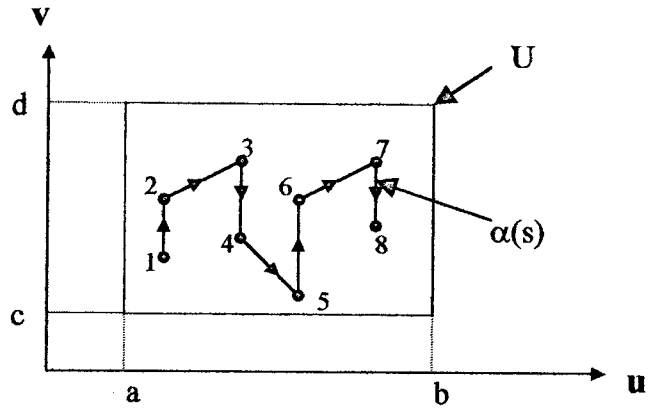

FIGURA 4.8: Procedimento de definição de percursos de pontos α(s) nos Parâmetros **u** e **v** de uma superfície paramétrica qualquer.

Neste momento podem ser determinadas algumas variedades de percursos para a execução pelo robô industrial. A figura 4.9 mostra alguns tipos básicos de movimentos sobre uma superfície plana que podem ser facilmente determinados. Eles são definidos como: (a) movimento em espiral partindo das bordas para o centro, (b) movimento em zig-zag, (c) movimento livre. Outros diversos tipos de movimentos são também possíveis de se determinar inclusive através de curvas ao invés de percursos de pontos.

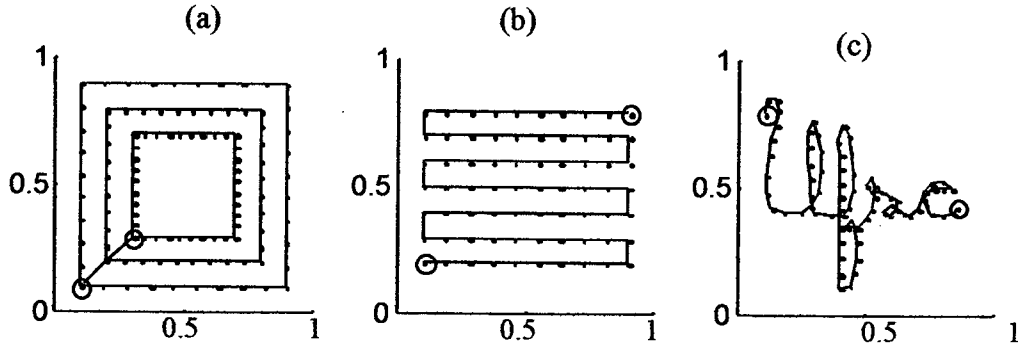

FIGURA 4.9: Demonstração gráfica em MatLab[54] dos dados de entrada para a seleção de diferentes percursos nos parâmetros de uma superfície paramétrica qualquer.

A seleção de pontos é determinada através de interfaces interativas, como as ilustradas na seção 5.2 de utilização do programa desenvolvido, para a definição dos parâmetros relativos aos diferentes percursos $\alpha(s)$. Um percurso em zig-zag, por exemplo, pode ser determinado através dos afastamentos do percurso com relação às bordas da superfície, o número de passos a serem executados e o número de pontos nas isoparamétricas.

O procedimento, equações (4.1) e (4.2), é ilustrado na figura 4.10, e simplifica enormemente o problema reduzindo-o a uma questão de parametrização de percursos sobre um subconjunto do plano. O modelo $g(u,v)$ da superfície (4.10b) e um percurso de pontos $\alpha(s)$ nos parâmetros da superfície (4.10), resultam num percurso de pontos $\gamma(s)$ (4.10c) no espaço externo do manipulador, permitindo posterior determinação de uma trajetória a ser coberta pelo efetuador do manipulador.

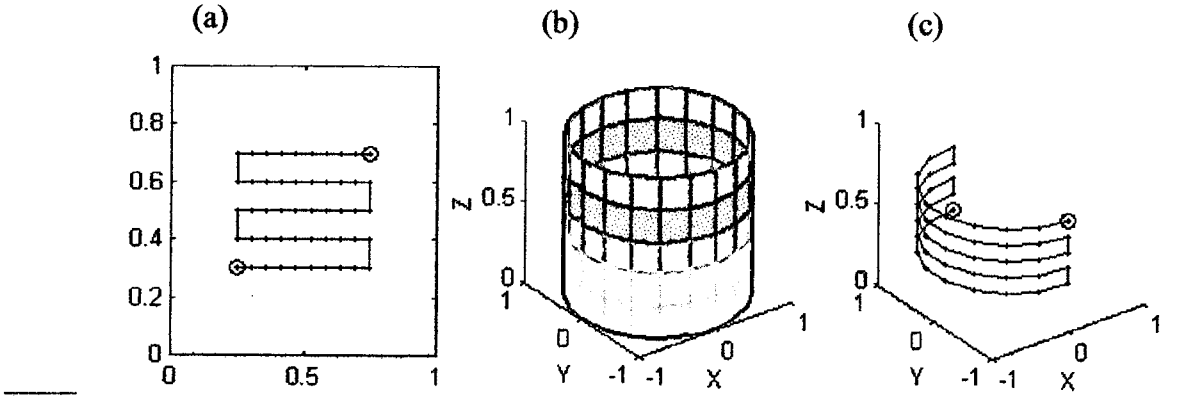

FIGURA 4.10: Procedimento de definição do percurso (c) de pontos no espaço cartesiano através de (b) um modelo de superfície e (a) um modelo de percurso nos parâmetros desta.

A seleção de um percurso de pontos nos parâmetros da superfície juntamente com a topologia da superfície definem um percurso de pontos a ser seguido no espaço cartesiano. A idéia é que uma variedade de percursos de pontos possam ser mais facilmente sugeridos e determinados para um diversificado campo de tarefas.

Agora, torna-se necessário um método para a geração de trajetórias cartesianas para que tenhamos os dados de posição, velocidade e aceleração no tempo da movimentação para o cumprimento do percurso proposto.

4.4.2 A GERAÇÃO DE TRAJETÓRIAS CARTESIANAS

A geração das trajetórias cartesianas é determinada através da história no tempo das posições obtidas no módulo de seleção dos pontos do percurso.

Como comentado na seção 3.5.2.3, de um ponto de vista matemático, uma curva gerada usando os vértices de um polígono de definição é dependente de algum esquema de interpolação ou aproximação de modo a estabelecer alguma relação entre a curva e o polígono.

Este esquema é provido por meio de uma função de base. A formulação de curvas "B-Splines" ("Basis Splines"), detalhada na referência [44], constitui um método de geração de curvas no qual utiliza-se da aproximação dos vértices de um polígono de definição através de uma função de base.

As rotinas utilizadas para a integração da geração de trajetórias cartesianas no sistema de programação fora de linha foram desenvolvidas por BOOR [07].

O Apêndice 1 apresenta uma revisão das formulações de curvas "B-splines" mais detalhada matematicamente, permitindo seu melhor entendimento. Esta revisão foi realizada a partir das referências [07] e [44], e nestas referências podem ser encontrados tratamentos ainda mais elaborados que apresentam as características e alguns exemplos detalhando estas formulações de curvas.

Neste momento será revisada a equação base utilizada nesta formulação, e serão consideradas suas características mais relevantes, a partir da referência ROGERS *et al* [44]. Em

seguida são apresentados os casos específicos de interesse que foram utilizados neste trabalho, bem como alguns resultados exemplificando-os.

Tomando P(t) como o vetor de posições ao longo da curva como uma função do parâmetro t, uma curva "B-Spline" é dada por

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t) \qquad t_{min} \le t \le t_{máx} \qquad 2 \le k \le n+1 \qquad (4.3)$$

onde B_i são os vetores de posições dos n+1 vértices do polígono de definição e $N_{i,k}$ são as funções de base normalizadas da "B-Spline".

Nesta aproximação, dentre outras propriedades importantes que podem ser mais detalhadamente observadas em ROGERS *et al* [44] e no apêndice 1, destacam-se:

1. A curva obedece a regra do polígono convexo com respeito ao seu polígono de definição.
2. A curva não oscila sobre qualquer linha reta mais do que o polígono de definição.
3. A curva geralmente segue a forma do seu polígono de definição.
4. A característica de não globalidade das "B-Splines", que se deve ao fato de que cada vértice B_i está associada a uma única função de base; então, cada vértice afeta a forma da curva apenas numa faixa de valores de parâmetros onde a função de base é não zero.
5. Qualquer transformação afim pode ser aplicada à, curva aplicando-se esta aos vértices do seu polígono de definição.
6. O ordem k da curva pode ser variada conforme a exigência de continuidade desta; k varia de 2 a um valor igual ao número de vértices do polígono de definição.

As três primeiras características são extremamente importantes, pois nos permite obter curvas de posição no espaço cartesiano que, apesar de não serem interpolantes, se aproximam dos pontos do percurso de maneira suave e comportada e, de certo modo, de maneira previsível.

A característica de não globalidade também é de extremo valor nesta formulação de curvas, pois nos permite manusear localmente certos pontos críticos destas, sem alterar o comportamento das curvas por inteiro.

A penúltima característica nos permite facilmente testar o comportamento das trajetórias cartesianas em diferentes posições no espaço, o que é extremamente valoroso em robótica, pois pequenas diferenças no espaço cartesiano podem alterar em muito o comportamento no espaço de juntas do manipulador selecionado para a execução da tarefa.

A flexibilidade e versatilidade das funções de base "B-Splines" e, portanto da curva resultante, podem ser obtidas nos diferentes modos de controle a serem usados para definir a forma da curva, destacados por ROGERS *et al* [44]:

- Mudança do vetor de referência: periódico e uniforme, aberto e uniforme ou não uniforme.
- Mudança da ordem k da função de base.
- Mudança do número e das posições dos vértices do polígono de definição.
- Utilizando vértices múltiplos no polígono de definição.
- Utilizando valores múltiplos no vetor de referência.

Para demonstrar esta etapa da atividade de planejamento da tarefa, as figuras 4.11 à 4.14, que seguem, ilustram o comportamento da aproximação de um percurso de pontos cartesianos através do uso de três diferentes formulações de curvas "B-Splines", e suas derivadas primeira e segunda, com um vetor de referência t=0:60 Aberto e Uniforme e ordem k=4, conseqüentemente grau 3, obtidas com a utilização das rotinas desenvolvidas por DE BOOOR[07] em MatLab [22].

A figura 4.11 ilustra o primeiro percurso de pontos cartesianos, que é definido através de dez pontos linearmente distribuídos numa reta P1=[0, 1, 1] à P2=[1, 0, 1]. Os outros dois percursos são idênticos ao primeiro, mas possuem vértices a mais devido à multiplicidade m=1 e m=2, respectivamente, nos vértices da extremidade do polígono de definição.

As figuras 4.12 a 4.14 ilustram, respectivamente, as posições, as velocidades e as acelerações no tempo obtidas. Nestas, pode ser observado o recurso da multiplicidade nos vértices do polígono de definição, que foi utilizado num caso particular dos extremos do percurso.

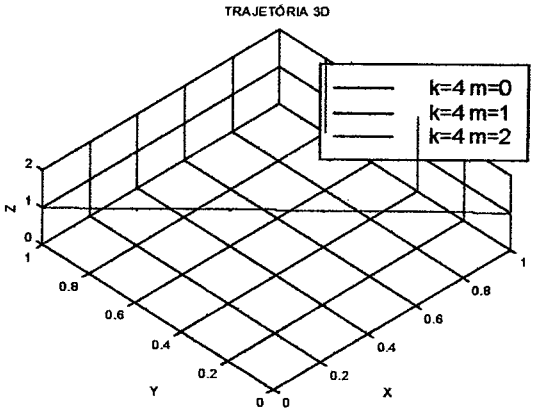

FIGURA 4.11: Percursos de pontos no espaço cartesiano com multiplicidade variante nos extremos.

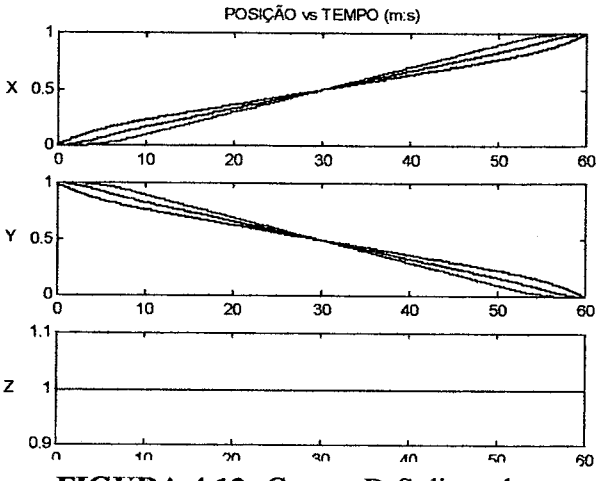

FIGURA 4.12: Curvas B-Splines de posição no tempo para a aproximação dos percursos da figura 4.11.

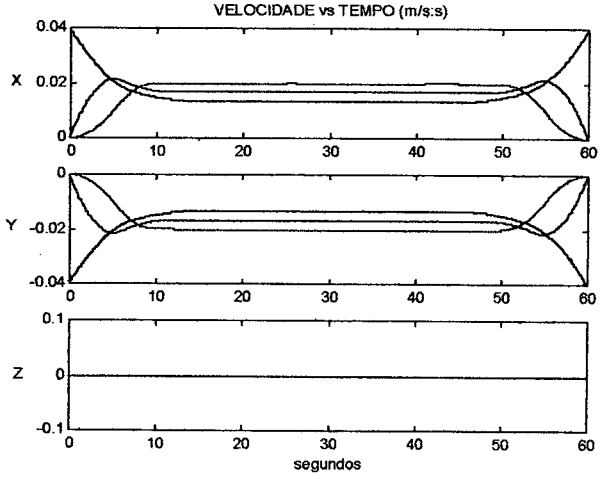

FIGURA 4.13: Derivada primeira das curvas de posição obtidas e ilustradas na figura 4.12.

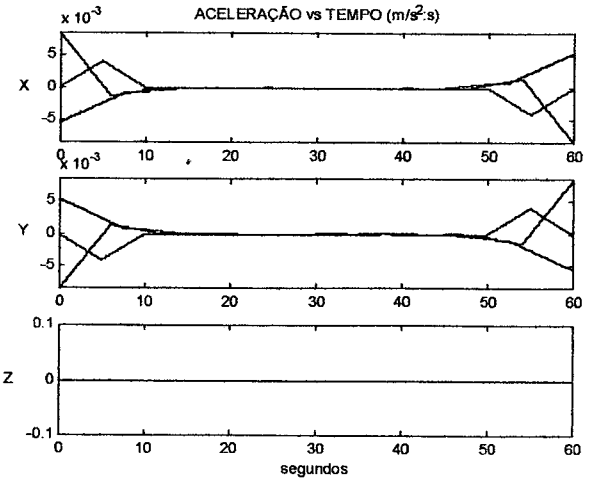

FIGURA 4.14: Derivada segunda das curvas de posição obtidas e ilustradas na figura 4.12

curva continua, que se inicia em condições nulas de velocidade e aceleração, realiza a movimentação e termina novamente com condições nulas.

A idéia básica é gerar os dados de curvas cartesianas 3D, que possam ser controlados através do uso das formulações "B-Splines" e suas propriedades. Como será visto nos testes, a devida escolha dos pontos nos parâmetros da superfície e do vetor de referência da curva "B-Spline" são extremamente úteis em robótica, pois permitem realizar um controle, com certo nível de habilidade e flexibilidade, dos dados gerados. Trata-se de um método de planejamento de certo modo genérico, por não estar preso a determinadas características das atividades de fabricação, podendo ser utilizado para diferentes tipos de atividades. E ainda, por se tratar de um método relativamente simples em que podemos realmente gerar curvas cartesianas e, como será visto na seção seguinte, transcrevê-las para diferentes manipuladores industriais, permite realizar prontamente a atividade de planejamento, análise, verificação e validação da programação.

As figuras 4.15 à 4.18 ilustram a possibilidade de extensão do método, demonstrado anteriormente, para curvas de diferentes graus.

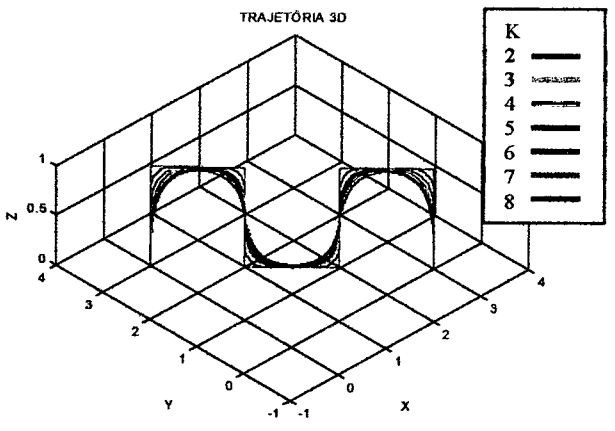

FIGURA 4.15: Percurso de pontos e as curvas de aproximação "B-Splines" com vetor de referência aberto e uniforme de acordo com variação da ordem k e dos m vértices coincidentes nos extremos.

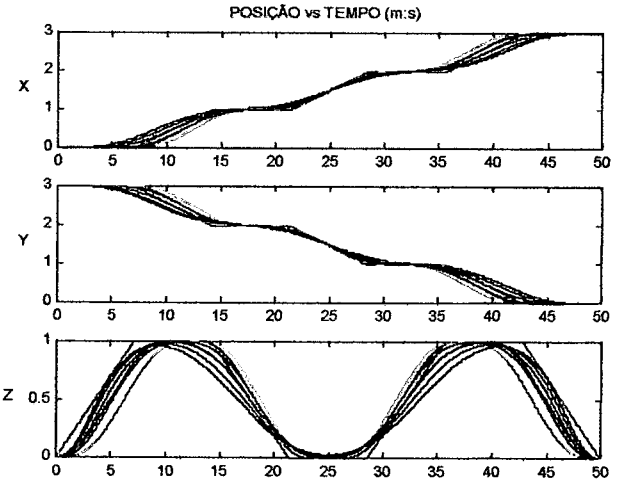

FIGURA 4.16: Curvas "B-Splines" de posição no tempo para a aproximação do percurso ilustrado na figura 4.15.

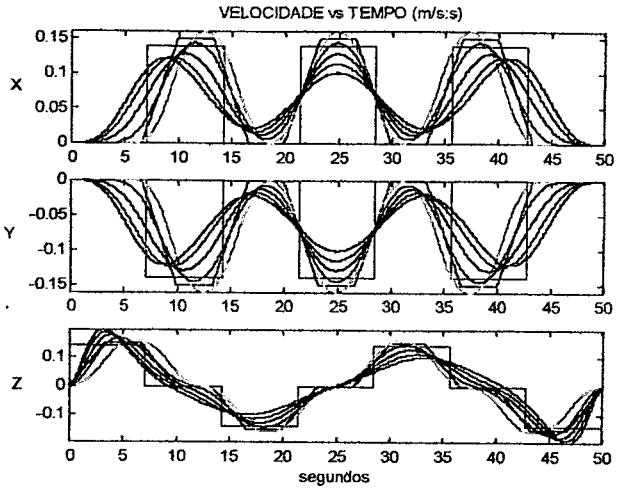

FIGURA 4.17: Derivada primeira (Velocidades) das curvas de posição obtidas e ilustradas na figura 4.16.

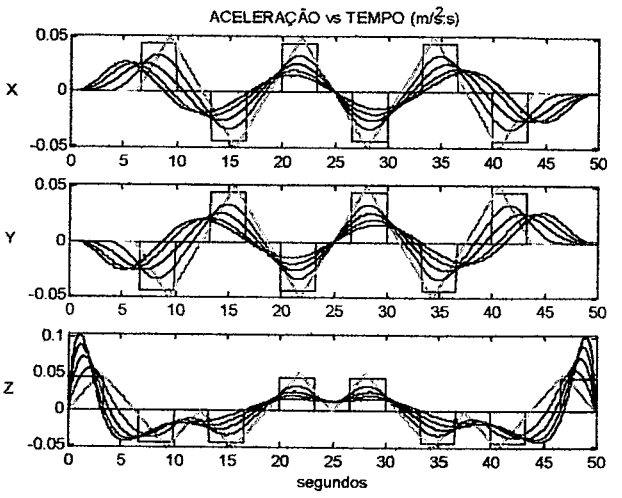

FIGURA 4.18: Derivada segunda (Acelerações) das curvas de posição obtidas e ilustradas na figura 4.16.

Neste caso não foi utilizado um percurso linear de pontos. Com isso pode-se observar mais detalhadamente o efeito da variação da ordem da curva "B-Spline". Podemos observar a suavização das curvas de ordem superior comparadas às curvas de grau inferior. As figuras 4.16 à 4.18 ilustram, respectivamente, as posições, as velocidades e as acelerações no tempo obtidas. Nestas, também pode ser observado o recurso da multiplicidade nos vértices do polígono de definição, para a obtenção de velocidades e de acelerações nulas nos extremos. Com isso tem-se um método que permite inicialmente gerar trajetórias prontas para rodar nos controladores dos robôs industriais.

Deste modo, a integração da formulação de curvas "B-Splines" desenvolvida por BOOR [07] é utilizada para aproximar as posições cartesianas, obtidas no procedimento de seleção dos pontos. Através de "B-Splines" de ordem k variáveis, conseqüentemente de grau k-1, obtém-se trajetórias contínuas de posição, velocidade e aceleração num total de tempo especificado para a execução do percurso determinado.

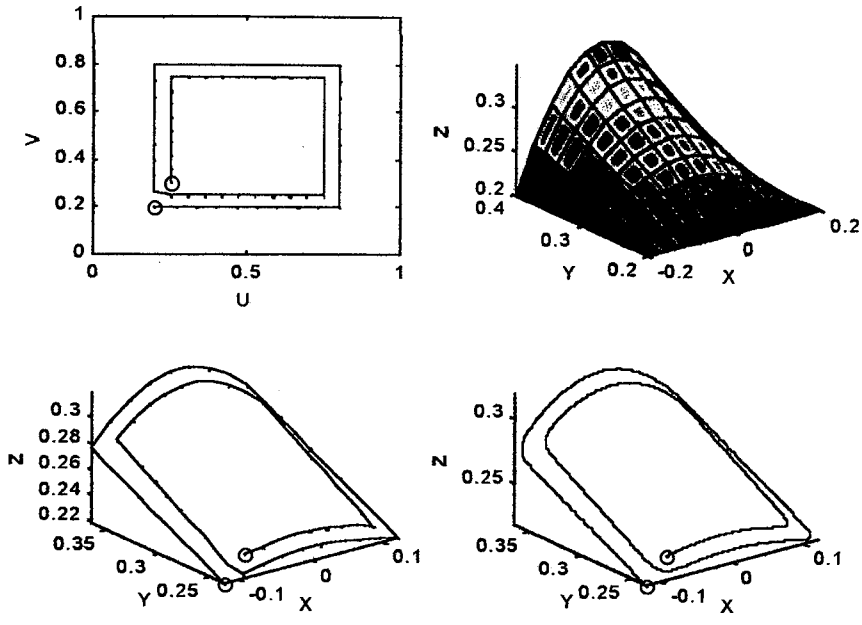

FIGURA 4.19: Ilustração demonstrativa do procedimento de geração de curvas cartesianas 3D através de um percurso de pontos (a) nos parâmetros de uma superfície (b), gerando um percurso de pontos cartesianos 3D (c), para aproximação com curvas "B-Splines" (d).

Com isso é possível gerar os dados da trajetória cartesiana. A figura 4.19 ilustra o procedimento no qual um percurso nos parâmetros de uma superfície possibilita gerar um percurso de pontos cartesianos $\gamma(s)$ para aproximação do percurso uma curva "B-Spline" P(t) no tempo.

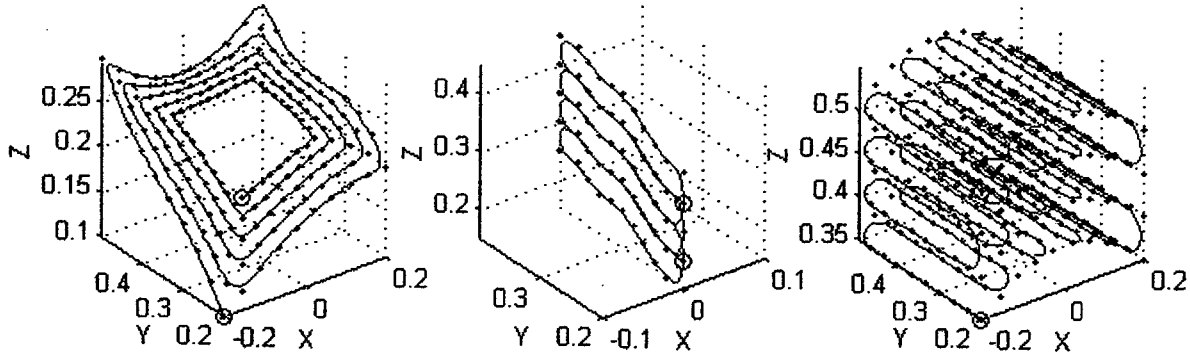

FIGURA 4.20: Construção de uma trajetória sob a superfície (a), e de trajetórias compostas através de percursos sob o contorno do objeto (b) e (c).

A figura 4.20 ilustra a possibilidade de geração de diferentes percursos e trajetórias de posições, de velocidade e as acelerações para o planejamento e definição de tarefas no espaço cartesiano através do procedimento.

A figura 4.21, demonstra a saída gráfica da interface para visualização da trajetória de posição especificada pelo procedimento de geração de tarefas ilustrado na figura 4.19.

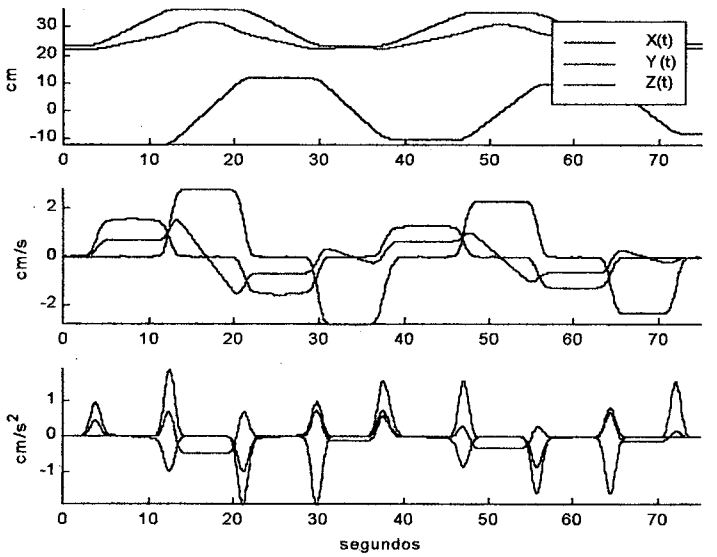

FIGURA 4.21: Visualização das curvas de posição, velocidade e aceleração no tempo para a tarefa especificada e ilustrada na figura 4.19.

O procedimento automatiza e simplifica a especificação da tarefa no espaço cartesiano 3D, garante que a trajetória para execução da tarefa percorra a forma do percurso estabelecido sob a superfície em questão, no tempo especificado pelo usuário e, ainda, podem ser avaliados tantos pontos ao longo da curva quantos necessários para a programação do controlador do robô. Tem-se uma perfeita definição de uma trajetória contínua, que pode ser analisada e otimizada em seus pontos críticos numa determinada tarefa.

Neste caso a orientação do efetuador foi especificada pela determinação de ângulos constantes da orientação do pulso, como na figura 4.21, onde então as derivadas são nulas. Mas, assim como foi obtida a trajetória de posição do efetuador do manipulador, poderíamos obter a

trajetória de orientação a partir da aproximação dos ângulos correspondentes às propriedades da superfície nos pontos, como por exemplo a normal ou a tangente à superfície.

4.5 A ANÁLISE CINEMÁTICA

Neste módulo, as trajetórias cartesianas devem ser mapeadas através da cinemática inversa, para o espaço de juntas do modelo de manipulador desejado.

Para tal, é utilizado o pacote de rotinas em MatLab [22] desenvolvidas por CORKE [09], que são endereçadas à área de robótica.

A aquisição do modelo cinemático do manipulador é feita a partir de um arquivo no qual consta uma matriz que descreve os parâmetros de sua cinemática, usando a convenção de Denavit-Hartenberg. Cada linha da matriz corresponde à um elo do manipulador e as colunas aos parâmetros de Denavit-Hartenberg e o tipo de junta de cada elo.

O cálculo da cinemática inversa do manipulador é determinado usando o método numérico, já conhecido na literatura, desenvolvido com o uso dos recursos algébricos do MatLab CORKE [09]. As soluções numéricas baseiam-se em cálculos iterativos e fornecem apenas uma entre as possíveis soluções, porém podem ser utilizadas de maneira genérica, pois ao contrário da solução analítica, independem da estrutura cinemática do robô.

A determinação da trajetória do manipulador pode ser dada de duas maneiras com a integração dos recursos desenvolvidos por CORKE [09] e BOOR [07]. Na primeira, com o uso da cinemática inversa, resolve-se o problema geométrico para obtenção das posições das juntas do manipulador selecionado a partir do percurso de pontos definido no espaço cartesiano. Então, se não houver problemas de singularidade, as coordenadas de juntas são aproximadas através das formulações "B-splines", obtendo assim as posições, velocidades e acelerações no tempo das juntas do manipulador. Na segunda, de posse das posições, velocidades e acelerações no tempo

no espaço cartesiano, geradas através da aproximação "B-Splines" do percurso de pontos cartesianos, as posições e as velocidades de juntas no tempo são obtidas com o uso do pacote de rotinas de robótica do MatLab. Porém neste caso, não há uma resposta generalizada para o mapeamento das acelerações cartesianas para o espaço de juntas.

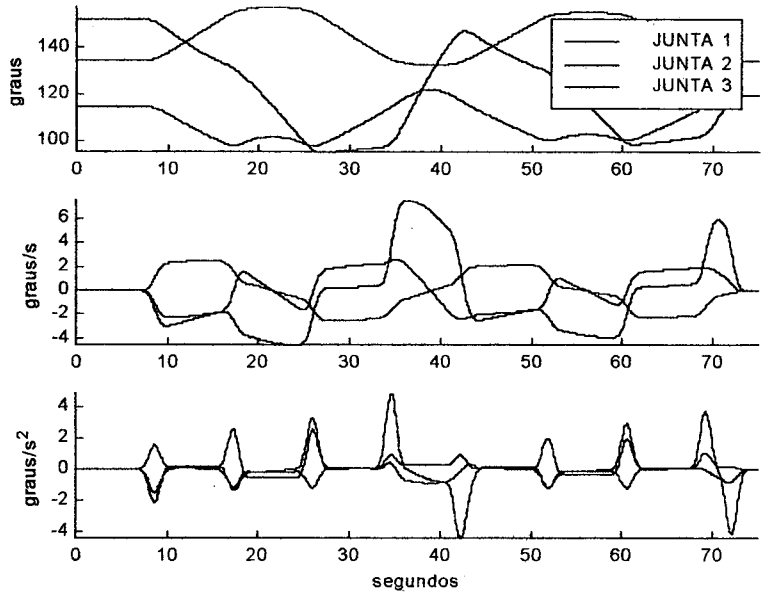

FIGURA 4.21 : Visualização das trajetórias das juntas de um manipulador tipo Puma 560 para uma tarefa como a da figura 4.19.

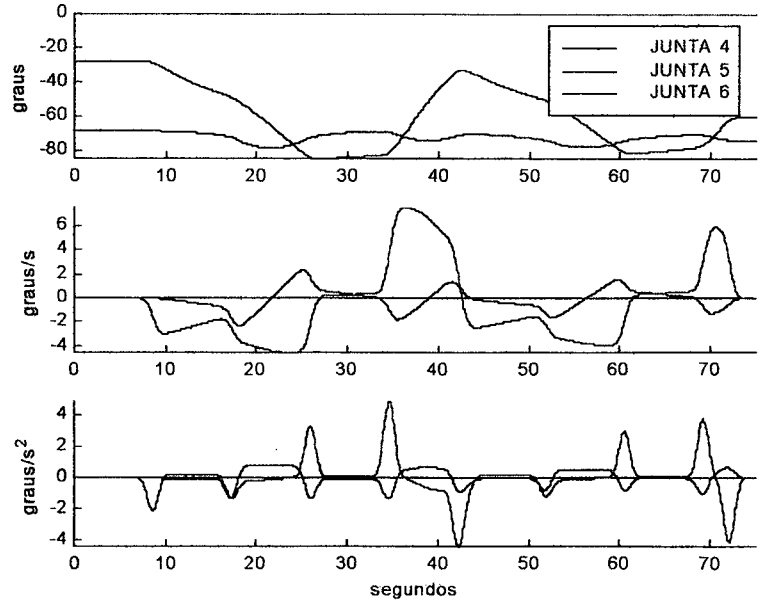

FIGURA 4.22 : Visualização das trajetórias das juntas de um manipulador tipo Puma 560 para uma tarefa como a da figura 4.19.

As posições e as velocidades das juntas do manipulador ficam assim determinadas. Os resultados podem então ser analisados com respeito às diferentes restrições do modelo de manipulador selecionado. A partir do gráfico de posições de juntas é verificado se as faixas de juntas são permissíveis. Com o gráfico de velocidades de juntas é conferido se estas são admissíveis. Ainda, colisões entre o manipulador e a superfície são visualmente verificadas através da animação gráfica das configurações da cadeia cinemática do manipulador durante o seguimento da trajetória. Algoritmos de testes de colisão não foram implementados.

A animação das configurações para execução da tarefa para o modelo de manipulador selecionado pode ser visualizada, como na figura 4.23, inclusive em diferentes vistas figura 4.24.

Além disso, vale citar que diferentes modelos de manipuladores podem ser simulados, assim como diferentes posições da superfície com respeito ao manipulador, possibilitando a escolha do manipulador que melhor se adapta a tarefa e o melhor arranjo para o posicionamento da superfície.

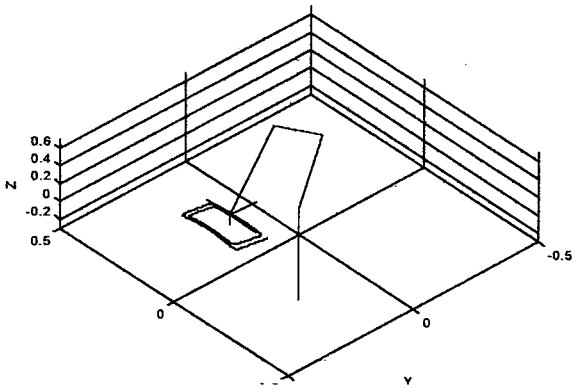

FIGURA 4.23 : Simulação da trajetória no espaço cartesiano 3D e das configurações da estrutura do manipulador.

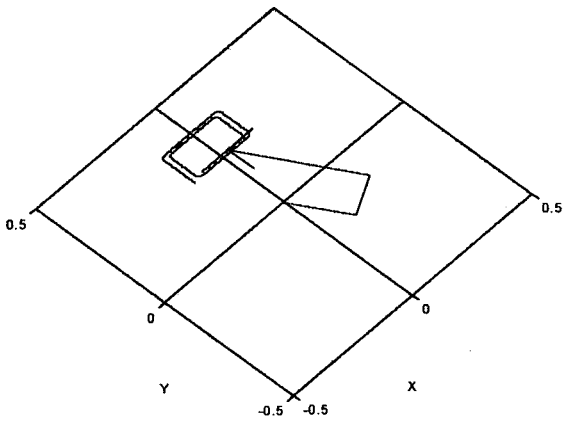

FIGURA 4.24 : Visualização de diferentes vistas da simulação 3D.

Uma vez constatada que as trajetórias geradas podem ser cumpridas pelo manipulador, os dados são manipulados para a transcrição em linguagem de robô.

4.6 A TRANSCRIÇÃO EM LINGUAGEM DE ROBÔ

Como visto no capítulo 2, a utilização de linguagens de programação em robótica está ligada com o fornecimento dos dados necessários para descrever o ciclo de trabalho do manipulador. Estas incluem instruções para mover o manipulador, realizar a leitura de sensores, enviar sinais de saída e muitas outras instruções para programação do controlador.

Nesta pesquisa, apenas os dados referentes à movimentação do manipulador são usados para geração do arquivo de instruções ao robô. Tanto os dados descritivos das trajetórias no espaço cartesiano, quanto no espaço de juntas, podem ser transcritos em comandos para execução da tarefa, dependendo da linguagem de programação.

Assim como nos outros sistemas de programação gráfica dos robôs industriais, já vistos no capítulo 3, os dados de posição, de velocidade e de aceleração devem ser transcritos e tornar-se disponíveis ao software com a linguagem de programação do controlador específico do manipulador que se deseja executar a tarefa. A casos em que o programa de planejamento é o mesmo da programação.

Basicamente, os dados para a movimentação são colocados em forma de uma matriz retangular e devem se tornar disponíveis ao controlador do robô. Duas opções podem ser tomadas neste momento: na primeira opção, trancreve-se todos os dados num arquivo já com os comandos correspondentes, gerando assim um arquivo pronto para rodar no controlador do robô, bastando carregá-lo na memória deste (Teste 5). Na segunda opção, a matriz de dados de movimentação é disponibilizada ao hardware e software do controlador, para que seja executado um programa de busca e execução de comandos com esta matriz (Teste 6).

CAPÍTULO 5

ANÁLISE E RESULTADOS

5.1 INTRODUÇÃO

Este capítulo apresenta a análise e os resultados obtidos com a implementação de uma interface para geração de trajetórias. A metodologia proposta é integrada na interface construída usando a linguagem de programação MatLab [22] e as suas rotinas para geração de diálogos [60]. Além das rotinas padrão para processamento e visualização de dados, são utilizados os pacotes de Robótica [09] e "B-Splines" [07] para efetuar as simulações.

São incluídos, ainda, alguns testes para exemplificar o uso do programa, a demonstração de resultados relacionados à metodologia para a geração de trajetórias, e o estudo de caso para a integração de dois diferentes manipuladores industriais.

5.2 UTILIZAÇÃO DO PROGRAMA

Neste item são apresentados os procedimentos para a utilização das rotinas do programa. Serão descritos procedimentos básicos como os de instalação e de inicialização do programa, e também os procedimentos de uso da interface gráfica interativa desenvolvida.

Com as funções já implementadas e integradas em MatLab, pode-se cadastrar modelos de robôs, cadastrar modelos de superfícies, especificar e criar diferentes trajetórias sobre a superfície para simulação cinemática e dinâmica do manipulador, visualizar os dados para a análise da execução das tarefas, como também transcreve-los para diferentes robôs industriais.

A seguir são descritos os procedimentos de instalação e inicialização, para em seguida detalhar o cadastramento de robôs e superfícies, e as fases de interação do usuário com a interface desenvolvida.

5.2.1 INSTALAÇÃO E INICIALIZAÇÃO DO PROGRAMA

A instalação do programa PathBuilder[1] é realizada em dois passos. No primeiro passo o usuário ou administrador de sistema deve copiar, via disquete ou rede, a árvore de diretórios do programa e do pacote de robótica para um diretório qualquer do computador que se deseja usar. (Figura 5.1) O Programa ocupa relativamente pouco espaço (Figura 5.2)

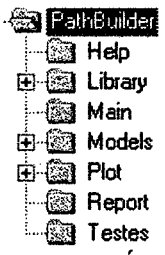

FIGURA 5.1: Árvore de diretórios do Programa.

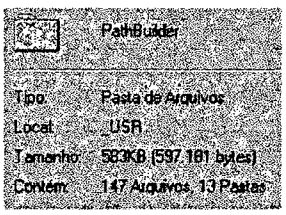

FIGURA 5.2: Espaço em disco ocupado pelo programa.

Para usar o programa é necessário que o software MatLab versão 5.1 com o pacote de "B-Splines" esteja instalado no computador, ou compartilhado via rede.

[1] Path Builder: tradução de Construtor de Percursos para o Inglês.

Feito isto inicia-se o segundo passo, que consiste em fornecer o caminho ("Path") dos diretórios onde estão as rotinas do programa. Para tal deve se acionar o comando >**Set Path** no menu >**File** da {**Command Window**} do MatLab, onde então a janela ilustrada na Figura 5.3 é aberta e os caminhos podem ser fornecidos.

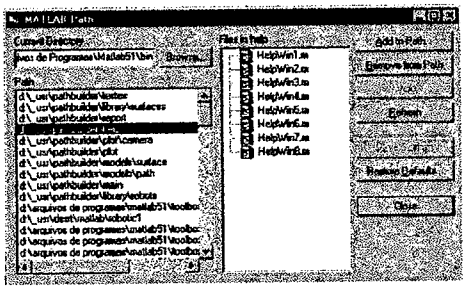

FIGURA 5.3: Interface de comando do MatLab para fornecer os caminhos dos diretórios do programa.

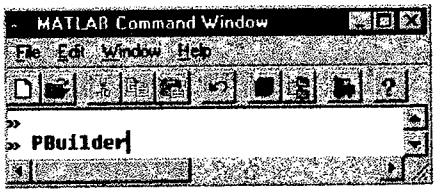

FIGURA 5.4: Comando de inicialização a partir do MatLab da interface gráfica desenvolvida.

Uma vez realizado os dois primeiros passos, para inicializar o programa é necessário, então, chamá-lo a partir do prompt do MatLab através do comando **>>PBuilder**, abreviação de Path Builder, como ilustrado na Figura 5.4.

Com isso, a interface gráfica é iniciada e torna-se disponível para o uso das rotinas para aquisição dos dados dos arquivos cadastrados de modelos de superfícies e de robôs. O cadastramento e edição de modelos de superfícies e robôs é realizado a partir do editor de texto do próprio MatLab, num formato específico para utilização no programa, como será descrito a seguir.

5.2.2 CADASTRAMENTO DE UM ROBÔ

O cadastramento de um robô para a utilização no programa é realizado da mesma maneira que a utilizada nos algoritmos implementados por CORKE [09] em MatLab. Nos algoritmos utilizados no toolbox, a modelagem geométrica de qualquer manipulador de cadeia cinemática simples e aberta, com até seis graus de liberdade, pode ser realizada através da utilização de uma "matriz de descrição" (***dh***) onde constam os parâmetros da convenção de Denavit-Hartenberg [03] e os tipos de juntas de cada elo, conforme mostrado na Tabela 5.1.

COLUNAS	*SÍMBOLOS*	*DESCRIÇÃO*
1	αi	ângulo de torção do elo
2	Ai	comprimento do elo
3	θi	ângulo de rotação do elo
4	Di	distância de compensação do elo
5	σi	opção do tipo de junta: 0 para revolução; 1 para prismática

TABELA 5.1 : Designação das colunas das matrizes dh do toolbox.

Para um manipulador com ***n*** juntas a matriz de descrição ***dh*** tem dimensão **n×5**.

A figura 5.5, a seguir, ilustra o formato que o arquivo para aquisição de dados do modelo geométrico de um manipulador deve possuir. A matriz de descrição deve ser atribuída a uma variável local de retorno de uma função, cujo nome de chamada deve ser o mesmo que o do arquivo, só que este último acrescido da extensão .m, que é o formato de arquivo para MatLab. Nesta figura temos um exemplo de um manipulador Puma 560 [09], que acompanha o pacote de robótica. Os dados são armazenados num arquivo puma560.m, e podem ser editados no editor de texto do MatLab. Novos modelos podem ser criados clicando no ícone de arquivo novo ou no menu

>FILE>NEW , e ainda, impressos, salvos em disquete e excluídos. Informações e exemplos adicionais sobre a modelagem e utilização da matriz de descrição de modelos de manipuladores podem ser encontrados na referência [09].

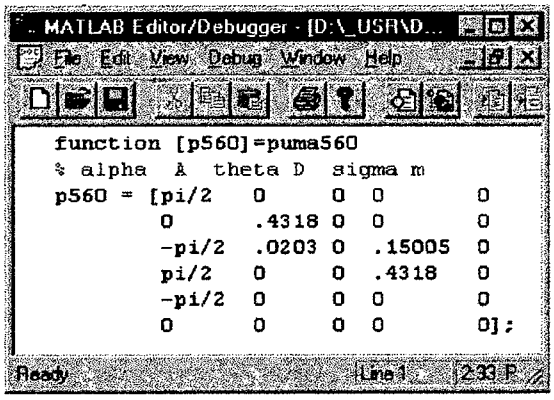

FIGURA 5.5 : Interface para o cadastramento e edição de modelos de manipuladores.

5.2.3 CADASTRAMENTO DE UMA SUPERFÍCIE

Assim como no procedimento de cadastramento de um robô, o procedimento de cadastramento de uma superfície é realizado a partir de um arquivo editado também no editor de texto do MatLab. Contudo, como visto no capítulo anterior, a quantidade de variáveis de dados a serem adquiridas é maior. Para tal, convencionou-se uma seqüência com um número máximo de oito variáveis a serem retornadas e, o nome da função de chamada deve ser o mesmo que o nome do arquivo.

Para exemplificar, a figura 5.6, ilustra um arquivo de modelo de superfície do tipo Bézier [17]. A chamada deste arquivo pela interface do programa, como será descrito na seção 5.2.4, permite então carregar a quantidade de variáveis que devem ser adquiridas ***n***, o tipo de entidade a

ser manipulada pela interface ***entity***, as matrizes de pontos referentes à superfície ***Xnet, Ynet*** e ***Znet*** (Apêndice 2), e variáveis nulas ***null5, ..., null8*** que não são utilizadas para este tipo de superfície, mas que podem ser eventualmente utilizadas para outros modelos de superfícies.

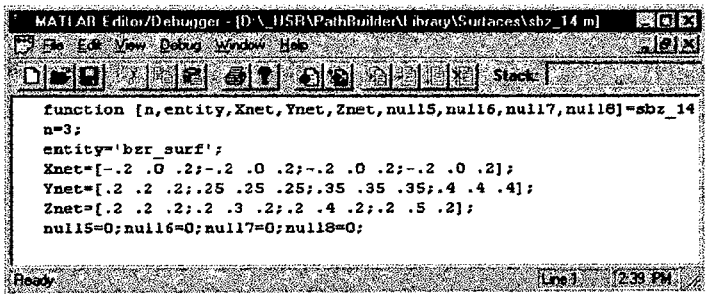

FIGURA 5.6 : Interface para o cadastramento e edição de modelos de superfícies.

5.2.4 ENTRADA DE DADOS DA SIMULAÇÃO

Ao iniciar o programa, como descrito na seção 5.2.1, a interface principal para entrada de dados da simulação é exibida como ilustrado na Figura 5.7.

A interface de entrada de dados da simulação permite escolher um modelo de superfície para ser trabalhada ❶, selecionar um modelo de robô para execução da tarefa ❷, especificar ou criar uma trajetória cartesiana sob a superfície a ser percorrida pelo efetuador do modelo de robô ❸, especificar uma orientação constante do efetuador ❹, iniciar os cálculos da simulação ❺, finalizar a simulação ou obter informações a respeito do programa e o seu uso ❻ e, finalmente, limpar os dados especificados e selecionados para iniciar uma nova simulação ❼. Como descrito nos itens 5.2.2 e 5.2.3, os procedimentos de cadastramento e edição de modelos de robôs e de superfícies é realizado a partir do editor de texto do MatLab.

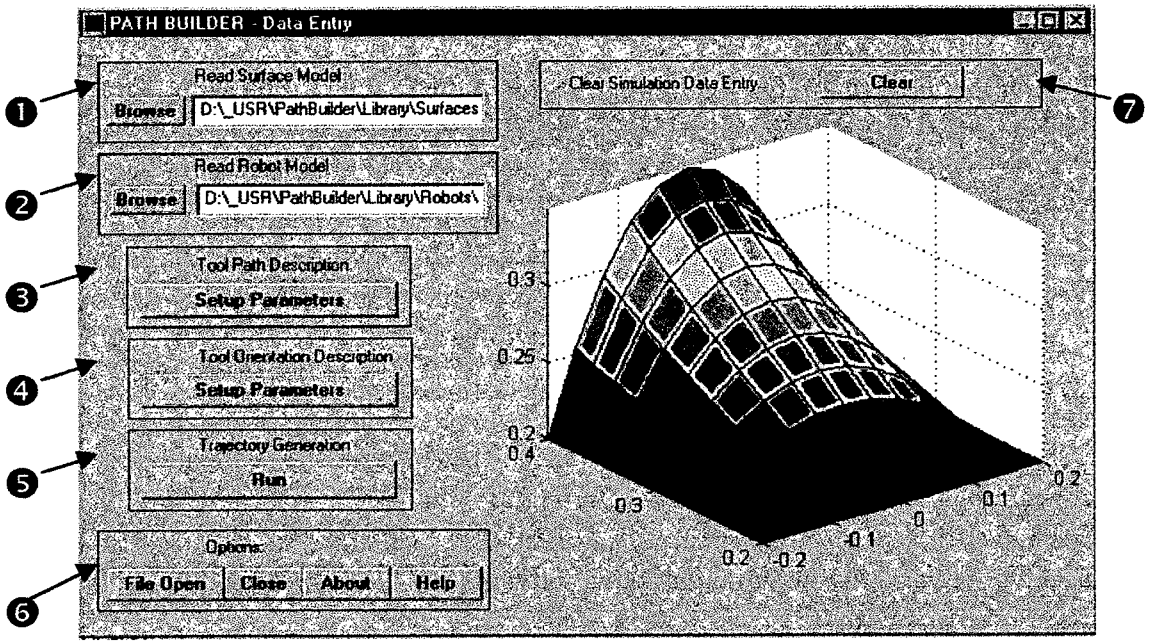

FIGURA 5.7: Interface Gráfica em MatLab para a entrada de dados da simulação.

A seguir temos uma descrição das etapas de entrada de dados da simulação, com a descrição mais detalhada das funções presentes na interface.

5.2.4.1 LEITURA DO MODELO DE SUPERFÍCIE

Apontando o botão ❶ **[BROWSE]**, exibido na figura 5.4, aciona-se uma janela que consiste de um gerenciador de arquivos, como o ilustrado na figura 5.8, a partir do qual o usuário pode caminhar na estrutura de diretórios do computador, possibilitando a seleção do arquivo com os dados de descrição do modelo de superfície que deseja-se adquirir os dados. O gerenciador tem ainda outras funções, como: excluir, renomear, copiar e compactar arquivos.

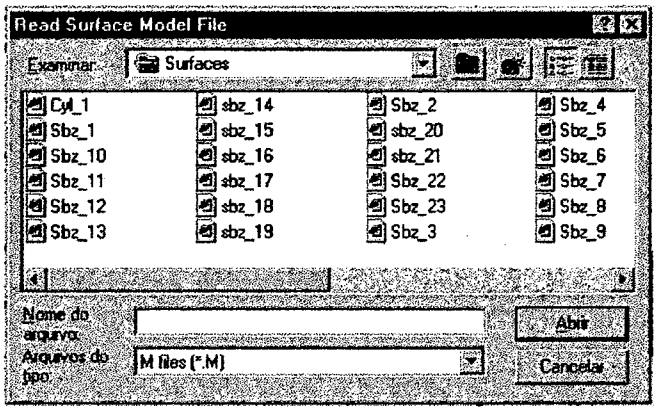

FIGURA 5.8: Interface de seleção do arquivo de dados do modelo de superfície a ser utilizado na simulação.

5.2.4.2 LEITURA DO MODELO DE ROBÔ

A leitura do modelo de robô é realizada da mesma maneira que a leitura de modelo de superfície. A partir de um arquivo em MatLab são adquiridos os dados relevantes à geometria do modelo de superfície que se deseja manipular.

Apontando o botão ❷ [**BROWSE**], exibido na figura 5.4, também é aberta uma janela que consiste em um gerenciador de arquivos, como exibido na figura 5.9, no qual o usuário tem a função principal de fornecer o caminho do arquivo na estrutura de diretórios do computador e o nome do arquivo que contém o modelo de robô. Outras funções mais também podem ser acessadas no gerenciador de arquivos, como visto na seção anterior.

Uma vez selecionados os modelos de superfície e de robô que serão utilizados na simulação, o usuário deve especificar, conforme será visto nas duas próximas seções, o percurso a ser realizado sob a superfície, bem como a orientação que o efetuador deve tomar durante o percorrimento do percurso.

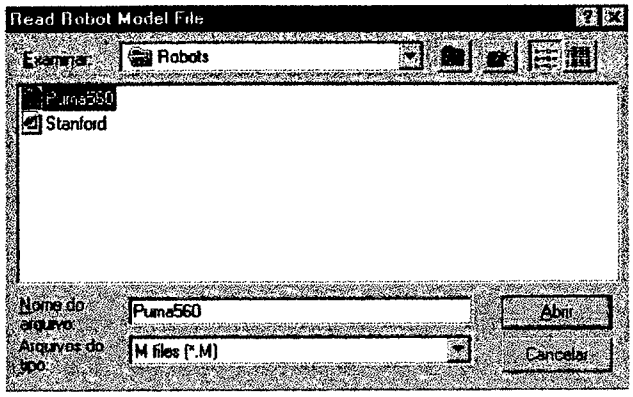

FIGURA 5.9: Interface de seleção do arquivo de dados do modelo de manipulador para execução da tarefa.

5.2.4.3 ENTRADA DOS PARÂMETROS DO PERCURSO DO EFETUADOR

Como visto no capítulo 4, a metodologia proposta faz uso de um percurso de pontos nos parâmetros de uma superfície paramétrica.

Apontando o botão ❸ **Tool Path Description / [SETUP PARAMETERS]**, exibido na figura 5.7, é aberta uma janela com um menu como o ilustrado na figura 5.10, onde o usuário tem então quatro opções de tipos de percursos ❶ à ❹ para exemplificar a metodologia proposta, a opção de cancelar a janela ❺ ou ainda, obter informações com respeito a janela em utilização❻ .

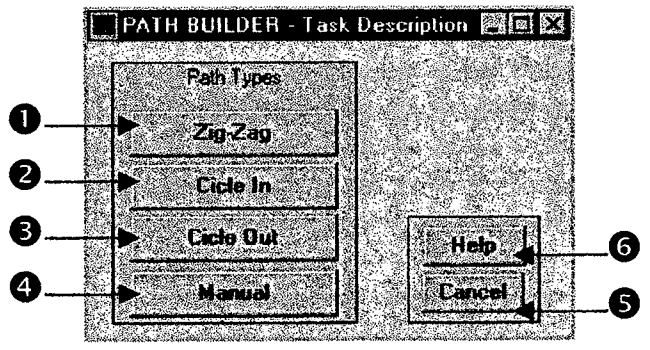

FIGURA 5.10: Janela com o menu de opções para especificação ou criação de percursos.

Apontando o botão ❶ [ZIGZAG], exibido na figura 5.10, é aberta uma janela {**Zig–Zag Setup**} com uma interface de entrada de dados, como a ilustrada na figura 5.11 a seguir, na qual o usuário deve editar os parâmetros da tarefa, dados pelo número de passos do percurso em zig-zag , o tempo de execução da tarefa e os afastamentos relativos às bordas da superfície biparamétrica.

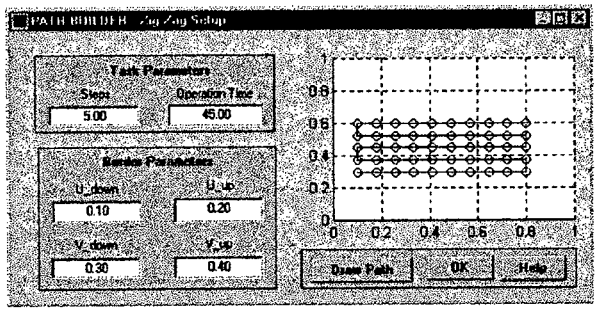

FIGURA 5.11: Interface da entrada de dados para especificação de um percurso em zigzag.

No exemplo ilustrado na figura 5.11, o usuário editou uma especificação de percurso em zig-zag com cinco passos, para ser executado em 45 segundos, e afastamentos variados com respeito às bordas da superfície. Uma vez editados pelo usuário os dados relativos ao percurso da tarefa que se deseja executar, o botão [**DRAWPATH**] deve ser acionado para que se tenha a visualização do percurso, para em seguida realizar a confirmação [**OK**] da especificação ou cancela-la [**X**]. O usuário pode ainda, como em todas as janelas do programa, obter ajuda com respeito aos comandos da janela [**HELP**].

Outro dois exemplos para especificação de percursos foram implementados. Estes são acessados apontando os botões ❷ e ❸ da figura 5.10 para especificação de percurso em ciclo de fora para dentro [**CICLE IN**] e de dentro para fora [**CICLE OUT**], respectivamente. Ambos possuem as mesmas opções da interface para especificação do percurso em zig-zag, diferindo

apenas na forma e em dois parâmetros a mais que devem ser especificados para determinar a largura e altura do núcleo do percurso.

A figura 5.12, a seguir, ilustra um exemplo do percurso em ciclo de fora para dentro. Neste exemplo o usuário especificou um percurso de quatro passos, a ser executado em 60 segundos, com afastamentos iguais relativos às bordas da superfície e de valor 0.1, onde o núcleo do ciclo tem uma largura de 0.3 e uma altura de 0.2.

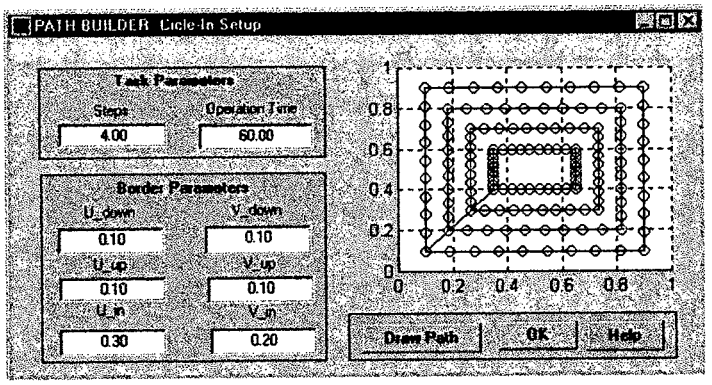

FIGURA 5.12: Interface de entrada de dados para a especificação dos percursos em ciclo.

Finalmente, a quarta e última opção para especificação de percurso sob a superfície, que é acessada apontando o botão ❹ **[MANUAL]** ilustrado na Figura 5.11, consiste de uma interface para determinação de um percurso livre nos parâmetros da superfície, com uma entrada manual via mouse, e para a edição do tempo total para execução da tarefa. A figura 5.13 ilustra esta interface em que um usuário especificou um tempo de 120 segundos para execução de uma tarefa com 24 pontos.

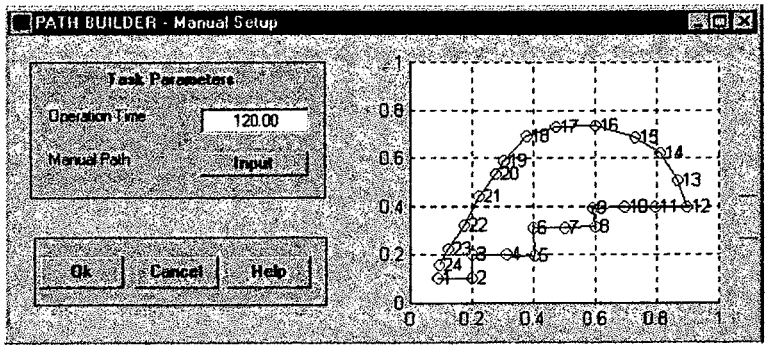

FIGURA 5.13: Interface de entrada de dados para a especificação de um percurso livre.

Vale observar que nesta seção fizemos uma explicação desta etapa da entrada de dados da simulação; entretanto na publicação LIN[32], possui uma explicação mais detalhada dos parâmetros para especificação dos percursos em zig-zag, ciclo para dentro, ciclo para fora e manual.

5.2.4.4 ENTRADA DOS PARÂMETROS DA ORIENTAÇÃO DO EFETUADOR

Apontando o botão ❹ **Tool Orientation Description / [SETUP PARAMETERS]**, exibido na figura 5.7, é aberta uma janela com um menu como o ilustrado na figura 5.14, onde o usuário tem então a possibilidade de variar os ângulos "roll", "pitch" e "yaw" [03], através de uma barra de rolagem, de um sistema de referência (colorido) com relação ao sistema XYZ correspondente a posição zero do efetuador do manipulador.

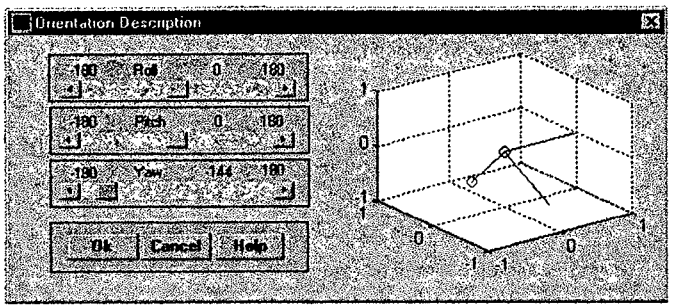

FIGURA 5.14: Interface para a entrada de dados da orientação do efetuador do manipulador.

5.2.4.5 EXECUÇÃO, CANCELAMENTO, REINICIALIZAÇÃO, INFORMAÇÕES E AJUDA

Após realizados os procedimentos de entrada de dados da simulação, dados pelas quatro seções anteriores, o usuário deve apontar o botão ❺ **[RUN]**, ilustrado na Figura 5.7, para iniciar o processamento dos dados de entrada e obter os dados para visualização da saída referente à simulação que foi especificada.

Ainda na Figura 5.7 tem-se as opções do menu de botões ❻ **Options** / **[CLOSE]** **[ABOUT]** **[HELP]**, onde o usuário pode encerrar o programa, obter informações ou ajuda do programa, respectivamente, e ainda a opção dada pelo botão ❼ **[CLEAR]** que permite ao usuário reinicializar a entrada de dados do programa.

5.2.5 VISUALIZAÇÃO DOS DADOS DE SAÍDA DA SIMULAÇÃO

Uma vez apontado o botão **[RUN]**, descrito na seção anterior, é percorrido o tempo de processamento de dados, para em seguida abrir a janela **{Visualization}**, conforme ilustrada na Figura 5.15, para a visualização dos dados obtidos da simulação.

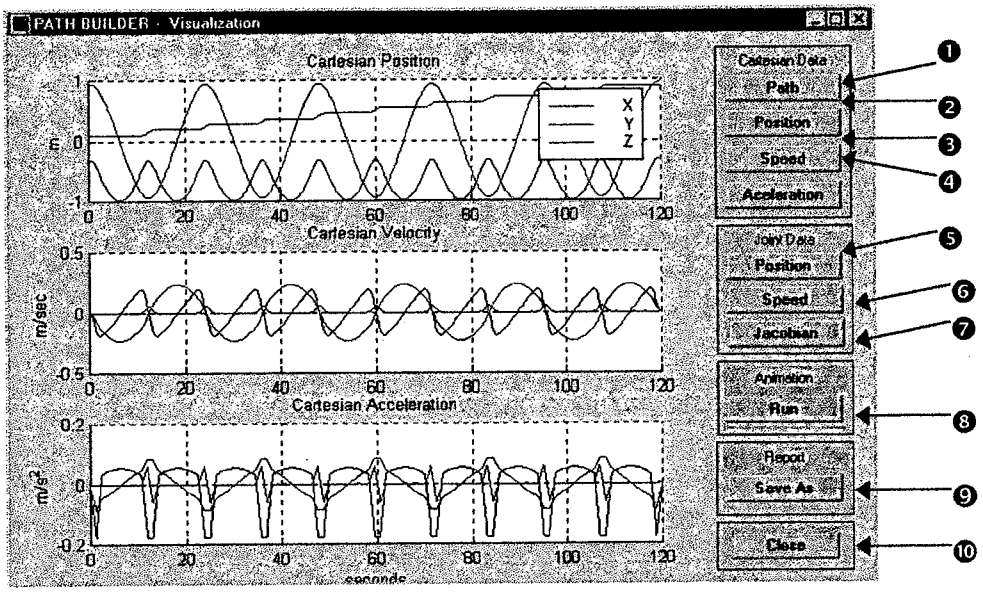

FIGURA 5.15: Interface de visualização dos dados de saída da simulação.

Os dois primeiros menus de botões **Cartesian Data / Joint Data** permitem, respectivamente, visualizar as trajetórias cartesianas e as trajetórias de juntas para o cumprimento da tarefa especificada. Já os menus **Animation / Report** permitem visualizar uma animação do modelo do manipulador cumprindo uma tarefa e armazenar em arquivo um relatório com os dados calculados, respectivamente. A seguir temos uma descrição das ações correspondentes ao acionamento de cada botão da interface de visualização de dados da simulação.

5.2.5.1 VISUALIZAÇÃO DA ANIMAÇÃO DA TRAJETÓRIA 3D

Apontando o botão ❶ **[PATH]**, ilustrado na Figura 5.15, é exibida uma animação gráfica 3D da trajetória cartesiana especificada para simulação, como ilustrado na Figura 5.16. Com o gráfico 3D exibido com o uso das rotinas de exibição do MatLab, pode-se através dos menus **Roll / Orbit**

rotacionar a vista, via mouse, para visualização de diferentes pontos do espaço cartesiano. Outros comandos de interação com o usuário também estão disponíveis, como o recurso de aproximação da vista do gráfico [Zoom] e a repetição da animação [Replay].

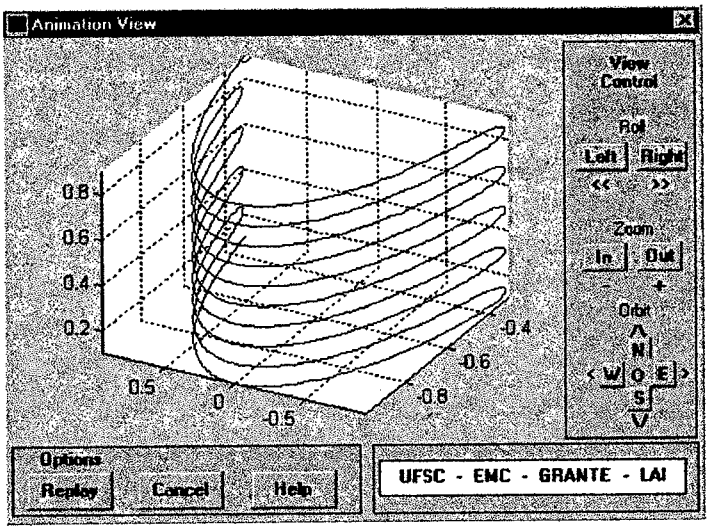

FIGURA 5.16: Visualização da trajetória 3D.

5.2.5.2 DADOS DA TRAJETÓRIA NO ESPAÇO CARTESIANO

Apontando os botões ❷ **[POSITION]** ❸ **[VELOCITY]** ❹ **[ACCELERATION]** do menu **Cartesian Data**, ilustrados na Figura 5.15, são exibidos os gráficos de posições no tempo, velocidades no tempo e acelerações no tempo, calculados segundo as equações 4.1, 4.2 e 4.3, e ilustrados nas figuras 5.17, 5.18 e 5.19, respectivamente. Assim como em todo gráfico 2D exibido com o uso das rotinas de exibição do MatLab, o usuário pode aproximar uma faixa do gráfico, via mouse, para melhor a visualização dos dados, como ilustrado na figura 5.20.

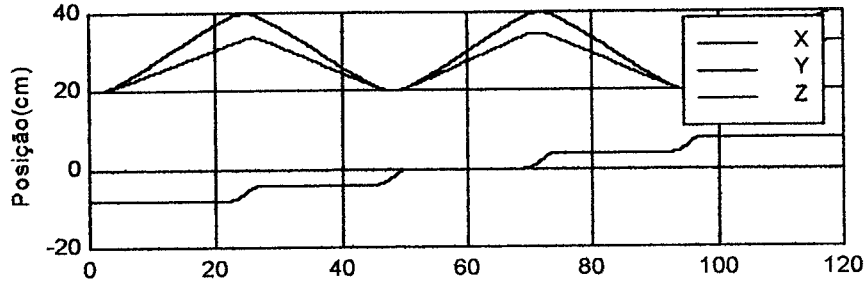

FIGURA 5.17: Visualização das posições cartesianas no tempo.

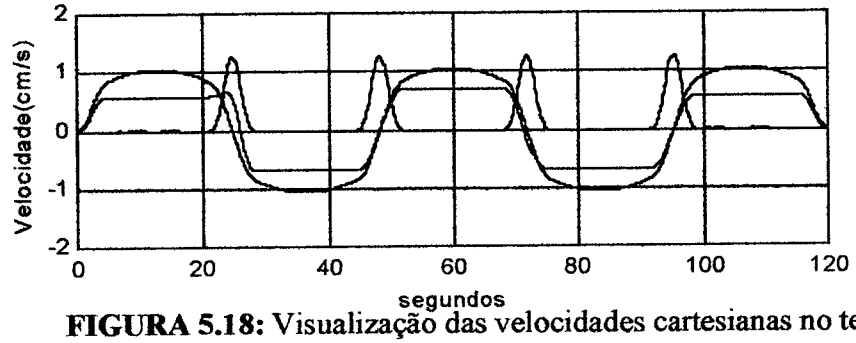

FIGURA 5.18: Visualização das velocidades cartesianas no tempo.

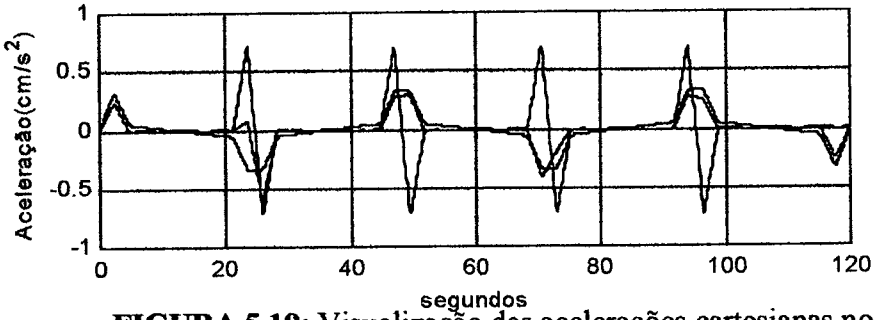

FIGURA 5.19: Visualização das acelerações cartesianas no tempo.

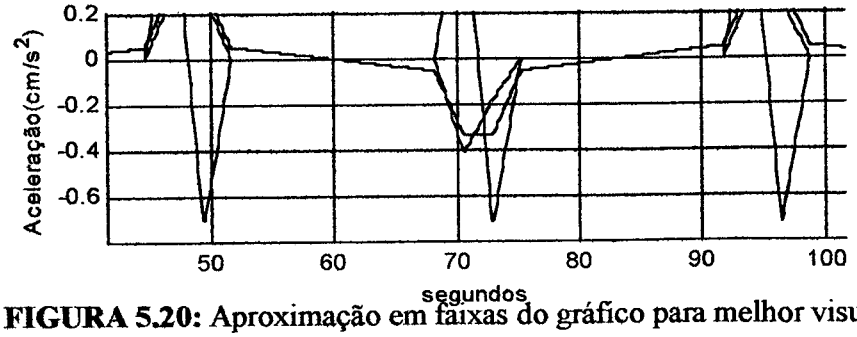

FIGURA 5.20: Aproximação em faixas do gráfico para melhor visualização.

5.2.5.3 DADOS DA TRAJETÓRIA NO ESPAÇO DE JUNTAS

Apontando os botões ❺ [**POSITION**] ❻ [**VELOCITY**] do menu **Joint Data**, ilustrado na Figura 5.15, são exibidos os gráficos de posições no tempo e velocidades no tempo, calculados com o das rotinas de Robótica [16], e conforme ilustrados nas figuras 5.21 e 5.22, respectivamente. Novamente o usuário pode aproximar uma faixa do gráfico, via mouse, para melhor visualização dos dados, como na seção anterior.

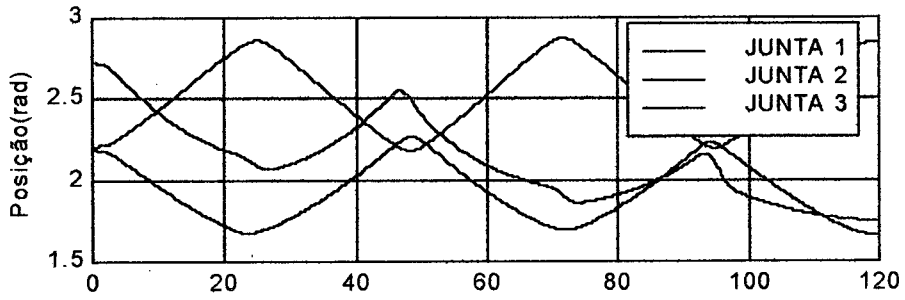

FIGURA 5.21: Visualização das posições das juntas no tempo.

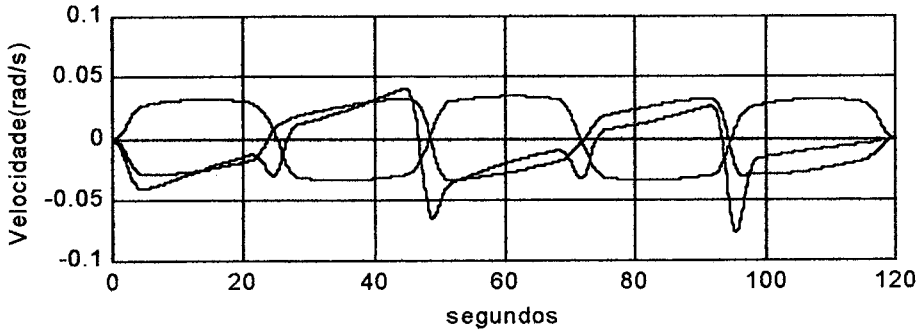

FIGURA 5.22: Visualização das velocidades das juntas no tempo.

5.2.5.4 ANIMAÇÃO DA TAREFA

Apontando o botão ❼ **[RUN]** do menu **Animation**, ilustrado na Figura 5.15, é aberta um janela, ilustrada na Figura 5.23, para exibição de uma animação das configurações do manipulador no cumprimento das posições da trajetória especificada, calculadas com o uso das rotinas do pacote de Robótica [09]. O usuário pode ainda aproveitar as rotinas de rotação e aproximação da vista, de modo a melhor visualizar a animação das configurações, interações estas ilustradas nas figuras 5.24 e 5.25.

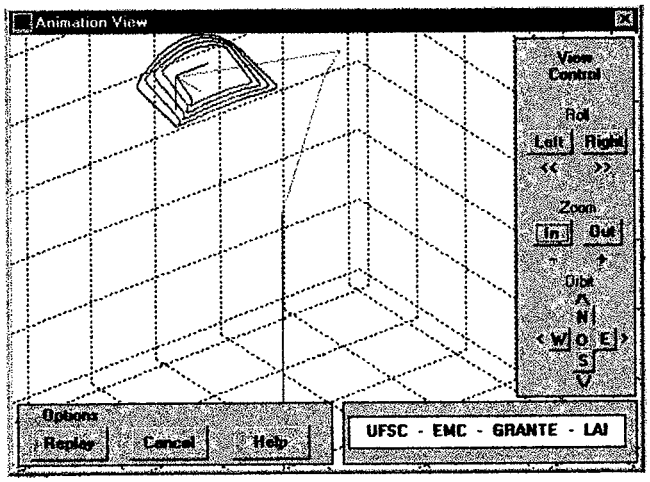

FIGURA 5.23: Janela de visualização da animação das configurações do manipulador durante a execução do movimento especificado.

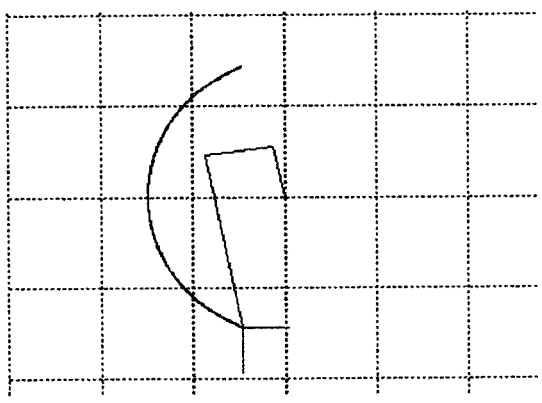

FIGURA 5.24: Rotação da vista da animação (Vista Superior).

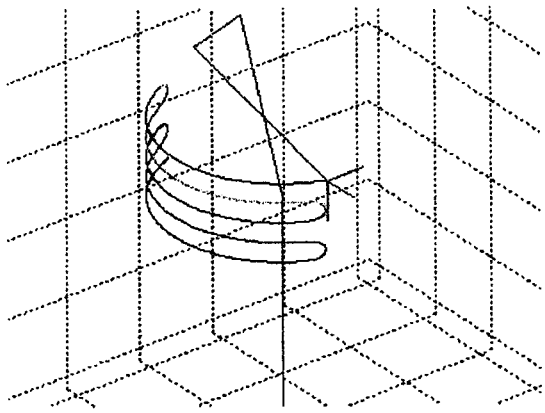

FIGURA 5.25: Aproximação da vista da animação.

5.2.5.5 RELATÓRIO DOS DADOS DA SIMULAÇÃO EM ARQUIVO

Apontando o botão ❽ [**SAVE AS**] do menu **REPORT**, ilustrado na Figura 5.15, é aberta um janela que consiste em um gerenciador de arquivos, como ilustrado na Figura 5.26. Este permite ao usuário salvar os dados calculados num arquivo com formato de tabela de dados, conforme ilustrado na Figura 5.27, no diretório indicado da árvore do computador e com extensões **.m**, **.dat**, **.txt**, ou mesmo, **.doc**.

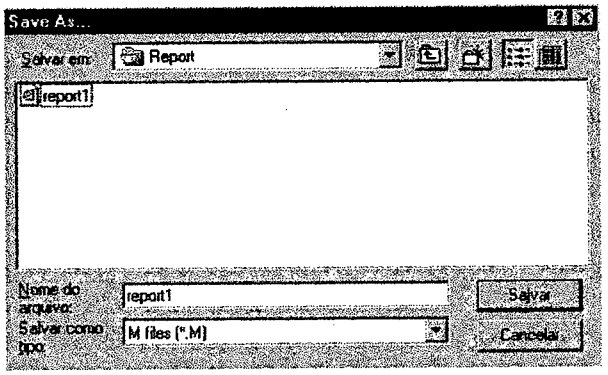

FIGURA 5.26: Interface para salvar os dados em arquivo.

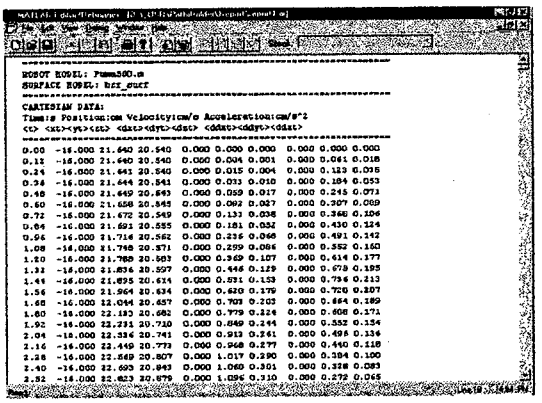

FIGURA 5.27: Interfaces do editor de texto do MatLab para visualização dos arquivos com os dados gerados.

5.3 RESULTADOS

Esta seção apresenta uma série de seis testes que visam permitir a análise, a verificação e a validação da metodologia proposta. Os quatro primeiros testes buscam analisar o comportamento dos dados gerados quanto à sua qualidade, grau de controle, precisão e flexibilidade. Os dois outros testes visam demonstrar a possibilidade de interface dos dados gerados pelo método proposto para a programação de dois diferentes manipuladores.

A seguir temos a lista dos testes que foram realizados:

- Teste 1 – Verificação da Metodologia Utilizada para Geração de Curvas.
- Teste 2 – Comparativo do Controle da Trajetória através da Seleção dos Pontos.
- Teste 3 – Comparativo do Controle da Trajetória através do Vetor de Referência.
- Teste 4 – Comparativo do Controle da Trajetória através da Ordem da Curva.
- Teste 5 - Programação no Espaço Cartesiano para o Controle de Posição de um Robô Articulado[59].
- Teste 6 - Simulação Cinemática e Programação no Espaço de Juntas para o Controle de Posição e Velocidade de um Robô Tipo Scara.

5.3.1 GERAÇÃO DE TRAJETÓRIA SOBRE UMA SUPERFÍCIE CILÍNDRICA

O intuito deste primeiro teste é analisar o método apresentado para geração de trajetórias cartesianas, detalhado no capítulo IV, de modo a verificar o seu comportamento.

Para tal considere uma hélice circular, conforme a equação dada a seguir

$$\begin{aligned} x(t) &= cos(t) \\ y(t) &= sen(t) \\ z(t) &= t / 2pi \\ t &= 0 : 2pi. \end{aligned} \qquad (5.1)$$

Realizaremos uma aproximação das coordenadas da hélice circular através do uso de uma superfície cilíndrica com um raio de 1 m ao longo do eixo z positivo e centro de massa em (0,0,0.5), como ilustrada na Figura 5.28, e do uso de um percurso nos parâmetros da superfície cilíndrica, como ilustrado na Figura 5.29.

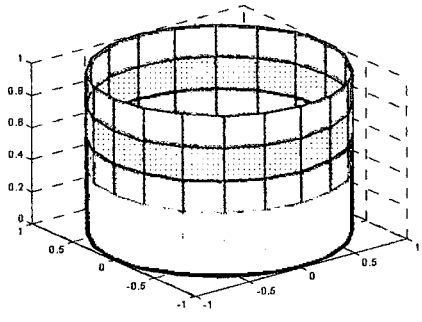

FIGURA 5.28 : Superfície cilíndrica.

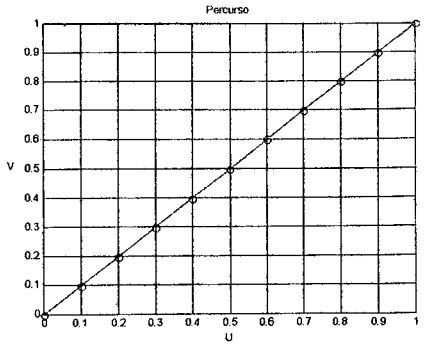

FIGURA 5.29 : Percurso nos parâmetros da superfície.

O teste consiste em realizar uma variação no número de pontos ao longo dos parâmetros da superfície (10, 30, 50, 80, 100, 150), para então comparar os resultados das aproximações com o uso da formulação de curvas "B-Splines" [07] com multiplicidade k=2 nos extremos do percurso de pontos, e ordem 4, com os obtidos pelas equações (5.1), possibilitando realizar uma análise de comportamento das curvas e das derivadas primeira e segunda.

As figuras 5.30 à 5.36 apresentam os resultados obtidos para o teste proposto.

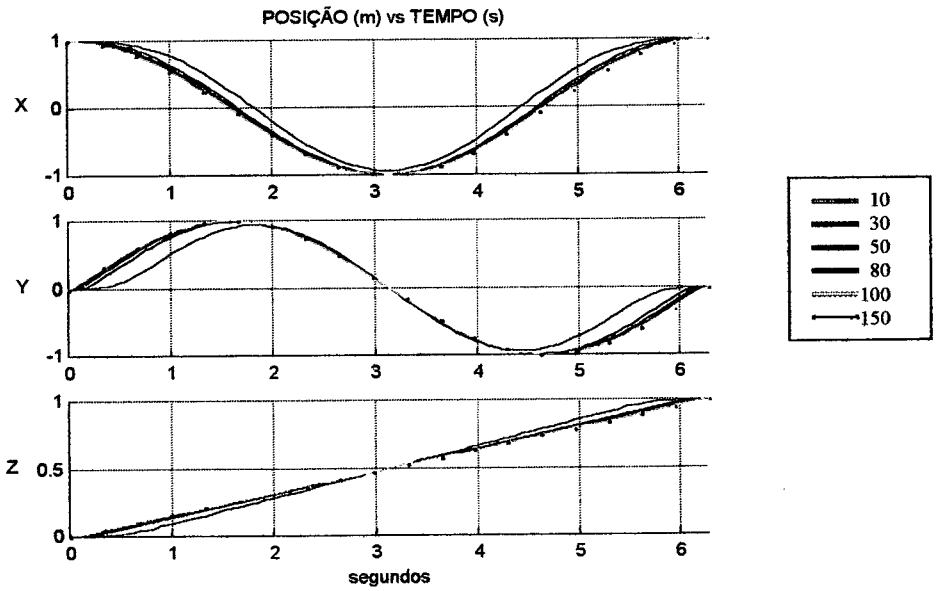

FIGURA 5.30: Comparativo das diferentes curvas de posição de acordo com o número de pontos no percurso nos parâmetros da superfície cilíndrica.

Inicialmente, podemos verificar nos gráficos das posições cartesianas no tempo, figura 5.30, que o aumento do número de pontos do percurso nos parâmetros da superfície cilíndrica diminui acentuadamente o desvio entre as coordenadas de posição no tempo obtidas pelas curvas "B-Splines" (Coloridas) e as coordenadas obtidas pela equação da hélice circular (HC). Observando as curvas com dez pontos, também podemos notar um desvio da posição no tempo maior nos trechos

inicial e final da coordenada Y, ocasionado pela maior tensão inserida na curva, através da multiplicidade k=2, para que a derivada primeira e segunda sejam nulas.

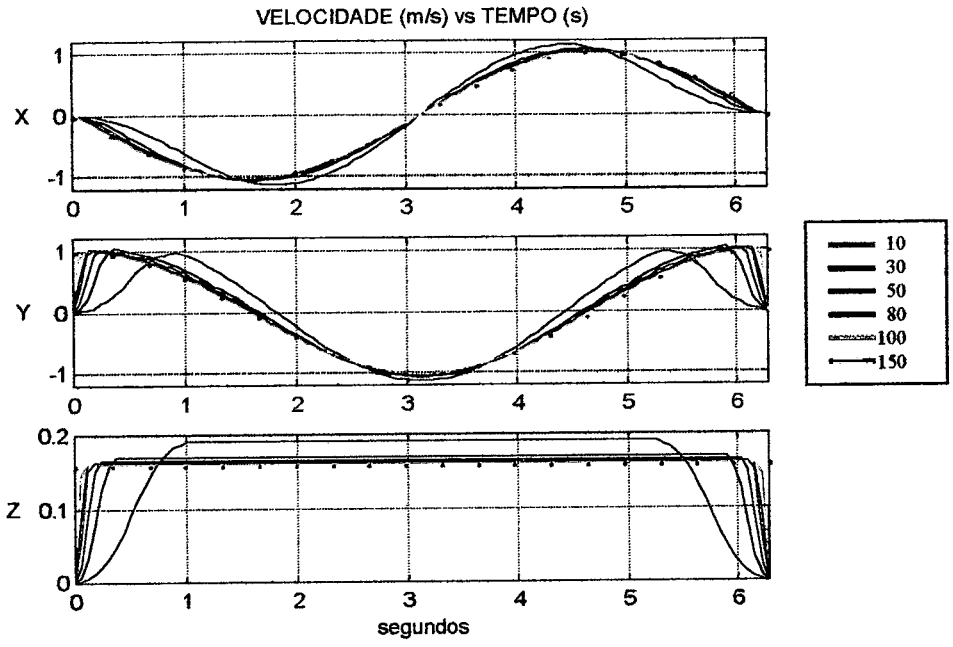

FIGURA 5.31: Comparativo entre as diferentes curvas de velocidade de acordo com o número de pontos do percurso nos parâmetros da superfície cilíndrica.

Quanto às curvas de velocidade, obtidas através da derivada primeira das curvas "B-Splines", e ilustradas na figura 5.31, podemos fazer as seguintes observações. As curvas de velocidade da coordenada X possuem menor desvio com a variação de pontos do percurso, uma vez que no gráfico de posições no tempo da coordenada X da hélice circular, as derivadas são naturalmente nulas e, assim, tem-se menor tensão ocasionada pela multiplicidade k=2 nos extremos. Observando as coordenadas Y e Z, podemos facilmente notar que a variação do número de pontos do percurso nos parâmetros da superfície ocasiona uma diminuição do intervalo de tempo para satisfação da imposição de coordenadas de velocidade nulas nos extremos.

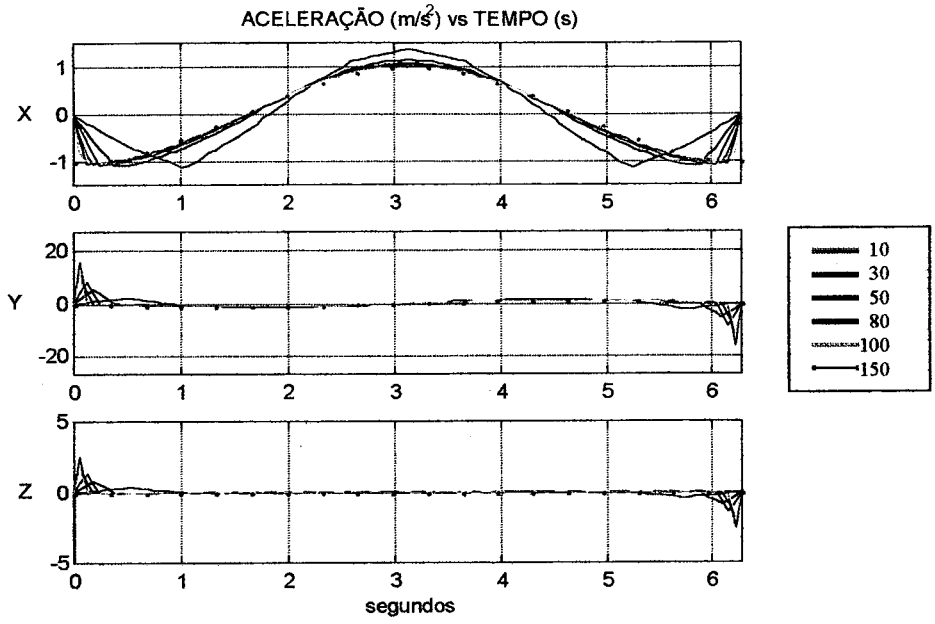

FIGURA 5.32: Comparativo entre as diferentes curvas de aceleração de acordo com o número de pontos do percurso nos parâmetros da superfície cilíndrica.

Os gráficos das curvas de aceleração no tempo, obtidos através da derivada segunda das curvas "B-Splines", e ilustrados na figura 5.32, também nos permite verificar características importantes na formulação de curvas utilizada. Nas curvas de aceleração da coordenada X, podemos observar os mesmos efeitos relatados para as curvas de velocidade das coordenadas Y e Z, ocasionado pela imposição das condições de contorno nula nos extremos. Já nas curvas de aceleração das coordenadas Y e Z, podemos verificar como efeito ocasionado pela variação do número de pontos do percurso, um aumento significativo dos valores de aceleração nos extremos das curvas, devido à diminuição do intervalo de tempo para satisfação das condições de contorno nulas.

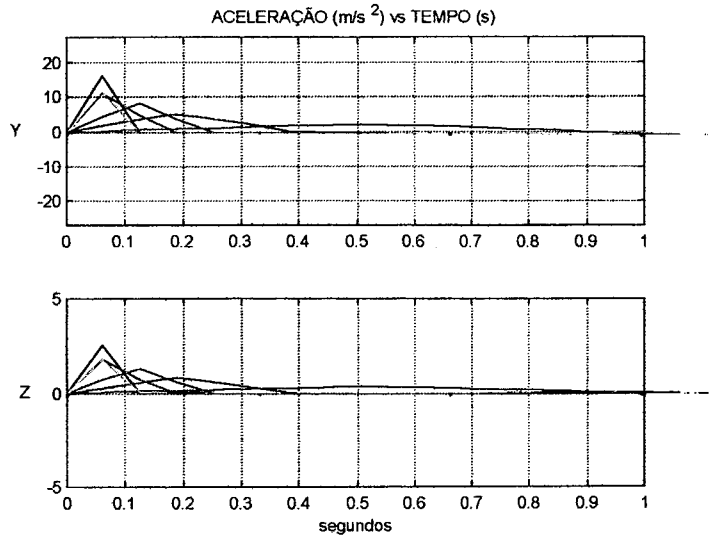

FIGURA 5.33: Aproximação em faixas dos gráficos das coordenadas de aceleração Y e Z da figura 5.32

A figura 5.33 ilustra uma aproximação do trecho inicial das curvas de aceleração nas coordenadas Y e Z. Nesta podemos observar com mais detalhe o aumento significativo dos níveis de aceleração. Independente da coordenada, o percurso de 10 pontos possui aproximadamente 1 segundo para satisfazer as condições de contorno nulas, já o percurso de 150 pontos tem aproximadamente 0.1 segundo para satisfaze-las, como resultado, temos um aumento de aproximadamente 7 vezes no nível de aceleração da coordenada y, e 5 vezes na coordenada z. Apesar dos resultados aparentarem impossibilitar o aumento do número de pontos para obtenção de precisão da metodologia apresentada, este tipo de problema pode ser contornado com a especificação de um tempo maior para a execução da tarefa ou, ainda, através de técnicas de controle da curva "B-Splines" gerada, apresentadas nos próximos testes.

A figura 5.34, a seguir, ilustra as trajetórias cartesianas 3D geradas no teste proposto, e a figura 5.35 ilustra a vista superior desta. A aproximação desta vista, ilustrada na figura 5.36, nos permite realizar uma análise da precisão do método para a geração de trajetórias cartesianas.

Nesta figura são ilustradas as faixas das curvas com 30, 50, 80, 100 e 150 pontos aproximando-se do ponto (X,Y)=(-1,0). Estas foram escolhidas devido ao fato de que, pela figura 5.35, podemos notar ser este ponto o de maior desvio com relação a hélice circular. Vale lembrar que a hélice circular possui 1m de raio e 1 m de altura, o que nos levar a notar que as curvas geradas com 80, 100 e 150 pontos possuem uma precisão de posicionamento com relação ao ponto de comparação (-1,0) de aproximadamente 0.35 mm, 0.65 mm e 1mm, respectivamente.

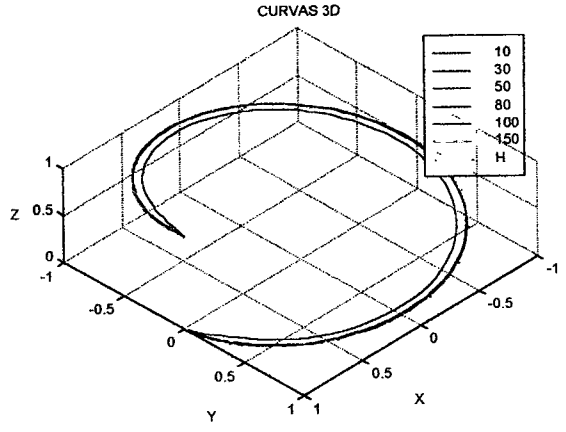

FIGURA 5.34: Comparativo entre as trajetórias cartesianas 3D com diferentes distribuições de pontos dos percursos nos parâmetros da superfície.

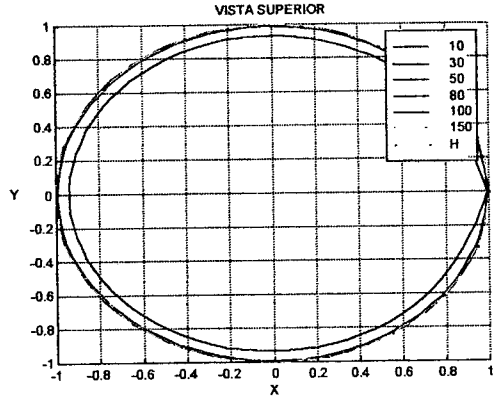

FIGURA 5.35. Vista superior da figura 5.34.

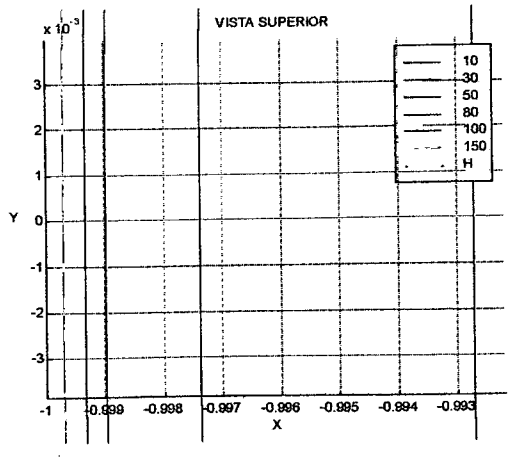

Figura 5.36: Aproximação da figura 5.35.

A tabela 5.2 exibe os tempos de processamento num computador pentium 133 Mhz e 16 Mb Ram para geração das curvas de posição, velocidade e aceleração do teste proposto, de acordo com a variação do número de pontos do polígono de definição e do números de pontos avaliados sob as curvas.

100ptos				
0.1100	0.1100	0.1100	0.3300	10
0.1100	0.1100	0.1100	0.3300	30
0.1200	0.1200	0.1400	0.3800	50
0.1300	0.1300	0.1800	0.4400	80
0.1700	0.1700	0.2100	0.5500	100
0.1800	0.1800	0.2300	0.5900	150
1000ptos				
0.1700	0.1700	0.1600	0.5000	10
0.1800	0.1700	0.1900	0.5500	30
0.1900	0.1800	0.2300	0.6000	50
0.1900	0.2200	0.2300	0.6400	80
0.2100	0.2500	0.2500	0.7100	100
0.2200	0.2500	0.2700	0.7400	150
10000ptos				
1.1000	1.0400	1.0500	3.1900	10
1.1000	1.0900	1.0500	3.2400	30
1.1000	1.0900	1.0500	3.2400	50
1.1500	1.1600	1.0900	3.4000	80
1.1500	1.1700	1.1500	3.4700	100
1.1700	1.1900	1.2100	3.5700	150
Posiç.	Veloc.	Aceler.	Total	

TABELA 5.2: Tempos de processamento em segundos das curvas de posição, velocidade e aceleração para diferentes números de pontos do polígono de definição e diferentes números de pontos avaliados sob a curva gerada.

Nos dados exibidos na tabela, podemos perceber, que tanto o aumento do número de pontos do polígono de definição, como o aumento do número de pontos avaliados sob a curva, aumentam o tempo de processamento das posições, velocidades e acelerações das coordenadas cartesianas. Comparando os tempos gastos no teste proposto com dados obtidos na literatura [45] e,

considerando que para cada processamento estão sendo calculados os dados das coordenadas cartesianas X, Y e Z e que temos o cálculo de uma trajetória cartesiana de precisão formulada através de pontos intermediários e condições de contorno nulas, os tempos de processamento são bastante satisfatórios.

Primeiramente, o que podemos ressaltar nos resultados deste primeiro teste é a facilidade e flexibilidade de uso da metodologia, uma vez que com um modelo de superfície paramétrica qualquer e um percurso de pontos nos parâmetros desta, a ser percorrido num determinado período de tempo especificado, gera-se automaticamente para análise e verificação, numa resposta relativamente rápida, curvas de posição, velocidade e aceleração com condições de contorno nulas, sejam no espaço de juntas, com o uso da cinemática inversa, ou no próprio espaço cartesiano. Em segundo lugar podemos ressaltar que o método permite obter tanto trajetórias de movimentação como de precisão, onde a amostragem das curvas obtidas podem ser variadas de acordo com o tipo de controlador que se utiliza. E por último, vale ressaltar que o método possibilita a programação de diversificadas tarefas de manipuladores, e não uma classe especifica de atividades, ou seja, constitui um método generativo através do uso de conceitos da computação gráfica.

5.3.2 Geração de Trajetória em Zig Zag em Superfície Paramétrica Plana

Este teste visa demonstrar a possibilidade de controle dos níveis de velocidade e aceleração da trajetória cartesiana a ser gerada, através da variação da seleção do percurso de pontos nos parâmetros da superfície.

Considere os percursos em zigzag (1) e (2), como ilustrados nas figuras 5.37 e 5.38. No percurso (1) os trechos inicial, terminal e com mudança de direção, são demarcados com pontos simples. Já no percurso (2), o trecho inicial e o terminal estão acrescidos de um ponto e, os trechos com mudança de direção do percurso tem dois pontos a mais do que no percurso 1, sendo um anterior à mudança de direção e outro posterior.

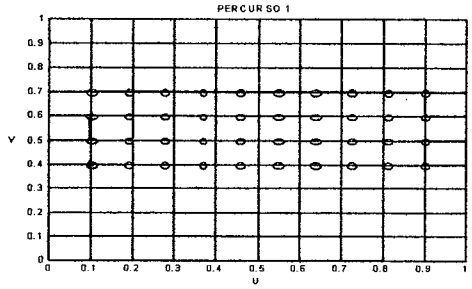

FIGURA 5.37: Percurso simples entre isoparamétricas.

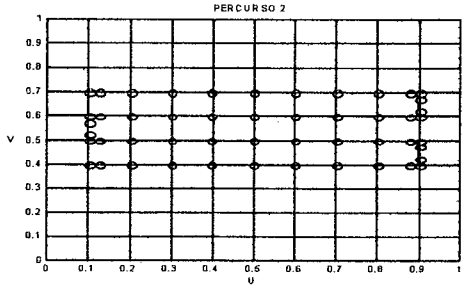

FIGURA 5.38: Percurso com maior número de pontos entre isoparamétricas.

A figura 5.39 ilustra os pontos cartesianos obtidos para os dois percursos de parâmetros sob uma superfície plana modelada a partir de uma superfície de Bézier (Apêndice 2).

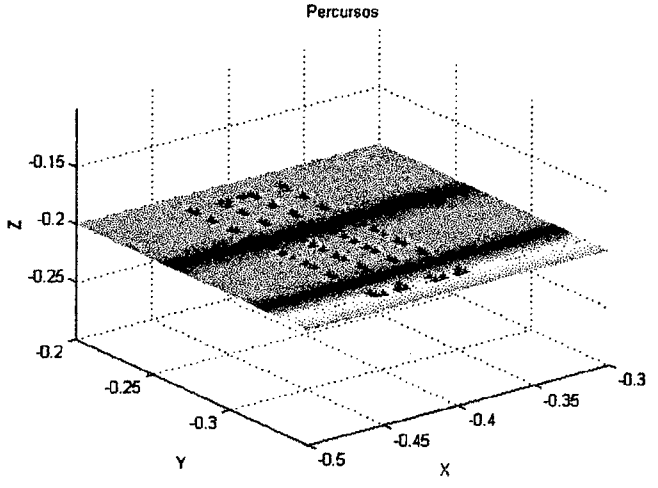

FIGURA 5.39: Comparativo entre os percursos (1) e (2), ilustrados nas figuras 5.37 e 5.38, sob a superfície plana modelada.

Com os pontos cartesianos obtidos a partir do procedimento detalhado na seção 4.4.1, é realizada a aproximação destes pontos através de curvas "B-Splines" de quarta ordem com vetor de referência uniforme e multiplicidade 2 nos extremos, como descrito na seção 4.4.2, e assim obter a trajetória cartesiana para o cumprimento da tarefa num total de 120 segundos especificados.

A figura 5.40, a seguir, ilustra os resultados obtidos para as trajetórias de posição no tempo dos dois diferentes percursos nos parâmetros da superfície. Em primeira análise, podemos observar nos gráficos que os trechos com mudança de direção ficam suavizados no percurso (2), principalmente na coordenada y, ocupando maior faixa de tempo devido ao maior número de pontos deste.

A figura 5.41, a seguir, ilustra o resultado das velocidades no tempo obtidas para os dois diferentes percursos especificados. Comparando os dois gráficos podemos notar que os trechos de velocidade constante na coordenada y ficam mais estreitos e com valor mais elevado (~40%) no percurso (2), devido à menor quantidade de tempo para o cumprimento deste, mas em compensação nos trechos com mudança de direção tem-se mais tempo para o seu cumprimento o que resulta numa suavização das curvas de velocidade tanto na coordenada y como na coordenada x.

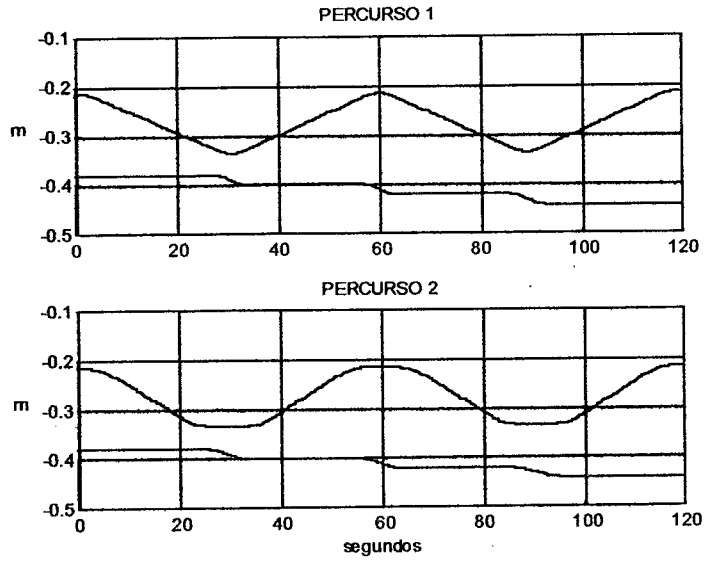

FIGURA 5.40: Comparativo entre as trajetórias de posição para diferentes distribuições de pontos dos percursos nos parâmetros da superfície plana.

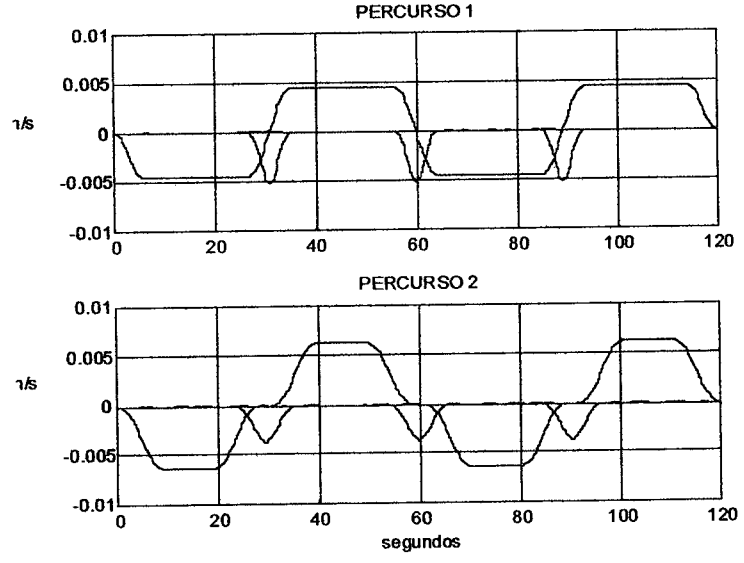

FIGURA 5.41: Comparativo entre as trajetórias de velocidade com diferentes distribuições de pontos dos percursos nos parâmetros da superfície plana.

O resultado mais expressivo pode ser notado comparando as trajetórias de aceleração, ilustradas na figura 5.42. Pode-se observar nos gráficos das trajetórias de aceleração dos percursos

(1) e (2), que nos trechos com mudança de direção, os níveis de aceleração baixaram sensivelmente no percurso (2), principalmente para a coordenada x que teve uma diminuição pela metade (2 m/s^2 para 1 m/s^2). Neste gráfico pode ser notada também uma diminuição, porém menor, das acelerações nos trechos inicial e terminal do percurso (2) para a coordenada y, devido à menor quantidade de pontos inseridos nestes trechos.

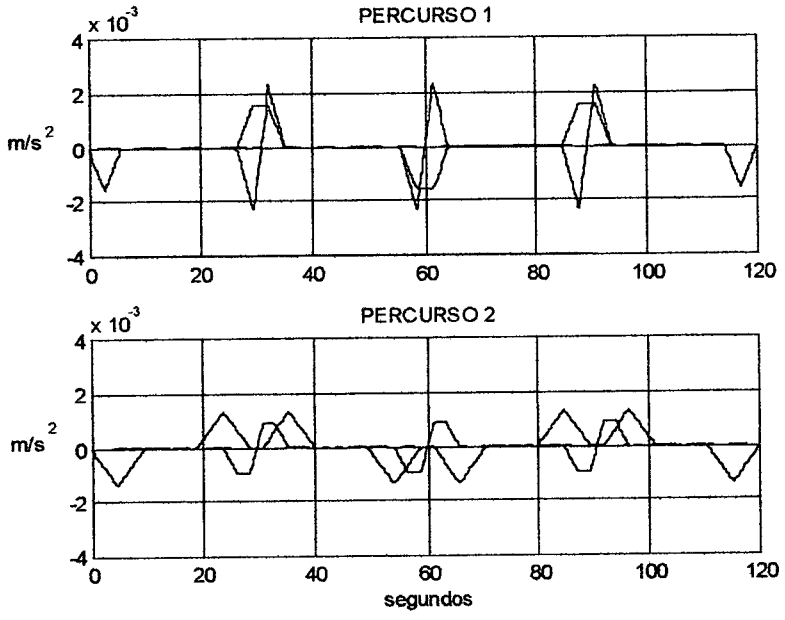

FIGURA 5.42: Comparativo entre as trajetórias de aceleração com diferentes distribuições de pontos nos percursos dos parâmetros da superfície plana.

A figura 5.43 ilustra as trajetórias cartesianas 3D, numa vista superior e em detalhe, mostrando a sensível variação da curva em função do número de pontos inseridos no seu polígono de geração.

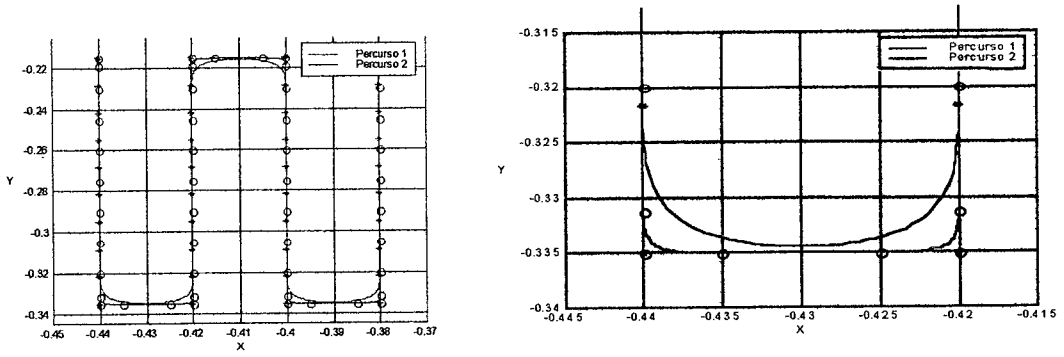

FIGURA 5.43: Comparativo das trajetórias cartesianas 3D com diferentes distribuições de pontos dos percursos nos parâmetros da superfície.

Com isso, o que se queria demonstrar nestes resultados, é uma primeira característica importante no uso das formulações das curvas "B-Splines" para obtenção de trajetórias no tempo: a possibilidade de antecipar o controle dos níveis de velocidade e aceleração através da seleção adequada do percurso de pontos para aproximação.

5.3.3 EFEITO DA VARIAÇÃO DO VETOR DE REFERÊNCIA

O teste 3 avalia a possibilidade de controle das curvas de posição, velocidade e aceleração na geração das trajetórias cartesianas no tempo com a variação do vetor de referência utilizado na formulação das curvas "B-Splines".

Os percursos (1) e (2) de pontos nos parâmetros da superfície usado neste teste são idênticos ao usado no percurso (1) do teste 2, bem como a superfície plana modelada a partir de uma superfície de Bézier. Ambos os percursos são especificados para serem executados em 45.5 segundos e aproximados através das formulações de curvas "B-Splines" de quarta ordem com multiplicidade 2 nos extremos. Porém, neste teste, o vetor de referência utilizado no percurso (1) é uniforme (~1.11s entre pontos), enquanto que no percurso (2) tem-se um vetor de referência não uniforme, onde os trechos com mudança de direção possuem maior tempo para execução (1.5s) e os outros trechos menor tempo para execução (1s). Os resultados dos dados de posição, velocidade e aceleração obtidos para o teste proposto estão apresentados nas figuras 5.44 à 5.46.

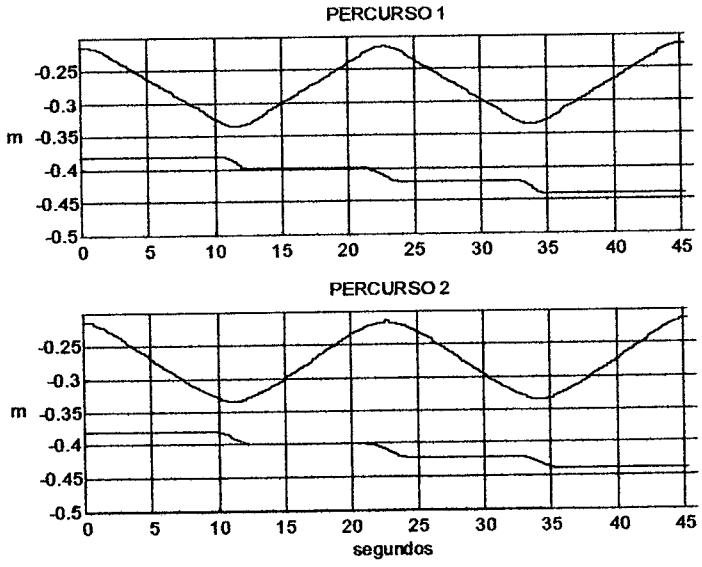

FIGURA 5.44: Comparativo das trajetórias de posição com vetores de referência uniforme (1) e não uniforme (2).

Inicialmente, nestas figuras podemos observar que para o percurso (1), apesar da mudança no tempo de execução com relação ao teste (2), a forma dos gráficos de posição, de velocidade e de aceleração permanece a mesma com relação ao teste (2), porém com valores maiores de velocidade

e aceleração, como era de se esperar. Já no percurso (2), pode-se notar que os trechos das trajetórias de posição tem maior montante de tempo para as mudanças de direção resultando numa suavização nestes, notada nas coordenada x e y, como era de se esperar, mas principalmente na coordenada y.

Quanto às trajetórias de velocidade, a diferença entre o uso de um vetor de referência uniforme e não uniforme é mais facilmente perceptível. Na coordenada y, observa-se que as faixas com mudança do valor de velocidade são maiores, apesar dos níveis de velocidade não terem baixado. Já na coordenada x, observa-se também o mesmo efeito, porém com uma diminuição significativa (aproximadamente 25%) dos níveis de velocidade.

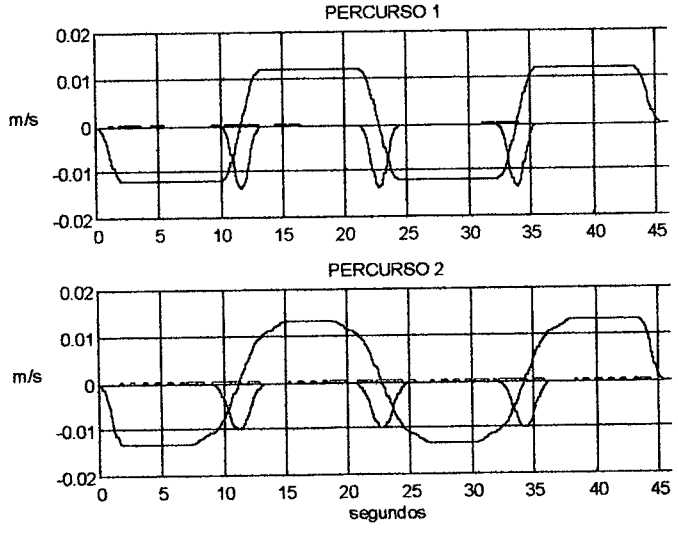

FIGURA 5.45: Comparativo das trajetórias de velocidade com vetores de referência uniforme (1) e não uniforme (2).

Finalmente, nos gráficos das trajetórias de aceleração, a diferença entre o uso de um vetor de referência uniforme e não uniforme fica mais acentuada. Nestes, observa-se que os níveis de aceleração nos trechos com mudança de direção ficam significativamente reduzidos (aproximadamente 40%).

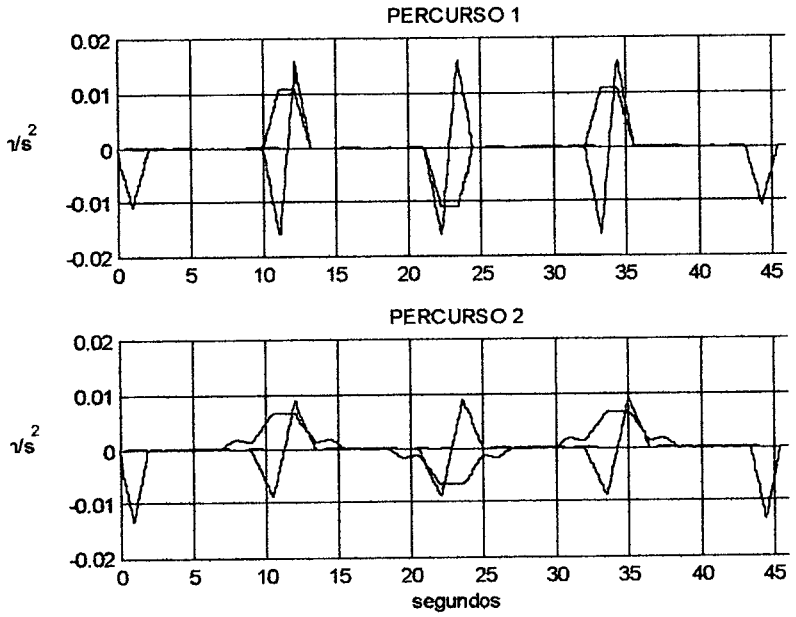

FIGURA 5.46: Comparativo das trajetórias de aceleração com vetores de referência uniforme (1) e não uniforme (2).

A figura 5.47, a seguir, ilustra a vista superior das trajetórias cartesianas 3D, no todo e em detalhe, demonstrando que a variação da forma destas para um vetor de referência uniforme e não uniforme é muito pequena.

Deste modo, fica demonstrado nestes resultados uma segunda característica importante no uso das formulações de curvas "B-Splines" para obtenção de trajetórias no espaço cartesiano: a possibilidade de antecipar o controle dos níveis de velocidade e aceleração através da seleção adequada do vetor de referência no uso da formulação de curvas "B-Splines".

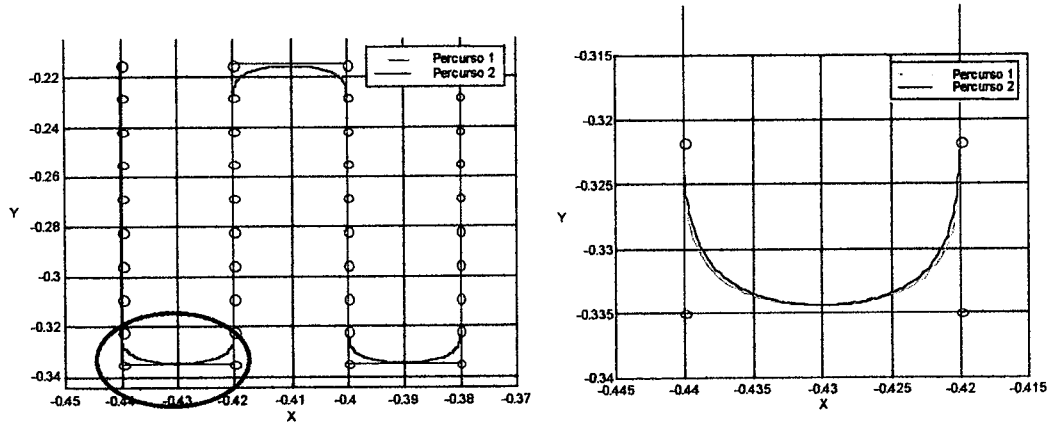

FIGURA 5.47: Comparativo das trajetórias cartesianas 3D com vetores de referência uniforme (1) e não uniforme (2). Vista superior da curva e ampliada em detalhe.

5.3.4 Efeito da Variação a Ordem da Curva sobre uma Superfície Esférica

O teste 4 visa verificar o comportamento das formulações das curvas "B-Splines" de acordo com a variação da ordem da curva a ser utilizada na aproximação dos pontos selecionados do percurso para execução.

Para tal, considere um percurso em zig-zag, como o ilustrado na figura 5.48, nos parâmetros de uma superfície esférica, como ilustrada na figura 5.49, para a obtenção dos pontos cartesianos do percurso. As aproximações são feitas através das formulações de curvas "B-Splines" de ordem 2 à 6.

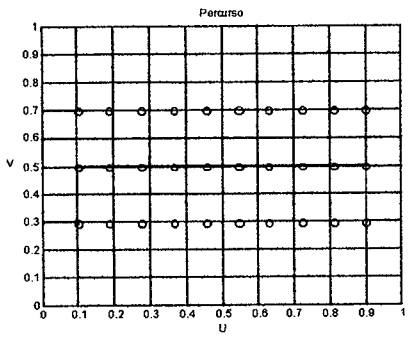

FIGURA 5.48: Percurso em Zigzag.

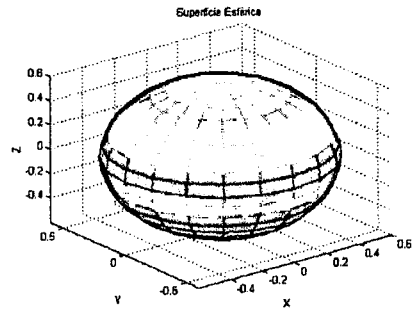

FIGURA 5.49: Superfície Esférica.

Na figura 5.50, a seguir, podemos observar as diferentes trajetórias de posição para as aproximações das coordenadas X, Y e Z levantadas. Inicialmente pode-se notar nos gráficos de posição das coordenadas X e Y que as curvas de ordem 2 aproximam os pontos através de segmentos lineares que passam nos pontos obtidos, enquanto que as curvas de ordem 6, conseqüentemente segmentada em polinômios de grau 5, ficam mais distantes dos pontos, sendo estas então curvas mais suaves. Na coordenada Z, nota-se uma diferença menos perceptível com a variação da ordem da curva.

Quanto aos gráficos de velocidade, apresentados na figura 5.51, o efeito de suavização das curvas com o aumento da ordem destas também pode ser percebido, principalmente na coordenada Z, em que a curva descontínua de velocidade de ordem 2 tem seus níveis suavizados em cerca de 50%, o que demonstra um resultado bastante satisfatório.

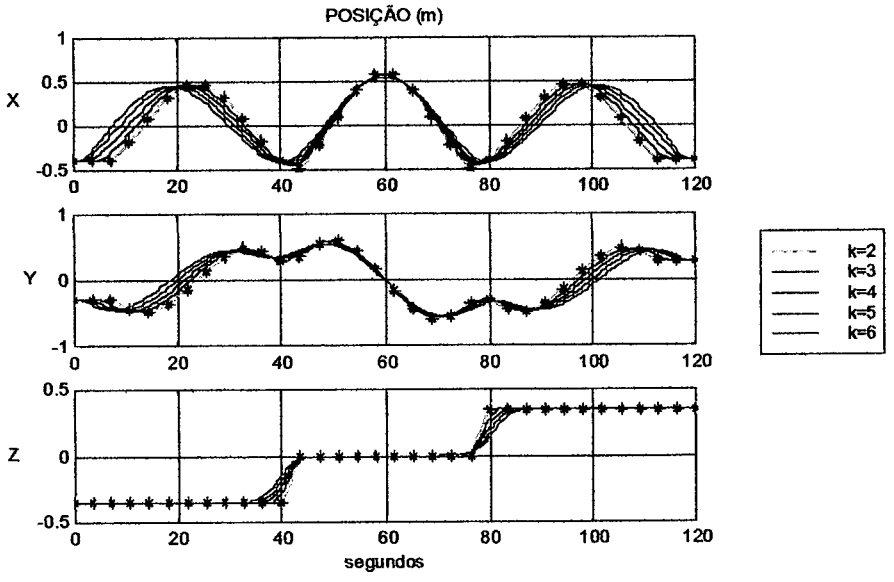

FIGURA 5.50: Comparativo das trajetórias de posição para curvas de ordem k=2,...,6.

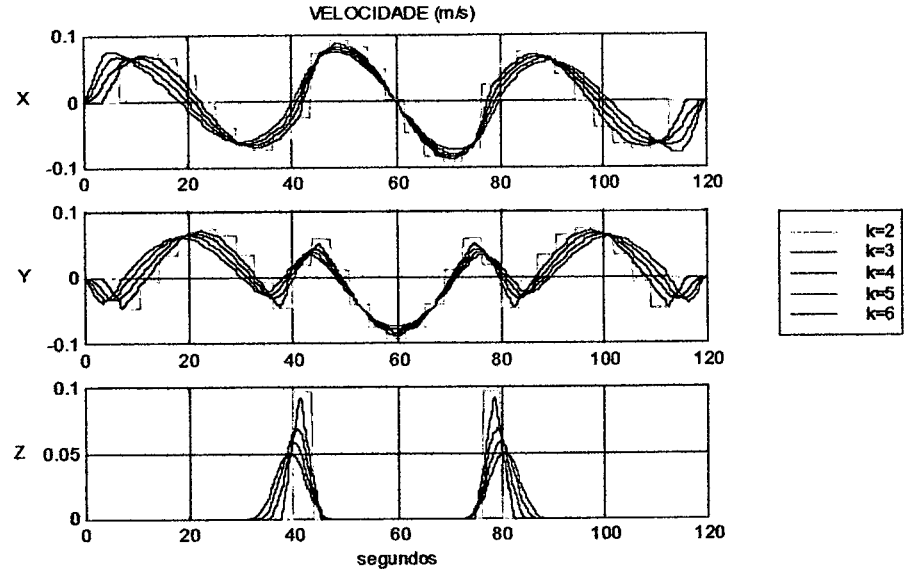

FIGURA 5.51: Comparativo das trajetórias de velocidade para curvas de ordem k=2,...,6.

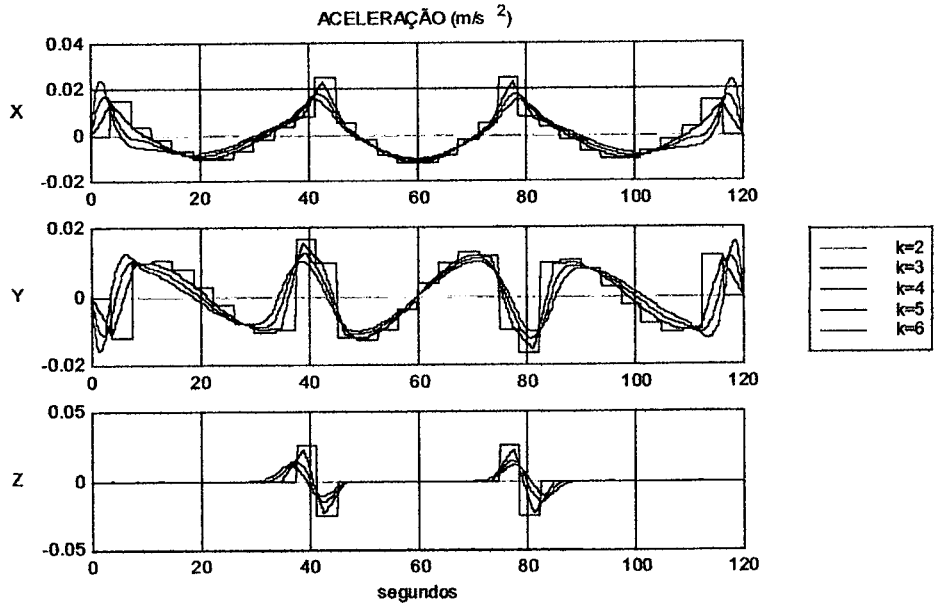

FIGURA 5.52: Comparativo das trajetórias de velocidade para curvas de ordem k=2,...,6.

Finalmente, na figura 5.52, são apresentados os resultados das trajetórias de aceleração, onde então as diferentes ordens das curvas fazem demonstrar mais claramente a suavização dos níveis de

aceleração mais claramente. Apesar de tal resultado ser perceptível nas três coordenadas, novamente a coordenada Z demonstra uma diminuição nos níveis de aceleração em torno de 50%.

A figura 5.53 ilustra a vista isométrica das trajetórias cartesianas 3D obtidas por aproximação através de curvas "B-Splines" de ordem 2 à 6. Nesta figura podemos visualizar que apesar de a diferença nos gráficos de posição, de velocidade e de aceleração no tempo ser bastante perceptível, a variação destas no espaço cartesiano não são tão perceptíveis, a menos nos trechos com mudança de direção, conforme a quantidade de pontos distribuídos no percurso idealizado.

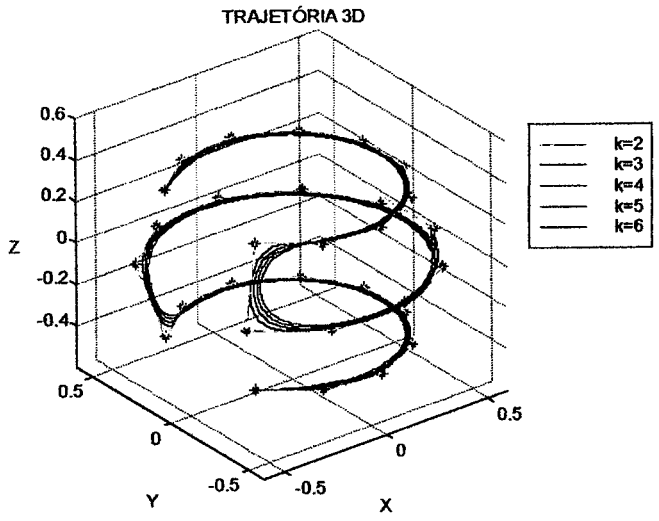

FIGURA 5.53: Variação das trajetórias para diferentes ordens da curva k=2,...,6.

A tabela 5.3, a seguir, exibe os tempos totais de processamento num computador Pentium 133 Mhz e 16 Mb Ram para geração das curvas de posição, velocidade e aceleração do teste proposto, de acordo com a variação da ordem das curvas.

Ordem da curva	Tempo de Processamento (s)
k=2	0.50
k=3	0.55
k=4	0.60
k=5	0.72
k=6	0.93

TABELA 5.3: Tempo de processamento de acordo com a variação da ordem da curva e avaliada em 1000ptos.

Podemos notar nos dados da tabela que, como era de se esperar, o aumento da ordem da curva aumenta significativamente o tempo de processamento. Uma vez que o percurso proposto possui 30 ptos e a curva gerada foi avaliada em 1000ptos, comparando com o dados da tabela 5.1, podemos verificar a consistência dos dados e, ainda, observar que a influência da ordem da curva no tempo de processamento é mais significativa do que o aumento do número de pontos do polígono de definição ou o aumento do número de pontos avaliados sob as curvas.

Deste modo, fica demonstrada nestes resultados uma terceira característica importante no uso das formulações de curvas "B-Splines" para obtenção de trajetórias no espaço cartesiano: a possibilidade de antecipar o controle dos níveis de velocidade e aceleração através da seleção adequada da ordem das curvas.

5.3.5 SIMULAÇÃO DO MANIPULADOR RHINO

Este teste visa demonstrar a possibilidade de interface dos dados gerados pelo programa desenvolvido em MatLab com um manipulador articulado do tipo Rhino, através da linguagem de programação RoboTalk [59].

Robo Talk é a linguagem de controle projetada para a série de robôs Rhino XR-3 e SCARA. Com esta pode-se controlar e utilizar completamente as características de um robô da série e dos controladores MARK III e IV. Esta linguagem é baseada em várias linguagens de controle robótico industriais, incluindo: a linguagem VAL da UNIMATION, a linguagem de operação da CINCINNATI MILACRON, a linguagem DARL da SEIKO, a linguagem RAIL da AUTOMATIX e, outras linguagens similares.

Neste teste, utilizaremos a linguagem Robo Talk para realizar a interface entre os dados obtidos da geração fora de linha de trajetórias cartesianas de posição e o controlador do robô Rhino XR-3. Este robô é um manipulador articulado com cinco eixos e seis motores.(Figura 5.54)

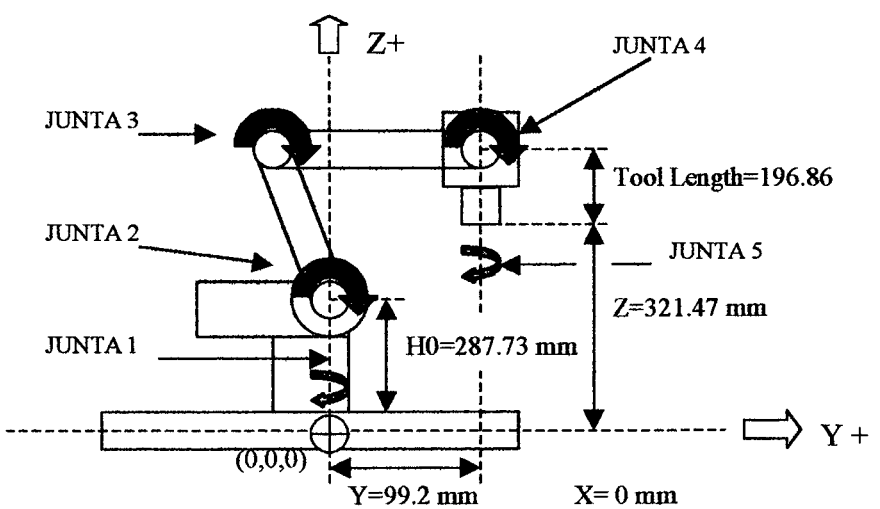

FIGURA 5.54: Localização das juntas ----, do sistema de coordenadas inercial --- e, dos dados da configuração da posição zero --- do robô Rhino XR-3.

Os cinco primeiros motores são usados para controlar a posição das cinco juntas e o sexto é responsável pela abertura e fechamento da garra do manipulador.

A linguagem Robo Talk suporta a programação de pontos no sistema de coordenadas cartesiano, permitindo ao programador especificar distâncias lineares, com relação a uma dada origem. Cinco coordenadas são utilizadas para definir um ponto: as posições X, Y e Z, dadas em distâncias lineares (mm ou in), e, os eixos A e T, que são deslocamentos angulares dados em graus. Neste teste são gerados dados apenas para a programação das coordenadas cartesianas de posição, sendo as coordenadas angulares constantes.

Inicialmente, com o dispositivo de aprendizagem ("Teach Pendant"), foi realizado o mapeamento de uma região no espaço de trabalho do robô Rhino, considerando o sistema de coordenadas de operação coincidente com o sistema inercial do manipulador. (Figura 5.55)

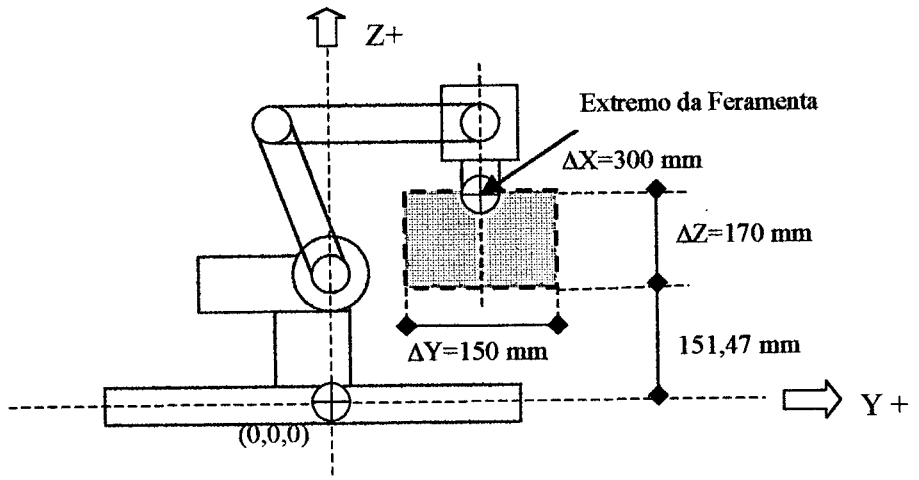

FIGURA 5.55: Espaço de trabalho mapeado, com relação a posição zero da extremidade da ferramenta, para o teste com o robô XR-3.

A partir da área mapeada foi modelada uma superfície plana, ilustrada na figura 5.56, através da formulação de superfície de Bézier. Especificou-se um percurso em zig-zag com dez passos nos parâmetros desta superfície, ilustrado na figura 5.57, para obtenção do polígono de definição da curva "B-Splines" a ser gerada.

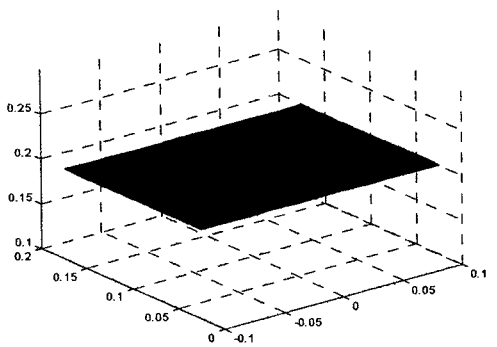

FIGURA 5.56: Superfície plana.

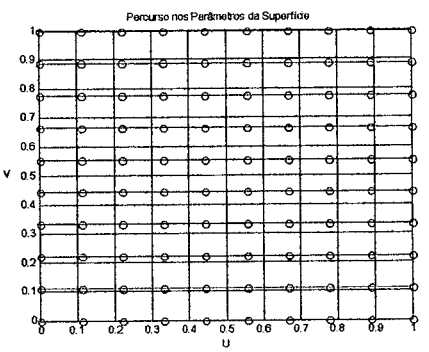

FIGURA 5.57: Percurso nos parâmetros.

De posse do polígono de definição, este pode ser aproximado através da formulação de curvas "B-Splines" com multiplicidade m=2 nos extremos e ordem k=4, para então serem obtidas as posições das coordenadas X, Y e Z no espaço de trabalho do manipulador, ilustradas nas figura 5.58. As trajetórias de posições no tempo são geradas, ilustradas na figura 5.59, sendo necessária a especificação de um tempo de 60 segundos para o cumprimento da tarefa. Porém, esta especificação não possui valor para o teste proposto já que estamos interessados apenas em obter as posições cartesianas do percurso a partir da curva gerada.

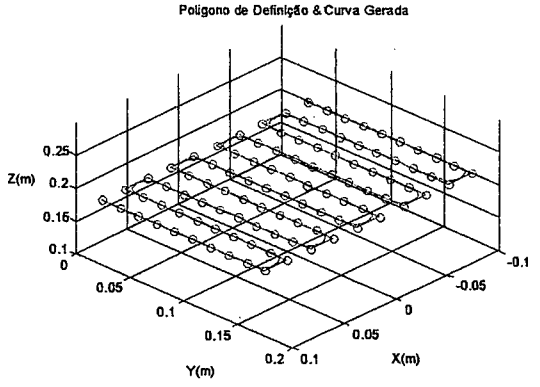

FIGURA 5.58: Polígono de definição e curva"B-Splines" 3D.

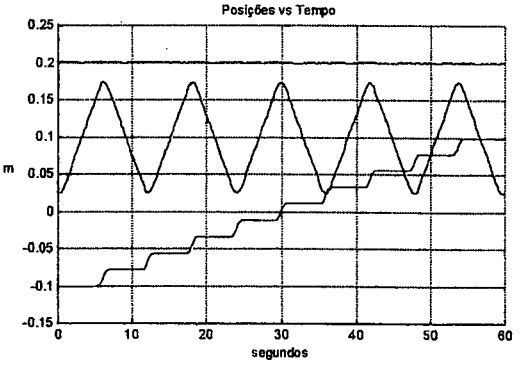

FIGURA 5.59: Coordenadas cartesianas no tempo.

Uma vez geradas as curvas das coordenadas cartesianas, estas podem ser avaliadas em tantos pontos quanto necessários. Neste teste pegamos uma amostragem de 250 pontos, como ilustrado na figura 5.60, de modo a gerar os dados para a programação e o controle no espaço cartesiano do manipulador Rhino.

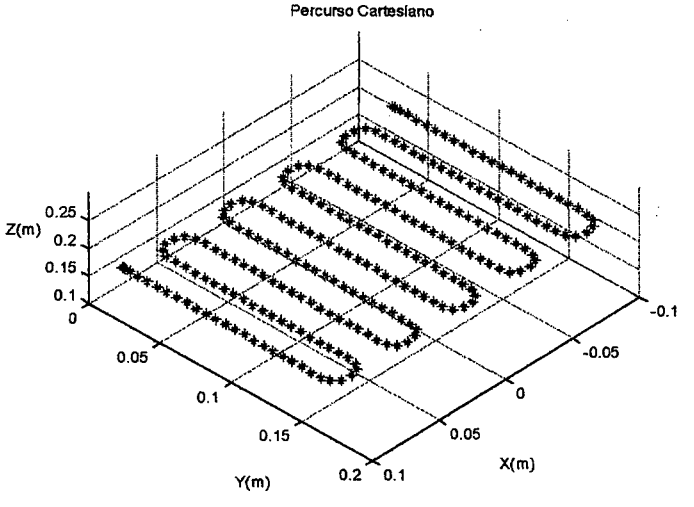

FIGURA 5.60: Percurso de pontos cartesianos para a programação e o controle no espaço cartesiano do manipulador Rhino.

Uma vez gerados os pontos cartesianos, devemos agora realizar a interface entre o programa de geração de dados e o software de entrada de dados para o controlador do manipulador. Para tal, foi utilizada uma rotina, figura 5.61, para imprimir em arquivo o comando da linguagem Robo Talk correspondente ao percorrimento de um percurso no espaço cartesiano. No caso, o comando SETPATHX <x>,<y>,<z> permite a programação deste tipo de percurso gerado. Como saída do teste temos um arquivo ascii com o comando para cada ponto cartesiano gerado. Este arquivo pode ser aberto no editor de texto do MSDOS, como ilustrado na figura 5.62, ou mesmo no editor do software de entrada de dados para o controlador, já que o sistema operacional deste software é o próprio MSDOS. A extensão do arquivo deve ser **.rt**, para que este possa ser diretamente carregado para o controlador do manipulador.

```
FUNCTION WRT_ATR (D);
FID = FOPEN('TESTE5.RT','W');
FOR I=1:LENGTH(D(:,1))
FPRINTF(FID,'SETPATHX <%3.0F>,<%3.0F>,<%3.0F> \N',D(I,:));
END;
FPRINTF(FID,'PATH');
FCLOSE(FID);
```

FIGURA 5.61: Rotina para salvar os dados gerados em linguagem Robo Talk num arquivo tipo ascii para o controlador do Rhino.

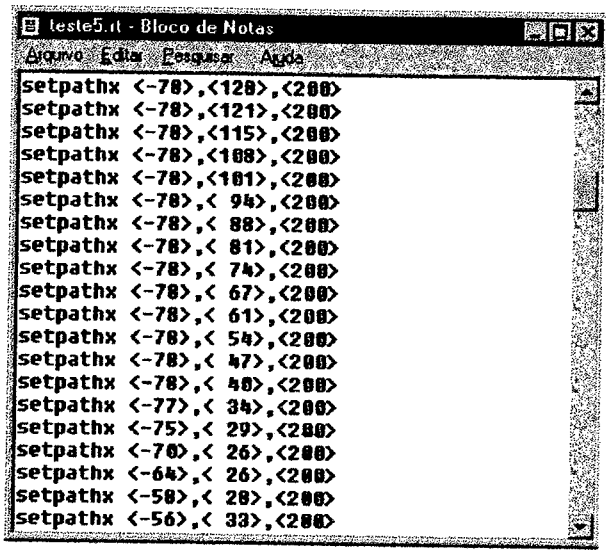

FIGURA 5.62: Arquivo teste5.rt no formato ascii gerado para programação do Rhino.

De posse do arquivo gerado, este pode ser carregado ao controlador do manipulador, e rodado. Para tal, é necessário que o disquete que contem o arquivo esteja disponível no computador que possui o programa de entrada de dados da linguagem RoboTalk. Após aberto o programa e estando disponível o teste gerado, através do comando **LOAD TESTE5.RT** o arquivo pode ser carregado para o computador e, com o comando **RUN TESTE5.RT**, o teste pode ser rodado.

Apesar de o movimento ser aproximado através de uma interpolação linear entre os pontos especificados, o que não possibilita a obtenção de um movimento preciso com relação as curvas geradas, o teste foi realizado com sucesso, já que o efetuador do manipulador percorreu os dez passos na área especificada para movimentação.

5.3.6 SIMULAÇÃO DO MANIPULADOR SCARA

Neste teste são apresentados os resultados encontrados no projeto [62] implementado com o robô Inter, do tipo SCARA, que foi desenvolvido na Universidade Federal de Zurique, ETH, sob encomenda da Universidade Federal de Santa Catarina, UFSC, e, atualmente, pertencente ao Laboratório de Automação Industrial (Fig. 5.63).

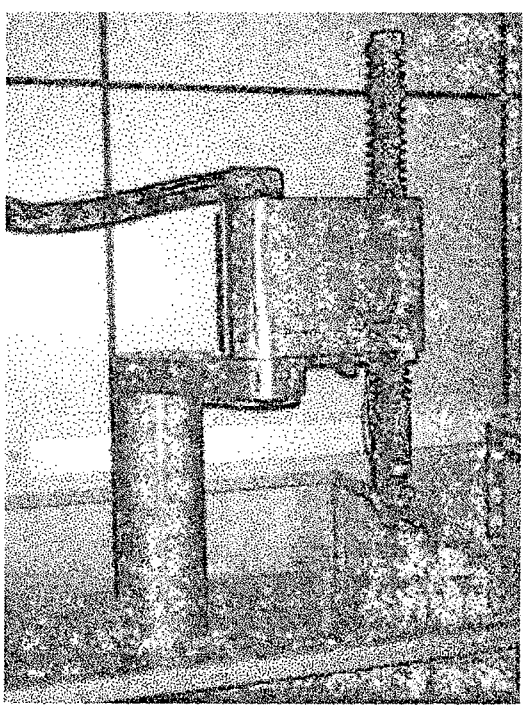

FIGURA 5.63: Foto do Robô SCARA no Laboratório de Automação Industrial (LAI).

O objetivo deste projeto foi desenvolver um novo módulo gerador de trajetórias em substituição ao existente, que possibilitava somente o movimento ponto-a-ponto. Desejava-se com este novo módulo permitir ao robô percorrer caminhos curvilíneos, indo de um ponto a outro em seu espaço de trabalho, passando por pontos intermediários pré-determinados sem paradas, representando as trajetórias através de funções suaves de posição, velocidade e aceleração.

A idéia consistiu em desenvolver uma interface gráfica amigável que permitisse ao usuário especificar uma seqüência de pontos inicial, final e intermediários para representar o caminho a ser seguido pelo manipulador no seu espaço de trabalho, devendo ser também especificado o tempo desejado para que o movimento seja executado. Com isso, o planejador geraria uma seqüência temporal de valores de posição, velocidade e aceleração e os enviaria ao sistema de controle do robô para efetivar o movimento.

A proposta de solução foi efetuada em dois módulos, onde o primeiro realiza a geração *off-line* dos parâmetros da trajetória utilizando funções do MatLab, assim como desenvolvido neste trabalho, e o outro um software capaz de ler, converter e enviar estes valores ao sistema de controle do robô.

O robô utilizado neste projeto é uma versão especial de robô articulado e possui quatro graus de liberdade. A primeira, segunda e quarta junta do robô são de rotação e giram em torno de eixos verticais; a terceira junta é de translação e se desloca verticalmente. A figura 5.64a apresenta esquematicamente o manipulador e suas respectivas juntas e elos e a figura 5.64b, os motores que movimentam estas juntas.

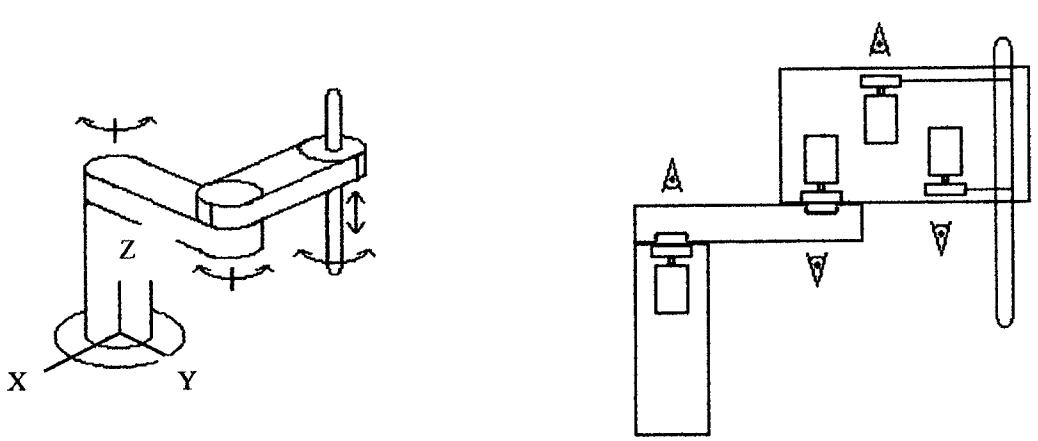

FIGURA 5.64: Esquema da Estrutura. a) Aspecto de juntas do robô e b) A montagem de seus motores.

Para proporcionar a interface do programa para geração de trajetórias criou-se uma figura no plano XY onde a região entre as duas circunferências concêntricas representa a área de trabalho do robô no plano XY. (Figura 5.65) Assim, o usuário utiliza o *mouse* para indicar os pontos da trajetória que o robô deve percorrer.

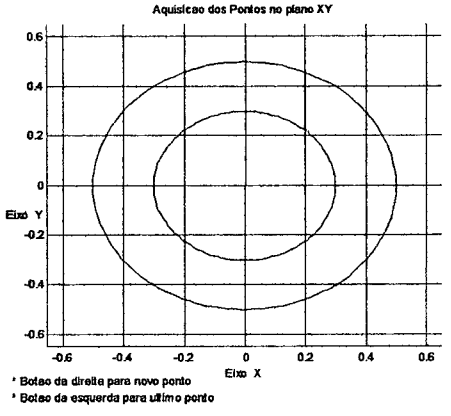

FIGURA 5.6 : Área de Trabalho do robô Inter no plano XY.

Os pontos inicial e final e os pontos de passagem (*via-points*) são armazenados e utilizados para gerar uma *"B-Spline"* que irá representar as posições no tempo e, através da primeira e segunda derivadas da *"B-Spline"*, encontra-se funções de velocidade e aceleração, respectivamente. Amostrando estas funções encontra-se a velocidade e aceleração desejadas do manipulador no espaço cartesiano. De posse dos valores de posição, velocidade e aceleração cartesianos, é feita a transformação destes em valores de deslocamento, velocidade e aceleração de juntas através da cinemática. Os valores calculados são plotados para verificar se estão dentro dos limites especificados. Os valores de deslocamento, velocidade e aceleração angulares calculados e verificados são armazenados em uma matriz n x m, onde n representa o número de pontos amostrados que compõem a trajetória e m são as nove colunas dos valores calculados.

Esta matriz é salva em um arquivo de dados que pode ser então transmitido à memória do robô. O arquivo é gerado no editor de texto do DOS e então transferido à memória do robô, no armário de controle, para que seja possível exibir o mesmo em uma janela de comunicação no ambiente Xoberon [63]. A transferência é feita por um *software* chamado TFTP (*"Trivial File Transfer Protocol"*), pertencente ao laboratório de robótica, que envia o arquivo em formato ASCII através da rede de comunicação existente entre o computador com o ambiente XOberon e o armário de controle. Cada uma das colunas do arquivo de dados representa a posição, velocidade e aceleração das juntas do robô e, a cada linha tem-se um novo passo de atualização de caminho (fig. 5.66).

$$\begin{matrix} \theta_{01} & \theta_{02} & \theta_{03} & \dot{\theta}_{01} & \cdots & \ddot{\theta}_{03} \\ \theta_{11} & \theta_{12} & \theta_{13} & \dot{\theta}_{11} & \cdots & \ddot{\theta}_{13} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ \theta_{n1} & \theta_{n2} & \theta_{n3} & \dot{\theta}_{n1} & \cdots & \ddot{\theta}_{n3} \end{matrix}$$

Figura 5.66: Arquivo Idealizado.

Os dados devem ser enviados ao controlador do robô, para que sejam tratados pelo sistema de controle, que por sua vez acionará, os atuadores para movimentar o manipulador. A taxa de atualização de caminho é definida como sendo de 1ms. Isto que dizer que a cada milisegundo novos valores são enviados ao sistema de controle do robô, que irá tentar corrigir o erro de posição existente, neste intervalo de tempo.

No primeiro teste foi definido um caminho no plano XY, composto por três pequenos segmentos. A figura 5.67 ilustra a área de trabalho do robô, no plano horizontal, onde pode ser observada a trajetória pretendida, composta pelos três segmentos de reta, juntamente com a função

interpolada que representa este caminho. Os pontos escolhidos pelo usuário são inseridos através da utilização do *mouse*, indicando-os diretamente na área de trabalho do robô.

A figura 5.68 ilustra o mesmo caminho descrito anteriormente. Pode ser observado que a função interpolada começa no ponto inicial e atinge o ponto final do caminho proposto. Nesta mesma figura, é possível observar que os dois pontos intermediários do caminho são responsáveis por sua forma, devido à função interpolada se aproximar destes dois pontos.

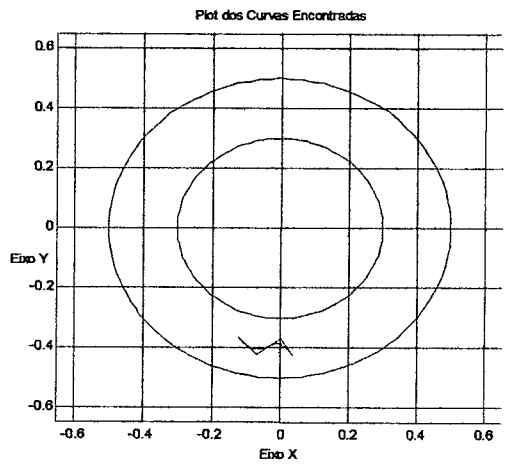

FIGURA 5.67:- Movimento no plano XY

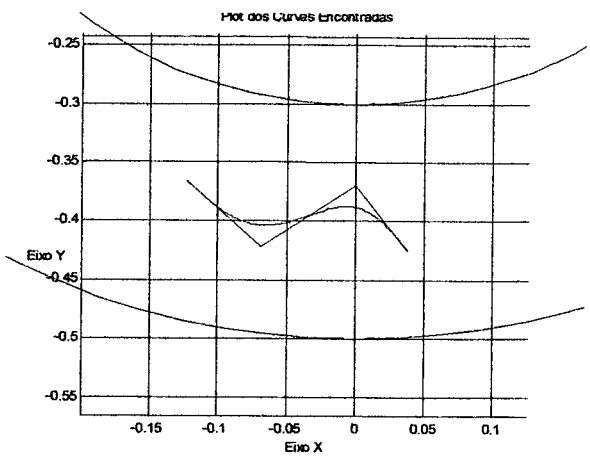

FIGURA 5.68: Aproximação do caminho proposto e função interpolada.

A figura 5.69 ilustra o segundo teste realizado, onde pode ser observada a função interpolada do caminho pretendido no espaço cartesiano, representada em três dimensões. Esta função representa um caminho através de uma elipsóide, com a intenção de testar o movimento do manipulador na execução de uma trajetória mais complexa. A construção desta curva é feita a partir de uma circunferência aproximada no plano horizontal e uma reta no plano vertical.

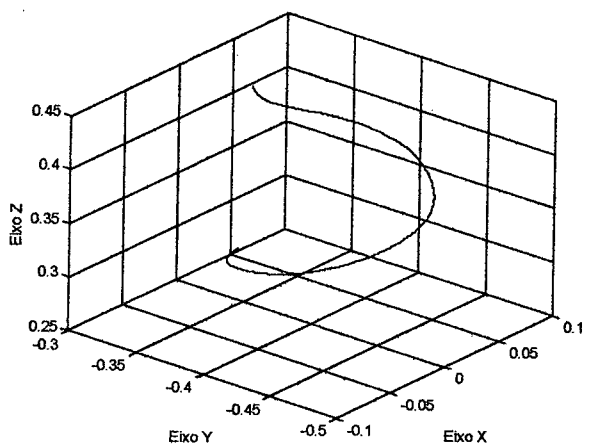

FIGURA 5.69: Representação em três dimensões da trajetória.

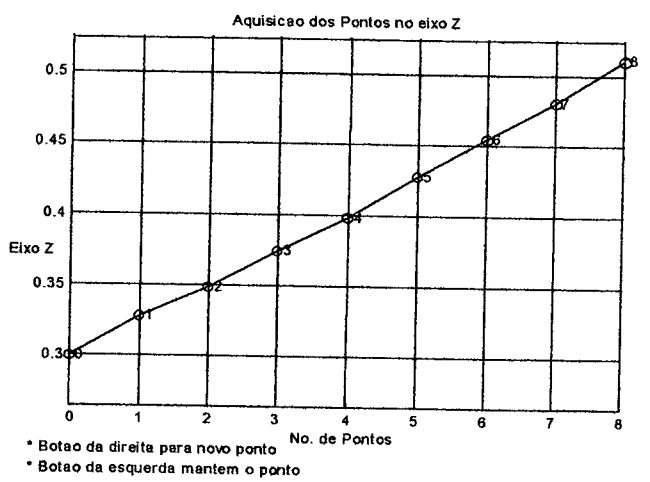

FIGURA 5.70: Representação do caminho no plano vertical.

A figura 5.70 ilustra o caminho composto por seus 8 segmentos de reta. Este caminho é representado no eixo Z para ilustrar a área de trabalho para a coordenada vertical.

As figuras 5.71, 5.72 e 5.73 ilustram a história temporal da posição, velocidade e aceleração angular da trajetória pretendida, respectivamente.

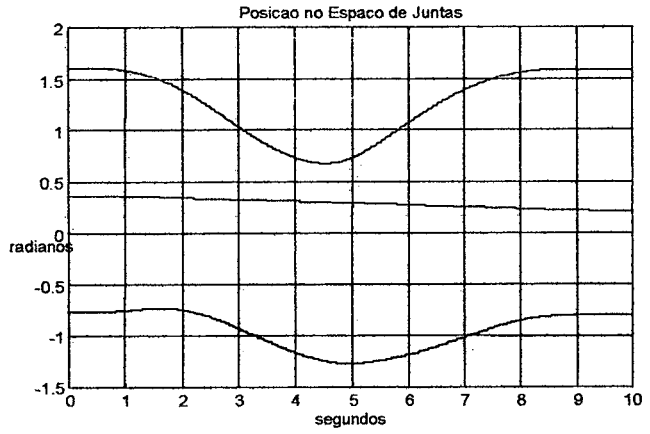

Figura 5.71: História temporal da posição angular.

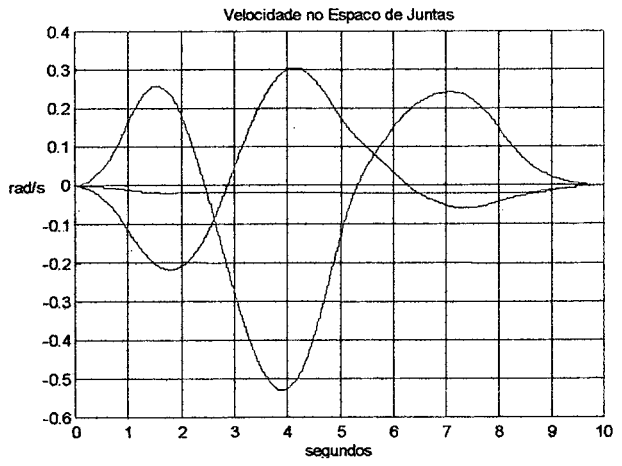

FIGURA 5.72: História temporal da velocidade angular.

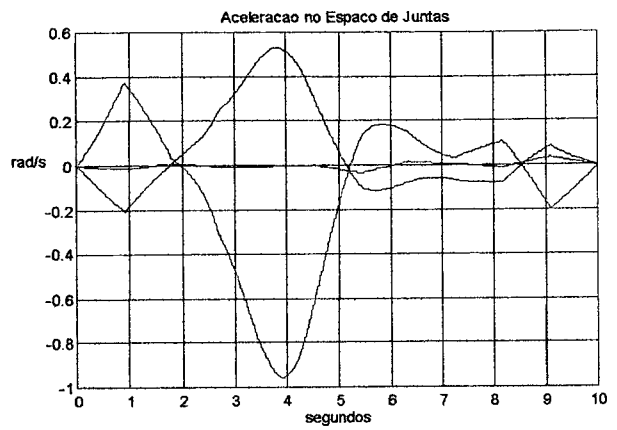

FIGURA 5.73: História temporal da aceleração angular.

Na pesquisa [63], o autor constata a necessidade de desenvolver uma forma de comparar a trajetória idealizada com a trajetória real executada pelo manipulador, caracterizando uma função não implementada na integração do sistema de programação e o de controle. Este é um ponto crítico, pois o fato das trajetórias de referência serem suaves não implica em um mesmo comportamento do manipulador. Para realização desta função é necessário que o manipulador esteja monitorado por sensores, tornando acessíveis os dados dos valores de juntas. Uma vez disponíveis estes dados podem ser comparados com os valores de referência, gerados pelos procedimentos de programação, permitindo a verificação do comportamento real do sistema. Esta verificação pode ser realizada visualizando os dados graficamente numa interface desenvolvida na própria linguagem de programação do controlador do manipulador, ou ainda, exportando os dados de saída, num determinado formato de arquivo ou protocolo de transferência, para visualização no próprio sistema de simulação e programação.

Entretanto, algumas conclusões importantes foram constatadas pelo autor [62]:

1. a transformação dos valores de velocidade do espaço cartesiano em valores equivalentes no espaço de juntas ocorre de maneira controlada e verificada;

2. os gráficos com a história temporal da velocidade e aceleração gerados com uso de "B-Splines" não possuem variações acentuadas em seus valores;

3. a utilização de figuras para representar a área de trabalho do robô, possibilitou uma grande facilidade e flexibilidade na escolha dos pontos, além de uma melhor visualização da forma do caminho proposto;

4. os dados de deslocamento, velocidade e aceleração de juntas puderam ser diretamente acessados pelo sistema de controle do robô.

CAPÍTULO 6

CONCLUSÕES E RECOMENDAÇÕES

6.1 CONCLUSÕES

Este trabalho apresentou o estudo da programação dos robôs industriais, através do uso de métodos gráficos e uma proposta de programa para o planejamento, simulação e programação de atividades de robôs industriais. A partir disso pode-se apresentar as seguintes conclusões:

Apesar de já existir uma grande quantidade de ferramentas de simulação e programação fora de linha de robôs industriais com resultados satisfatórios, algumas destas revisadas ao longo deste trabalho, constata-se que esta atividade oferece um campo amplo de pesquisa. Em primeiro lugar, pode ser destacada a existência da dificuldade de padronização das linguagens de programação dos robôs industriais, tanto no que se refere à interface entre os sistemas CAD/CAM, quanto no que diz respeito aos controladores dos robôs industriais. Isso dificulta o uso destes tipos de sistemas na tarefa de programação, pois exige o desenvolvimento de módulos dedicados para cada tipo de robô utilizado. Outra dificuldade consiste na aquisição de dados a partir dos modeladores geométricos ou "tecnológicos", uma vez que existem diferentes

especificações para exportação de dados a partir destes sistemas e um padrão realmente mundial não está ainda estabelecido.

Outro ponto importante, e que vale ser ressaltado, é a dificuldade de calibração das atividades planejadas e sua execução, discutido na seção 3.5.4, conseqüente das incertezas e imprecisões com respeito ao modelo simulado e o real. Além disso, apesar da grande quantidade de simuladores em uso, a utilização de tecnologias recentes, como a inteligência artificial e as redes de comunicação no chão de fábrica, ainda é incipiente.

Ao analisar a metodologia e o programa desenvolvido neste trabalho, consideramos as seguintes conclusões.

O estudo em geral e a metodologia apresentada possibilitaram o conhecimento do ciclo de informação necessário no desenvolvimento e aplicação de sistemas CAD/CAM para o planejamento, simulação e programação de tarefas de robôs industriais. Na metodologia apresentada e no programa desenvolvido faz-se uso de técnicas atuais de modelagem geométrica e de manipulação e aquisição de dados de modelos gerados em sistemas CAD convencionais. Isso possibilita a automatização do planejamento e geração de dados para o controle de tarefas de fabricação no espaço cartesiano, através de uma interface gráfica interativa e amigável. Além disso, estes dados podem ser mapeados e analisados no espaço de juntas do manipulador a ser simulado, para então serem transferidos para o sistema de controle do robô industrial que se deseja programar.

A interface desenvolvida incrementa a pesquisa e a didática na integração e racionalização do processo produtivo através da solução de problemas específicos da programação de manipuladores. Se aplicada em atividades industriais, a metodologia de programação fora de linha proposta pode contribuir na redução do tempo de robô fora de produção, na remoção do programador de ambientes potencialmente perigosos e insalubres, e na

simplificação de tarefas complexas. Também possibilita antecipar e testar diferentes alternativas para o cumprimento das tarefas, com o uso de diversos tipos de manipuladores e diferentes posicionamentos da superfície no espaço, e ainda verificar o comportamento cinemático e dinâmico do manipulador no cumprimento de tarefas.

E ainda, o trabalho mostra a possibilidade de uso das formulações de curvas "B-Splines" para facilitar a geração de trajetórias de movimentação e precisão, mostrada na seção **5.3.1**. A metodologia permite grande flexibilidade e controle dos dados gerados, ressaltado nas seções **5.3.2** a **5.3.4**, tendo ainda possibilidade de geração dos dados para interface de diferentes robôs industriais, demonstrado nas seções **5.3.5** e **5.3.6**.

Finalmente, apesar do propósito intencional do sistema ser a programação fora de linha, o conhecimento destes tipos de sistema de programação pode ser de grande valor para a programação "on-line". O uso de superfícies e de "métodos automáticos e simplificados de geração de trajetórias" [32] pode ser expandido para a programação on-line. Tendo certos parâmetros pré-definidos de comportamento a serem estabelecidos, pode-se também gerar aplicação para sistemas de visão e reconhecimento de imagens 3D, através da resposta automática do manipulador.

6.2 RECOMENDAÇÕES

Como recomendações para trabalhos futuros, em relação ao estudo da programação dos robôs industriais com o uso de técnicas e recursos da computação gráfica, sugere-se o seguinte:

1. Aprofundar o estudo do uso das técnicas de computação gráfica no desenvolvimento da ferramenta de planejamento, simulação e programação de atividades de robôs industriais de modo a adotar novas estratégias para a geração de trajetórias a partir de modelos em CAD. com modelos geométricos mais refinados, como as curvas "B-Splines" racionais, as superfícies "B-Splines" não racionais e racionais, modelos sólidos, e ainda, modelos e dados não geométricos, como os diagramas de ligações e dados das propriedades dos materiais. E ainda a integração de técnicas de aquisição de dados não apenas geométricos, mas também referentes a propriedades dos objetos através do uso de modeladores geométricos e outras especificações para troca de dados, como o STEP.

2. Aplicar novas técnicas no desenvolvimento dos sistemas gráficos interativos para a programação fora de linha, ou em tempo real, dos robôs industriais, como por exemplo, a integração de sistemas de visão, sistemas de sensores e lógica binária, ou ainda, a utilização de sistemas dotados de técnicas com algoritmos de inteligência artificial.

3. Utilizar soluções analíticas na cinemática inversa e incluir os aspectos dinâmicos do manipulador, bem como o efeito da flexibilidade dos seus vários elementos, fazendo com que a trajetória considere de uma forma explicita estes efeitos. Uma vez que no procedimento são gerados os dados de posição, velocidade e aceleração de juntas de diferentes manipuladores, estes podem ser utilizados eficientemente no estudo do comportamento dinâmico do manipulador, incrementando assim a capacidade de análise e validação do sistema.

4. Inclusão de algoritmos de teste de colisão no espaço externo do manipulador, bem como, algoritmos de teste automático para decisão do melhor posicionamento do produto com relação ao manipulador para determinação das trajetórias cartesianas, e conseqüentemente das trajetórias de juntas.

5. Desenvolvimento de um sistema de simulação de uma célula de trabalho, seja de um tipo de atividade especifica de fabricação ou de maneira generalizada, com elementos geométricos pré-definidos, onde o usuário estabelece a disposição destes na célula de trabalho e os tipos de interações entre estes.

Com relação ao programa desenvolvido que faz uso da metodologia apresentada, sugere-se os seguintes melhoramentos:

1. . Utilizar recursos de programação, análise e visualização mais apurados através de linguagens de alto nível orientadas ao objeto. Uma interface interativa com os diversos módulos coexistindo num mesmo ambiente gráfico pode ser proposta utilizando outras plataformas gráficas como o C++, Java ou OpenGL

2. Aplicar bases de conhecimento em fabricação de modo a adequar as capacidades de geração de trajetórias aos parâmetros e às restrições reais das atividades de fabricação, tal como, polimento, pintura, solda etc...

3. Aperfeiçoar a técnica através do uso de procedimentos de otimização. Com o uso de algoritmos de otimização pode-se buscar a melhor forma para a distribuição dos pontos do percurso e posteiror aproximação ou interpolação, buscando suavizar as trajetórias em pontos críticos onde haja mudanças nos valores das derivadas.

REFERÊNCIAS BIBLIOGRÁFICAS

[01] ALVARES, A. J. & ROMARIZ, L. S. **Desenvolvimento de um Manipulador com Dois Graus de Liberdade Controlado Remotamente Via Internet** V Congresso de Engenharia Mecânica Norte-Nordeste, Fortaleza, 1998. v.1, p. 529-536.

[02] AMBLER, A. P. **Languages for Programming Robots.** Robotics and Artificial Intelligence. Springer-Verlag Berlin Heidelberg, 1984.

[03] ASADA, H. & SLOTINE, J. J. E. **Robot Analysis and Control**. John Wiley & Sons, 1986.

[04] ASFAHAL, C. R. **Robots and Manufacturing Automation**. John Wiley & Sons, 1992.

[05] BAO, H. **Computer Aided Process Planning.** Robotics and Factories of the Future. Berlin 1984.

[06] BIEN, C. **Simulation a Necessity in Safety Engineering.** Robotics World, Dezembro, 1992.

[07] BOOR, C., **Spline Toolbox for use with Matlab**. MathWorks, Inc., 1995.

[08] CARACCIOLO,R.; CERESOLE,T.; DeMARTINO,T. & GIANNINI,F. **From CAD Models to Assembly Planning.** Graphics and Robotics, Springer-Verlag, 1987.

[09] CORKE, P. I. **A Robotics Toolbox for Matlab.** IEEE Robotics & Automation Magazine, p.24-32, março 1996.

[10] CRAIG, J. J. **Robot Calibration Facilities Off-Line Programming.** Robotics World, março 1992.

[11] CRAIG, J. J. **Introduction to Robotics: Mechanics and Control**. Addison-Wesley, 1986.

[12] CRANE, C. D., DUFFY, J. & LOCKE, M. **Off-Line Programming and Path Generation for Robot Manipulators.** CAD Based Programming for Sensory Robots. Berlin: Springer-Verlag, 1988.

[13] DIAS, A., SANTOS, E. C. & GUENTHER, R. **Uma Interface Interativa para a Programação "Off-Line" de Manipuladores.** VII Congresso Nacional de Engenharia Mecânica. Valdívia, Chile, 1996.

[14] DIAS, A., TOLEDO, L. B. & DEISENROTH, M. **An Off-Line Programming System for Industrial Robots.** In: FLEXIBLE AUTOMATION AND INTELLIGENT MANUFACTURING 1999, Tilburg, Netherlands, Proceedings of the Ninth International Faim Conference. New York: Begell House Inc, 1999. v.1. p.835-846.

[15] DIAS, A., TOLEDO, L. B. & DEISENROTH, M. **A CAD/Robotics System to "Off-Line" Programming of Industrial Robots,** XV Congresso Brasileiro de Engenharia Mecânica, Anais em CD-ROM – código: AAEACG, Águas de Lindóia, São Paulo, 1999.

[16] ERTHAL, J. L. **Estudo de Métodos para a Solução da Cinemática Inversa de Robôs Industriais para Implementação Computacional**. Dissertação de Mestrado do Departamento de Engenharia Mecânica, UFSC. Florianópolis,1992.

[17] FOLEY, J. D., VanDAM, A., FEINER, S. K. & HUGHES, J. F., **Computer Graphics: Principles and Practice**, Addison-Wesley Publishing Company, 1996.

[18] FU, K. S., GONZALES, R. C., & LEE, C. S. **Robotics: Control, Sensing, Vision and Intelligence**. New York: McGraw-Hill, 1987.

[19] GROOVER, M. P. **Automation, Production Systems and Integrated Manufacturing**. Prentice-Hall, Inc. 1987.

[20] GROOVER M. P., WEISS M., NAGEL R. N. & ODREY N. G., **Industrial Robotics: Technology, Programming and Applications**, McGraw Hill Inc., 1986.

[21] GROOVER, M. P. & ZIMMERS, W., **CAD/CAM: Computer Aided Design and Manufacturing**, Prentice-Hall Inc., 1984.

[22] HANSELMAN, D, & LITTLEFIELD, B., **The student edition of MATLAB - version 4: User's Guide**. The MathWorks Inc., 1995.

[23] HAUTAU,L.A. & DiPIETRO **Planning Robotic Production Systems.** Handbook of Industrial Robotics. John Wiley & Sons, 1985.

[24] HERNANDEZ , J. T.; VILLALOBOS, J. & ROGRIGUEZ, C. F., **Ambientes de Programación y Planificación de Robots**, Universidad de Los Andes, Santafé de Bogotá, Colombia, 1991.

[25] HOLLINGSHEAD, L. .L. **Elements of Industrial Robot Software.** Handbook of Industrial Robotics. John Wiley & Sons, 1985.

[26] HORDESKI M. F., **CAD/CAM Techiniques**, Prentice Hall Co, 1986.

[27] IMAM, I. & DAVIS, J. E. **Robot Simulation and Off-Line Programming – an Integrated CAE/CAD Approach.** CAD Based Programming for Sensory Robots. Berlin: Springer-Verlag, 1988.

[28] JAMSHIDI, M., PIN, F. & PIERROT, F. **Robotic and Manufacturing Systems.** Proceedings of the World Automation Congress, Montepellier, França, 1996.

[29] KATAJAMAKI, M. & KANERVA, J. **CAD/CAM- Revolutionizing Robot Applications Design.** 14th International Symposium on Industrial Robots. 7th International Conference on Industrial Robot Technology. Gothenburg, 1984.

[30] KOREN, Y. **Robotics for Engineers**. McGraw Hill, Inc. 1985.

[31] LATOMBE, J. C., **Automatic Synthesis of Robots Programs From CAD Specification.** Robotics and Artificial Intelligence. Berlim: Springer-Verlag, 1984.

[32] LIN, S. C. J. **Computer Numerical Control: From Programming to Networking.** Delmar Publishers, 1994.

[33] MORRIS, H. M. **Robot Programming Goes Off-Line.** Control Engineering, p.143-145, setembro, 1985.

[34] MOTTA, G. D. **A Engenharia Virtual é Realidade**. CADware Technology. n.5, p.29-33, 1997.

[35] NETO, M. S. **Robótica Industrial.** Pontifícia Universidade Católica do Rio de Janeiro, Publicação Interna, 1994.

[36] PARENT, M. & LAURGEAU, C. **Logic and Programming.** Robot Technology, v.5, p.153-182, Prentice-Hall, Inc. 1983.

[37] PEDRA, A. **Sistemas CAD: Tendências Tecnológicas.** CADware Technology. n.5, p.32-34, 1997.

[38] POLLMANN, W. & DZEMBRITZKI, H. **Off-Line Programming of Industrial Robots.** Off-Line Programming of Industrial Robots. IFIP,1987.

[39] POWELL, J. D., FRANKLIN, G. F. & NAEINI, A. E. **Feedback Control of Dynamic Systems.** Addison-Wesley Publishing Company, Inc. 1994.

[40] PRINZ, M., LIU, H. C., NNAJI, B. O. & LUETH, T. **From CAD-Based Kinematics Modeling To Automated Robot Programming.** Robotics & Computer Integrated Manufacturing, v. 12, n.1, p.99-109, 1996.

[41] RÁNKY, P. G. & HO, C. Y. **Robot Modelling: Control Applications with Software.** Springer-Verlag, Berlin, Heidelberg, New York,Tokyo, 1985.

[42] REMBOLD, U., NNAJI, B. O. & STORR, A., **Computer Integrated Manufacturing and Engineering**. Addison-Wesley Publishing Company Inc., Great Britain, 1994.

[43] REZENDE, R. **ROBCAD – Simulação e Programação Off-Line de Robôs.** CADware Technology. n.5, p.76, 1997.

[44] ROGERS, D. F. & ADAMS, J. A. **Mathematical Elements for Computer Graphics.** McGraw-Hill Inc., 1990.

[45] ROSA, E. **O Uso de Polinômios Cúbicos de Hermite no Planejamento de Trajetórias de Manipuladores**. Tese de Doutorado do Departamento de Engenharia Mecânica, UFSC. Florianópolis, 1991.

[46] SATA, T., KIMURA, F., HIRAOKA, H., SUZUKI, H. & FUJITA, T. **Comprehensive Modelling of a Machine Assembly for Off-Line Programming of Industrial Robots.** Off-Line Programming of Industrial Robots. IFIP,1987.

[47] SCHROER,B. J. & TEOH, W. **A Graphical Simulation Tool with Off-Line Robot Programming.** Simulation, agosto 1986.

[48] SHIGLEY, J. E. & MISCHKE, C. R., **Mechanical Engineering Design**, McGraw Hill, 1989.

[49] SOLAND, E. .J. & VanDenBROEK, J. A. **Progress in CAD-tools for Robot Based Flexible Automation Systems.** 14th International Symposium on Industrial Robots. 7th International Conference on Industrial Robot Technology. Gothenburg, 1984.

[50] SPONG, M. W. & VIDYASAGAR, M. **Robot Dynamics and Control**, John Willey & Sons, 1989.

[51] STARK, J. **What Every Engineer Should Know About: Pratical CAD/CAM Applications**, Marcel Dekker Inc. , 1986.

[52] TEICHOLZ, E. **CAD/CAM Handbook.** McGraw Hill, 1985.

[53] TOLEDO, L. B., DIAS, A. & GUENTHER, R. - **Um Sistema CAD/CAM para a Programação Fora de Linha de Manipuladores,** V Congresso de Engenharia Mecânica Norte-Nordeste, Fortaleza, Ceará, Brasil - Out,1998. v.1, p.537-544.

[54] TOLEDO, L.B. & DIAS, A. - **Um Ambiente Interativo para a Geração de Trajetórias e Programação de Robôs Industriais**, 8° Congreso Chileno de Ingenieria Mecanica, Concecpcion, Chile, Out. 1998.

[55] WALKER, M. W. **Kinematics and Dynamics.** Handbook of Industrial Robotics. John Wiley & Sons, 1985.

[56] YONG, Y. F., GLEAVE, J. A., GREEN, J. L. & BONNEY, M. C. **Off-Line Programming of Robots.** Handbook of Industrial Robotics. John Wiley & Sons, 1985.

[57] ZISKOVSKY, J. P. **Robotics – The First Step to CIM and the Factory of the Future.** Robotics and Factories of the Future. Berlin 1984.

[58] **Initial Graphics Exchange Specification (IGES), Version 2.0** ; U.S. Departament of Commerce, National Bureau of Standards, February 1983.

[59] **RoboTalk Version 3.81: User's Manual.** Rhino Robots Inc. January 1992.

[60] **The student edition of MATLAB - version 4: Building a Graphical User Interface**, The MathWorks Inc., 1994.

[61] **The Wiz Worx HomePage**: The Source for IGES Tools. http://www.wiz-worx.com

[62] FERNANDES, C. J. **Programação Gráfica do Manipulador Scara.** Monografia, UFSC, 1998.

[63] FERNANDES, C. J. **Definições dos Principais Módulos da Linguagem Xoberon** Apostila, UFSC, 1998.

APÊNDICE 1

A MODELAGEM DE CURVAS "B-SPLINES"

A1.1 INTRODUÇÃO A MODELAGEM DE CURVAS

Na robótica são necessários métodos para computar uma trajetória num espaço multidimensional que descreve o movimento requerido do manipulador. Esta trajetória refere-se à história no tempo das posições, velocidades e acelerações de cada grau de liberdade do manipulador. Inicialmente este problema inclui a interface humana de como especificar os pontos de controle para determinar a forma de uma curva que se deseja obter. A partir destes pontos, como mencionado na seção 3.5.2.3, uma variedade de métodos de modelagem de curvas pode ser proposta para descrever a trajetória no espaço, seja esta no espaço cartesiano ou de juntas do manipulador.

Basicamente, porém, duas linhas podem ser tomadas para a geração de trajetórias a partir dos métodos de modelagem de curvas: a interpolação ou aproximação deste percurso de pontos de controle. A figura A1.1, a seguir, ilustra os resultados obtidos para a geração de trajetória a partir de dois diferentes métodos. No primeiro método, realiza-se uma aproximação do percurso de pontos de controle através de curvas "B-Splines" de quarta ordem, conseqüentemente, grau 3. No segundo método realiza-se a interpolação do percurso de pontos através de curvas "B-Splines" cúbicas.

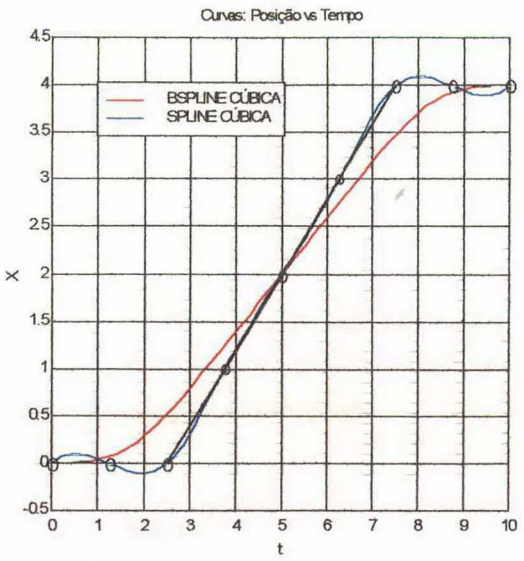

FIGURA A1.1: Curvas de Aproximação e de Interpolação.

No primeiro método a curva aproxima-se dos pontos intermediários, e passa somente pelos pontos de controle inicial e final.

Já no segundo método, a curva passa por todos os pontos de controle.

Neste trabalho estamos particularmente interessados no estudo do uso da formulação de curvas "B-Splines", por considerar que esta constitui um método em potencial para a geração de trajetórias de movimentação e precisão de manipuladores devido às suas diferentes características e propriedades citadas neste apêndice e no corpo principal do trabalho. A seguir esta formulação será mais detalhada.

A1 .2 A FORMULAÇÃO DE CURVAS "B-SPLINES"

Nesta seção será feito um breve apanhado das características e propriedades das formulações de curvas "B-Splines", com intuito de permitir um melhor entendimento destas. Adicionalmente as referências [44], [07] e [17] podem ser consultadas para um maior aprofundamento da matéria.

Tomando $P(t)$ como o vetor de posição ao longo da curva como uma função do parâmetro **t**, uma curva "B-Spline" é dada por

$$P(t)=\sum_{i=1}^{n+1} B_i N_{i,k}(t) \qquad t_{\min} \le t < t_{\max} \qquad 2 \le k < n+1_{\max} \qquad \text{(A1.1)}$$

onde B_i são os vetores de posições dos $n+1$ vértices do polígono de definição e $N_{i,k}$ são as funções de base normalizada da "B-Spline". [44]

Para cada i função de base normalizada de ordem k (grau $k-1$), as funções de base $N_{i,k}(t)$ são definidas pela formula recursiva de Boor [44], dada por:

$$N_{i,1}(t)=\begin{cases} \text{se } x_i \le t < x_{i+1} & =1 \\ \text{senão} & =0 \end{cases} \qquad \text{(A.1.2)}$$

e

$$N_{i,k}(t)=\frac{(t-x_i)N_{i,k-1}(t)}{x_{i+k-1}-x_i}+\frac{(x_{i+k}-t)N_{i+1,k-1}(t)}{x_{i+k}-x_{i+1}} \qquad \text{(A.1.3)}$$

Os valores de x_i são os elementos de um vetor de referência que satisfaz a relação $x_i \leq x_{i+1}$. O parâmetro t varia de t_{min} à t_{max} ao longo da curva $P(t)$. A convenção 0/0 = 0 é adotada.

Formalmente uma curva "B-Spline" é definida como uma função "Spline" polinomial de ordem k desde que ela satisfaça as seguintes condições [44] :

- A função $P(t)$ é um polinômio de grau $k-1$ em cada intervalo $x_i \leq t < x_{i+1}$.
- A função $P(t)$ e suas derivadas de ordem 1, 2, ..., $k-2$ são contínuas ao longo de toda a curva.

Além das propriedades citadas na seção 4.4.2, algumas outras propriedades interessantes também são mencionadas:

- Cada função de base é positiva ou nula para todos os valores dos parâmetros, isto é, $N_{i,k} \geq 0$.
- Exceto para $k = 1$ cada função de base tem precisamente um valor máximo.
- A soma das funções de base "B-Splines" para qualquer valor do parâmetro t é igual a 1, ou seja, $\sum_{i=1}^{n+1} N_{i,k}(t) \equiv 1$.

As formulações de curvas "B-Splines" possuem a característica de ser não global, devido ao fato de que cada vértice do seu polígono de definição está associado a uma única função de base. Deste modo, cada vértice afeta a forma da curva apenas na faixa de parâmetros em que sua função de base é diferente de zero. Está formulação também permite que a ordem da função de base e, portanto, do grau da curva resultante, seja modificado sem alterar o numero de vértices do polígono de definição.

As equações A.1.2 e A.1.3 demonstram que a escolha do vetor de referência tem uma influência significante nas funções de base $N_{i,k}$ da "B-Spline" e conseqüentemente na curva "B-Spline" resultante. A única restrição do vetor de referência é que este satisfaça a relação $x_i \leq x_{i+1}$; ou seja, seja uma série de números monotonicamente crescente.

Fundamentalmente três tipos de vetor de referência são usados: uniforme, aberto e uniforme (aberto) e não uniforme.

Num vetor de referência uniforme, os valores individuais de referência são igualmente espaçados. Exemplos deste são [0 1 2 3 4 5] e [-0.2 –0.1 0 0.1 0.2].

O vetor de referência aberto e uniforme tem multiplicidade nos valores de referência extremos igual a ordem k da função de base da "B-Spline". Os valores internos são igualmente espaçados. Alguns exemplos são: k=2 [0 0 1 2 3 4 4]; k=3 [0 0 0 1 2 3 3 3];

Formalmente, um vetor de referência uniforme e aberto é dado por [44]:

$$\begin{aligned} x_i &= 0 && 1 \leq i \leq k \\ x_i &= i-k && k+1 \leq i \leq n+1 \\ x_i &= n-k+2 && n+2 \leq i \leq n+k+1 \end{aligned}$$

O vetor de referência não uniforme tem valores de referência internos diferentemente espaçados e/ou multiplicidade nestes. Estes podem ser periódicos ou abertos. Como exemplos pode-se citar: [0 0 0 1 1 2 2 2] [0 1 2 2 3 4]

Como a fórmula de Cox de Boor (A1.2 e A1.3) usada para calcular as funções de base da "B-Spline" possui uma relação recursiva, uma função de base de uma dada ordem k depende desde as funções de base de ordem superior até a ordem 1. Para uma dada função de base $N_{i,k}$ esta dependência forma um padrão triangular dado por:

$N_{i,k}$

$N_{i,k-1}$ $N_{i+1,k-1}$

$N_{i,k-2}$ $N_{i+1,k-2}$ $N_{i+2,k-2}$

: : : ...

$N_{i,1}$ $N_{i+1,1}$ $N_{i+2,1}$ $N_{i+3,1}$ .. $N_{i+k-1,1}$

As derivadas de uma curva "B-Spline" em qualquer ponto na curva são obtidas pela diferenciação formal. Especificamente a partir da equação (A1.1) é obtida a primeira derivada

$$P'(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t) \tag{A1.4}$$

e a segunda derivada

$$P''(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t) \tag{A1.5}$$

com relação ao parâmetro t.

As derivadas das funções de base também são obtidas pela diferenciação formal da equação (A1.3). A derivada primeira da função de base é obtida

$$N'_{i,k}(t) = \frac{N_{i,k-1}(t) + (t - x_i)N'_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N'_{i+1,k-1}(t) - N'_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}} \tag{A1.6}$$

bem como a derivada segunda

$$N''_{i,k}(t) = \frac{2N'_{i,k-1}(t) + (t - x_i)N''_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N''_{i+1,k-1}(t) - 2N'_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}} \tag{A1.7}$$

A metodologia proposta neste trabalho integra esta formulação para a obtenção das trajetórias a serem executadas pelos manipuladores que se deseja controlar. O parâmetro t torna-se a variável tempo, de modo a permitir gerar trajetórias de posição, velocidade e aceleração, através da aproximação "B-Spline" e suas derivadas.

As Figuras A1.2 e A1.3 ilustram dois diagramas esquemáticos dos dados de entrada e saída do gerador de trajetórias utilizado na metodologia proposta.

No primeiro diagrama, tem-se a geração de trajetórias cartesianas e de orientação do efetuador do manipulador a partir dos vetores de posições Xp, Yp e Zp e orientações ϕp, φp e ψp, correspondentes ao pontos do percurso, para aproximação com as curvas "B-Splines" de ordem k e vetor de referência t. Como dados de saída do gerador de trajetórias temos as posições, velocidades e acelerações das coordenadas de posição e orientação do efetuador do manipulador. Neste caso deve-se realizar a cinemática inversa das posições, velocidades e acelerações obtidas para que se possa obter as trajetórias de juntas.

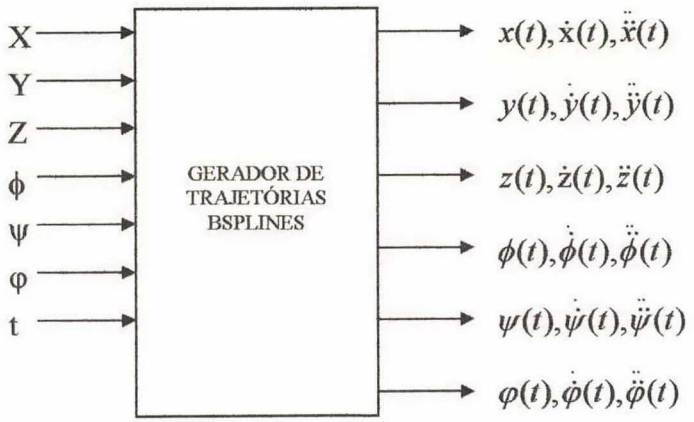

FIGURA A1.2 : Dados de Entrada e Saída do gerador de trajetórias "B-Splines" de posição e orientação para programação de manipuladores.

Já no segundo diagrama, deve-se realizar a cinemática inversa das posições e orientações do efetuador do manipulador para obter anteriormente os valores de juntas q_1, q_2, q_3, q_4, q_5 e q_6 correspondentes. Estes valores e o vetor de referência t são os dados de entrada do gerador de trajetórias, que processa a aproximação com as curvas "B-Splines" para obtenção das posições, velocidades e acelerações no tempo das juntas do modelo de manipulador utilizado.

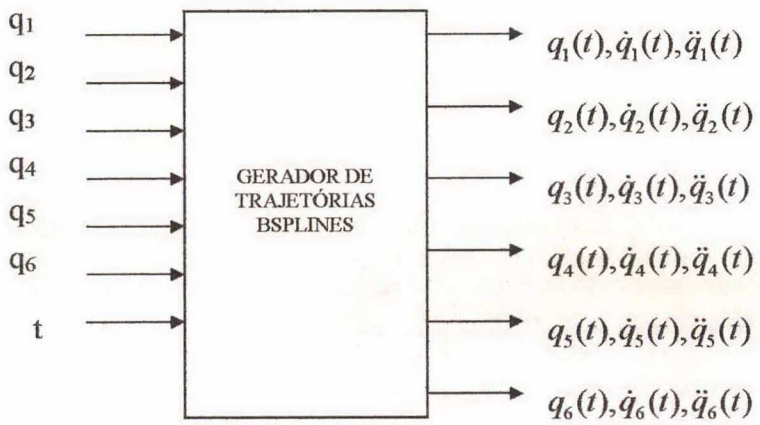

FIGURA A1.3 : Dados de Entrada e Saída do gerador de trajetórias "B-Splines" de posição e orientação para programação de manipuladores.

Esta formulação de curvas constitui numa das possibilidades a serem utilizadas para gerar as trajetórias cartesianas ou de juntas dos manipuladores. Esta é uma formulação cuja solução da trajetória não é colocada de uma forma explicita, e sim através de uma solução iterativa. Apesar disto, permite obter, em tempos de processamento relativamente rápidos, os dados necessários para a programação dos manipuladores. Esta formulação é satisfatória na programação fora de linha de manipuladores, pois podem ser geradas trajetórias de movimentação ou de precisão, descritas com funções suaves, de ordem elevada, e continuas na posição, velocidade e aceleração, e ainda, com possibilidades de controle da curva gerada com a manipulação dos parâmetros de definição destas.

A1.3 AS ROTINAS DO PACOTE DE "SPLINES" EM LINGUAGEM MATLAB

Com o pacote de rotinas em Matlab [07] é possível criar e trabalhar com funções e curvas polinomiais "piecewise", no ambiente interativo provido pelo Matlab.

Este pacote foi integrado através da interface desenvolvida com as rotinas GUI da linguagem Matlab [60], para geração das curvas pela aproximação "Bspline" de percursos de pontos.

Basicamente, para realizar a função do gerador de trajetórias ilustrado nas figuras A1.2 e A1.3 quatro rotinas do pacote foram utilizadas e estão ilustradas na figura A1.4.

```
spx=spmak(knots,Xp);
spy=spmak(knots,Yp);
spz=spmak(knots,Zp);
Xt=fnval(sp2pp(spx),t);
Yt=fnval(sp2pp(spy),t);
Zt=fnval(sp2pp(spz),t);
DXt=fnval(fnder(sp2pp(spx)),t);
DYt=fnval(fnder(sp2pp(spy)),t);
DZt=fnval(fnder(sp2pp(spz)),t);
DDXt=fnval(fnder(fnder(sp2pp(spx))),t);
DDYt=fnval(fnder(fnder(sp2pp(spy))),t);
DDZt=fnval(fnder(fnder(sp2pp(spz))),t);
```

FIGURA A1.4: Algoritmo com as rotinas utilizadas do pacote de Splines em Matlab para geração de curvas "Bsplines".

A variável knots é o vetor de referência, da equação A1.1. Xp, Yp e Zp são os vetores das posições cartesianas do percurso de pontos que devem ser aproximados. O parâmetro t é um vetor com valor inicial igual a zero e valor final igual ao tempo total que se deseja cumprir o percurso, assim como o vetor knots. Entretanto este vetor tem espaçamento entre valores conforme se deseja avaliar a função de aproximação. Por exemplo a cada 1ms.

A função spmak( ) é uma rotina do pacote de "Splines" que permite gerar uma função na forma Bspline, com o mínimo de informação. No caso, o vetor de referência knots e o vetor de dados cartesianos Xp, Yp ou Zp. A ordem k da "B-Spline" é computada pela relação k=length (knots)- length (Xp), onde a função length( ) retorna o tamanho do vetor em questão.

Com isso obtemos os parâmetros spx, spy e spz da função na forma "B-Spline", que são os dados de entrada para a função sp2pp( ), que os converte nos coeficientes da função polinomial "piecewise". Assim, finalmente, a função fnval( ) permite avaliar os valores ao longo do tempo t, e suas derivadas, através da função fnder( ).

APÊNDICE 2

A MODELAGEM DE SUPERFÍCIES DE BÉZIER

A2.1. INTRODUÇÃO A MODELAGEM DE SUPERFÍCIES

ROGERS [44] ressalta que: "As superficies e suas descrições desempenham um papel fundamental no projeto e na manufatura. A forma e a geometria da superficie são a essência do projeto tanto na sua funcionalidade quanto na sua estética".

O projeto e a manufatura de carrocerias de automóveis, de cascos de barcos, da fuselagem e asas de aviões são alguns exemplos óbvios. A descrição de superficies também tem um papel crucial na representação de dados obtidos em medicina, geologia, física e outros fenômenos naturais.

Na computação gráfica e nos sistemas CAD/CAM é vantajoso desenvolver um modelo matemático tridimensional real de uma superficie. Este modelo permite que as características da superficie, como por exemplo a curvatura e a normal a um ponto da superficie, sejam analisadas antecipadamente e de modo relativamente mais fácil. Este modelo também permite calcular quantidades fisicas dependentes da superficie, como por exemplo, o volume, a área, o momento de inércia, etc...

ROGERS[44] propõe que existem duas filosofias básicas envolvidas na técnica de descrição de superficies. A primeira, normalmente associada ao nome de "Cônicas", procura

criar uma superfície matemática a partir de dados conhecidos. A segunda, normalmente associada ao nome de Bézier, procura criar uma superfície matemática ***ab initio***.

Neste trabalho estamos interessados particularmente na segunda linha, que envolve as superfícies paramétricas, presentes na metodologia proposta. As superfícies paramétricas são muito utilizadas na computação gráfica e no projeto e manufatura auxiliado por computador. Exemplos de superfícies matemáticas paramétricas são as superfícies de revolução, as superfícies de varredura, as superfícies de "Bézier" e "Splines".

As duas últimas têm particularmente recebido atenção pois seu método de geração permitir modelar superfícies que não tenham nenhuma descrição analítica conhecida. Com a formulação de superfícies "Splines" é possível modelar entidades simples, como uma esfera, um plano ou uma superfície de revolução, como também complexas, por exemplo uma garrafa, um casco de barco ou mesmo um corpo humano.

Na integração desenvolvida neste trabalho para a programação de manipuladores, as superfícies paramétricas tem uma função determinante para a simplificação do ciclo de processamento de informações necessárias. A seguir serão detalhadas algumas metodologias de modelagem de superfícies paramétricas que são aplicáveis na metodologia proposta.

A2.2. A MODELAGEM DE SUPERFÍCIES PARAMÉTRICAS

O método proposto de programação, apresentado no capítulo 4, utiliza a descrição de modelos de superfícies paramétricas para gerar os pontos do percurso que deve ser aproximado para obtenção das trajetórias de posição, velocidade e aceleração que o manipulador deve cumprir.

Uma variedade de métodos de modelagem de superfícies parametricamente está presente na literatura especializada.

O método mais simples de se gerar uma superfície tridimensional consiste em rotacionar uma entidade bidimensional em torno de um eixo, como ilustrado na figura A2.1. Esta superfície é chamada de superfície de revolução. Um ponto sob uma superfície de revolução é especificado por dois parâmetros: s e ϕ.

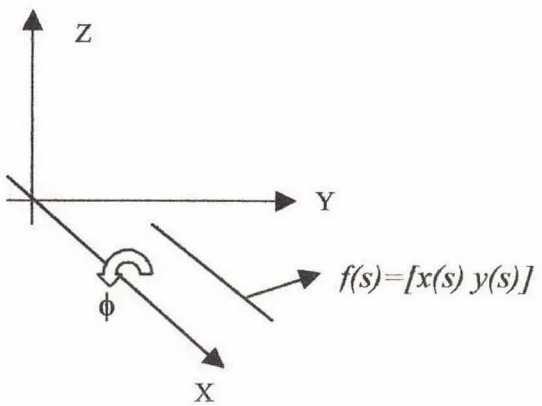

FIGURA A2.1: Parâmetros para modelagem de uma superfície de revolução.

Matematicamente, a equação de uma superfície gerada pela rotação de uma entidade (uma linha, ou uma curva, por exemplo) em torno do eixo x, entidade esta inicialmente sob o plano xy, tem a seguinte equação:

$$g(s,\phi) = [x(s) \quad y(s)\cos\phi \quad y(s)\operatorname{sen}\phi] \qquad \text{(A2.1)}$$

Outra entidade simples, que sua superfície pode ser facilmente descrita parametricamente, é a esfera. Uma esfera é gerada, por exemplo, rotacionando em torno do eixo x um semi-círculo centrado na origem do sistema de coordenadas pertencente ao plano xy. A figura A2.2 ilustra os parâmetros θ e ϕ e a constante r para a modelagem de uma superfície parametricamente

A partir da equação do semi-círculo, dada por

$$x = r\cos\theta \quad 0 \le \theta \le \pi \tag{A2.2}$$

$$y = r\,\mathrm{sen}\,\theta$$

a equação paramétrica da esfera é obtida

$$g(\theta,\phi) = [x(\theta) \quad y(\theta)\cos\phi \quad y(t)\,\mathrm{sen}\,\phi] \tag{A2.3}$$

$$= [r\cos\theta \quad r\,\mathrm{sen}\,\theta\cos\phi \quad r\,\mathrm{sen}\,\theta\,\mathrm{sen}\,\phi]$$

$$0 \le \theta \le \pi \quad 0 \le \phi \le 2\pi$$

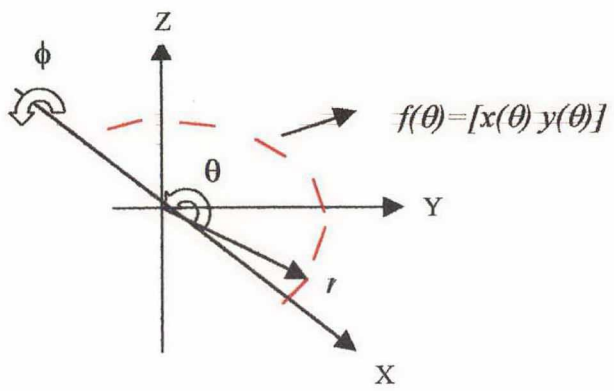

FIGURA A2.2: Parâmetros para modelagem de uma superfície esférica parametricamente.

Um método de modelagem de superfícies mais elaborado são as superfícies de Bézier, que são matematicamente descritas por [44]:

$$g(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,j} J_{n,i}(u) K_{m,j}(v) \tag{A2.4}$$

onde $J_{n,i}(u)$ e $K_{m,j}(v)$ são as funções de base de Bernstein nas direções dos parâmetros u e v.

As funções de base $J_{n,i}(u)$ e $K_{m,j}(v)$ são dadas por

$$J_{n,i}(u) = \binom{n}{i} u^i (1-u)^{n-i} \tag{A2.5}$$

$$K_{m,j}(v) = \binom{m}{j} v^j (1-v)^{m-j} \tag{A2.6}$$

com

$$\binom{n}{i} = \frac{n!}{i!(n-i)!} \tag{A2.7}$$

$$\binom{m}{j} = \frac{m!}{j!(m-j)!} \tag{A2.8}$$

onde $B_{i,j}$ é uma rede poligonal de definição. Os índices n e m são iguais a um a menos que o número de vértices do polígono nas direções u e v, respectivamente.

As figuras A2.3 e A2.4, a seguir, ilustram, respectivamente, uma possível rede poligonal de definição e a superfície de Bézier gerada a partir desta. As superfícies de Bézier são superfícies paramétricas, assim como as superfícies de revolução e de varredura, porém de descrição matemática mais complexa.

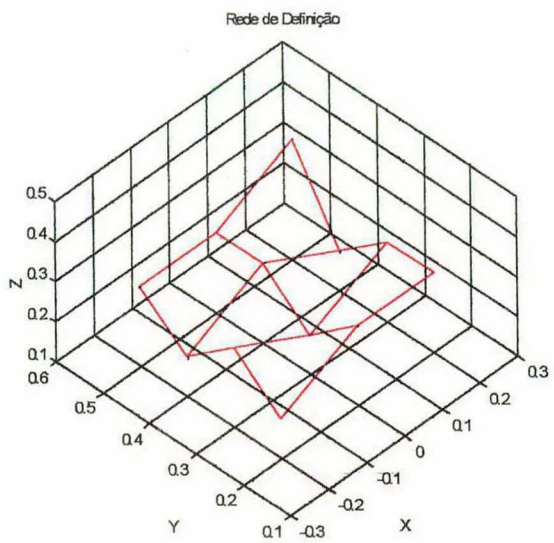

FIGURA A2.3: Rede de definição de uma superfície de Bézier.

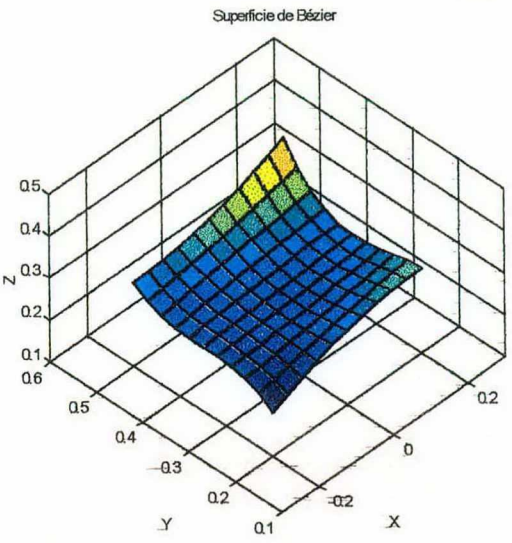

FIGURA A2.4: Visualização da superfície de Bézier gerada a partir da rede de definição da figura A2.2.

Algumas propriedades importantes destas superfícies podem ser citadas:

- O grau da superfície em cada direção paramétrica é igual a um menos que o número de vértices do polígono de definição na direção.
- A continuidade da superfície em cada direção paramétrica é de grau dois menos que o número de vértices na direção.
- A superfície geralmente segue a forma da rede poligonal de definição.
- Apenas os pontos das bordas da rede poligonal de definição são coincidentes com a superfície.
- A superfície é contida na rede poligonal de definição pela regra convexa
- A superfície é invariável sob uma transformação afim

Como detalhado na seção 4.2 onde foi apresentada a metodologia proposta, a técnica de geração de trajetórias para execução pelo manipulador utiliza um percurso de pontos nos parâmetros de uma superfície paramétrica para definir um percurso cartesiano 3D. Portanto, o primeiro propósito de manipulação do modelo consiste em obter os vetores de posição *Xp*, *Yp* e *Zp* do percurso de pontos cartesiano 3D, a partir dos vetores *Up* e *Vp* do percurso de pontos nos parâmetros do modelo matemático da superfície em questão. Já o segundo propósito consiste em permitir visualizar o modelo da superfície em questão no monitor do computador.

Outras propriedades e características da superfície de Bézier podem ser revisadas na referência [44], assim como as descrições matemáticas das superfícies "Splines", que são modelos paramétricos também mais elaborados e aplicáveis na metodologia proposta, e de grande interesse no uso de sistemas CAD/CAM atualmente.

A2.3. CÓDIGO EM LINGUAGEM MATLAB PARA MODELAGEM DE SUPERFÍCIES DE BÉZIER

No programa desenvolvido em linguagem MatLab, alguns modelos matemáticos de superfícies paramétricas são suportados pelo sistema. Dentre estes, foi desenvolvida uma rotina para descrição matemática das superfícies de Bézier.

A figura A2.5, a seguir, ilustra o código em linguagem MatLab para obtenção de um percurso cartesiano de pontos sob uma superfície de Bézier. Como dados de entrada da rotina desenvolvida no programa, tem-se os vetores de dados Up e Vp, que é o percurso de pontos nos parâmetros u e v da superfície de Bézier, e as matrizes de dados Xnet, Ynet e Znet, que definem os pontos cartesianos da rede poligonal de definição $B_{i,j}$ da superfície.

```
function [Xp,Yp,Zp]=BzrSrfPt(Up,Vp,Xnet,Ynet,Znet)
N=[ ]; M=[ ]; U=[ ]; V=[ ];
[uc vc]=size(Xnet);
u=uc-1; v=vc-1;
for j=0:u
   for i=0:u
      if (i+j)<=u
         N(i+1,j+1)=[fat(u)/(fat(j)*fat(u-j))]*(fat(u-j)/(fat(u-i-j)*fat(i)))*(-1)^(u-i-j);
      else
         N(i+1,j+1)=0;
      end;
      U(j+1)=1;
  end;
end;
for j=0:v
   for i=0:v
      if (i+j)<=v
         M(i+1,j+1)=[fat(v)/(fat(j)*fat(v-j))]*(fat(v-j)/(fat(v-i-j)*fat(i)))*(-1)^(v-i-j);
      else
         M(i+1,j+1)=0;
      end;
      V(j+1)=1;
   end;
end;
np=length(Up);
for l=1:np
         Xp(l)=U*N*Xnet*M*V';
         Yp(l)=U*N*Ynet*M*V';
         Zp(l)=U*N*Znet*M*V';
end;
```

FIGURA A2.5: Código em MatLab para geração de percursos de pontos cartesianos a partir de superfícies de Bézier, segundo equações A2.4 a A2.8.

Como dados de saída temos os vetores *Xp*, *Yp* e *Zp* que representam os valores cartesianos do percurso de pontos 3D.Deste modo, no programa podem ser obtidos diferentes percursos, partindo de diferentes modelos de superfícies, determinando a flexibilidade do método de geração de percursos de pontos cartesianos 3D.

APÊNDICE 3

CAD/CAM E O CONTROLE NUMÉRICO

A3.1. INTRODUÇÃO

O controle numérico ("NC – Numerical Control") conquistou espaço no meio industrial por ser um modo efetivo para incrementar a produtividade e aumentar a qualidade dos produtos. A automatização do planejamento e da programação das atividades de sistemas de produção com o uso das técnicas de CAD/CAM foi inicialmente empregada no campo do controle numérico. Foram gastos esforços durante um tempo considerável, e um corpo significante de conhecimento foi construído nesta área. As tecnologias NC e CNC possuem um alto nível de sofisticação para automatização, apoiada principalmente no progresso das tecnologias de hardware e de software.

Considerando a importância do conhecimento das técnicas, já consagradas, de CAD/CAM no controle numérico, abordaremos este assunto com intuito de aprofundar a matéria em dissertação. Serão brevemente revisados alguns conceitos e características do processamento de informações nos processos que utilizam máquinas ferramenta numericamente controladas, especificamente do seu planejamento e da sua programação.

A3.2 O PLANEJAMENTO DO PROCESSO NC

Como comentado na seção 3.3, o esforço de desenvolvimento e integração dos sistemas CAD/CAM visa auxiliar o processamento de informações necessários ao projeto do produto e planejamento da manufatura. Na figura A3.1 tem-se um diagrama bem definido do fluxo de informações dos processos NC, apresentado por LIN [32], onde estão destacadas as funções que devem ser coordenadas e requeridas para o sucesso de qualquer operação NC.

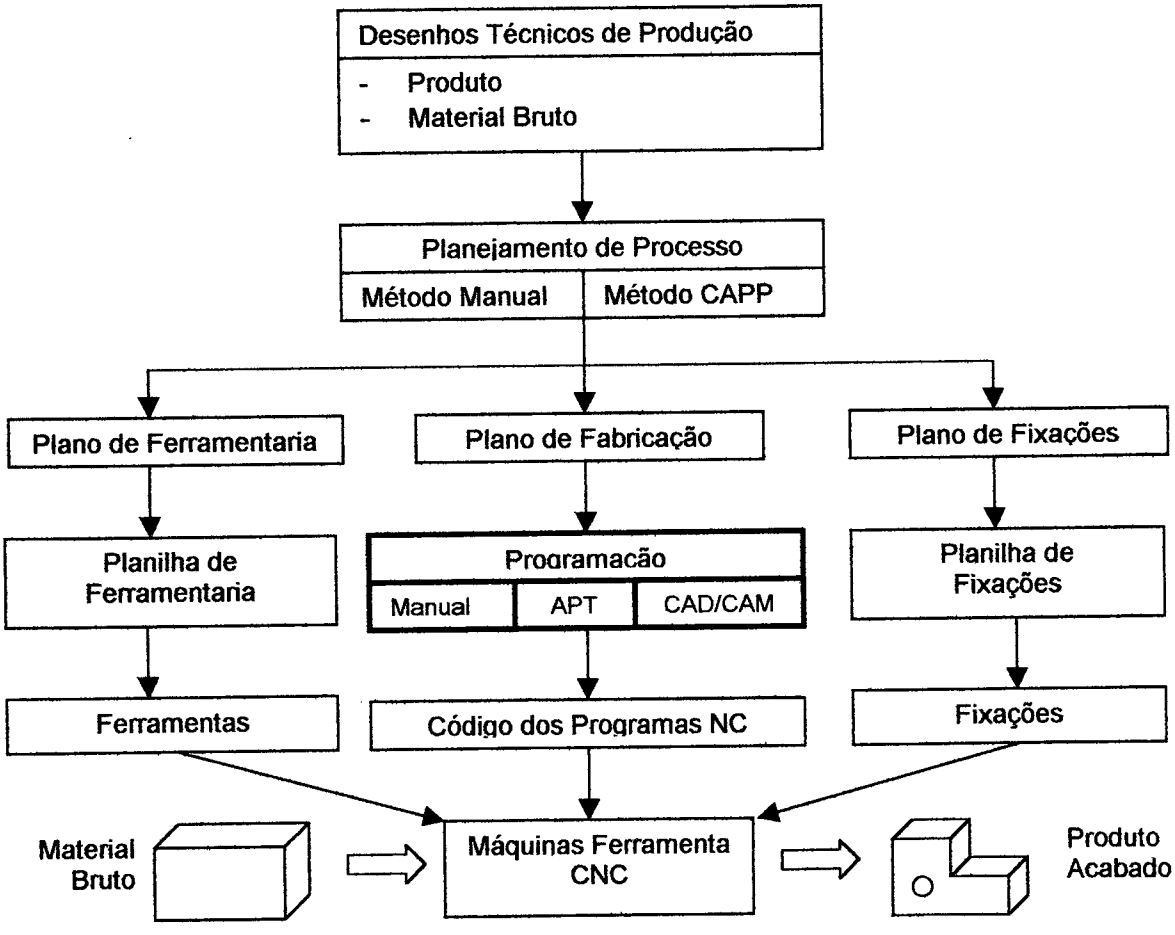

FIGURA A3.1: Diagrama de Fluxo dos Processos NC [32].

Após o projeto do produto, o estudo do desenho técnico da peça é o primeiro passo para o planejamento do processo NC. Neste, identificam-se as características da peça e as dimensões críticas, em coordenadas medidas com relação a um ponto de referência, para a futura

programação do percurso da ferramenta. O próximo passo consiste em examinar o desenho técnico do material bruto, que fornece a informação da peça antes que se inicie o processo de fabricação. Em seguida, utiliza-se as característica da peça para especificar os processos de fabricação e as seqüências de operação, bem como os tipos de ferramentas de corte, os percursos das ferramentas e os dispositivos de fixação a serem utilizados em certas condições de fabricação (velocidade de corte, taxa de alimentação, profundidade de corte). As planilhas de fixação e de ferramentaria são utilizadas no chão de fábrica para preparar as ferramentas e dispositivos de fixação necessários.

Antes que o programador inicie a programação, um plano de processo deve ser desenvolvido para esquematizar todos os passos necessários para a fabricação da peça.

LIN [32] define o plano de processo como: "um mapa para as operações NC. O ***planejamento de processo*** é o ato de preparar detalhadamente o plano de operação para transformar uma peça, a partir do material bruto, em uma forma acabada final. Trata-se do procedimento de determinação dos processos que devem ser usados, as seqüências destes, com as devidas ferramentas e dispositivos de fixação. O planejamento do processo é o mecanismo que faz a ligação entre o projeto de engenharia e a fabricação no chão de fábrica".

Segundo LIN [36], as funções presentes no planejamento de processo são:

1. Preparação do Material Bruto;
2. Seleção dos Processos;
3. Sequenciamento dos Processos;
4. Seleção dos Parâmetros de Fabricação;
5. Planejamento do Percurso da Ferramenta;
6. Seleção das Máquinas;
7. Seleção das Ferramentas;
8. Seleção das Fixações.

Trata-se de um processo sofisticado que requer um significativo conhecimento e experiência por parte de quem o desenvolve. Dentre as qualificações destacadas por [32], incluem-se:

1. Habilidade de interpretação de desenhos técnicos e tolerâncias geométricas;
2. Familiaridade com os processos de fabricação;
3. Familiaridade com três principais parâmetros de fabricação: velocidade de corte, taxa de alimentação e profundidade de corte;
4. Habilidade de selecionar as ferramentas de corte apropriadas;
5. Habilidade de selecionar os métodos e os dispositivos de fixação;
6. Familiaridade com análises de custo.

Atualmente existem duas aproximações para realizar a tarefa do planejamento de processo: planejamento de processo manual e planejamento de processo assistido por computador ("CAPP – Computer Aided Process Planning").

A3.3 O PLANO DE FABRICAÇÃO

LIN [32] define o plano de fabricação como: "um plano de operação que delineia todos os passos e informações necessários para produzir uma peça. Este é o núcleo do Plano de Processo". O autor ainda apresenta um diagrama do fluxo das informações referentes à atividade de planejamento do processo NC e preparação da sua documentação, como ilustrado na figura 3.12.

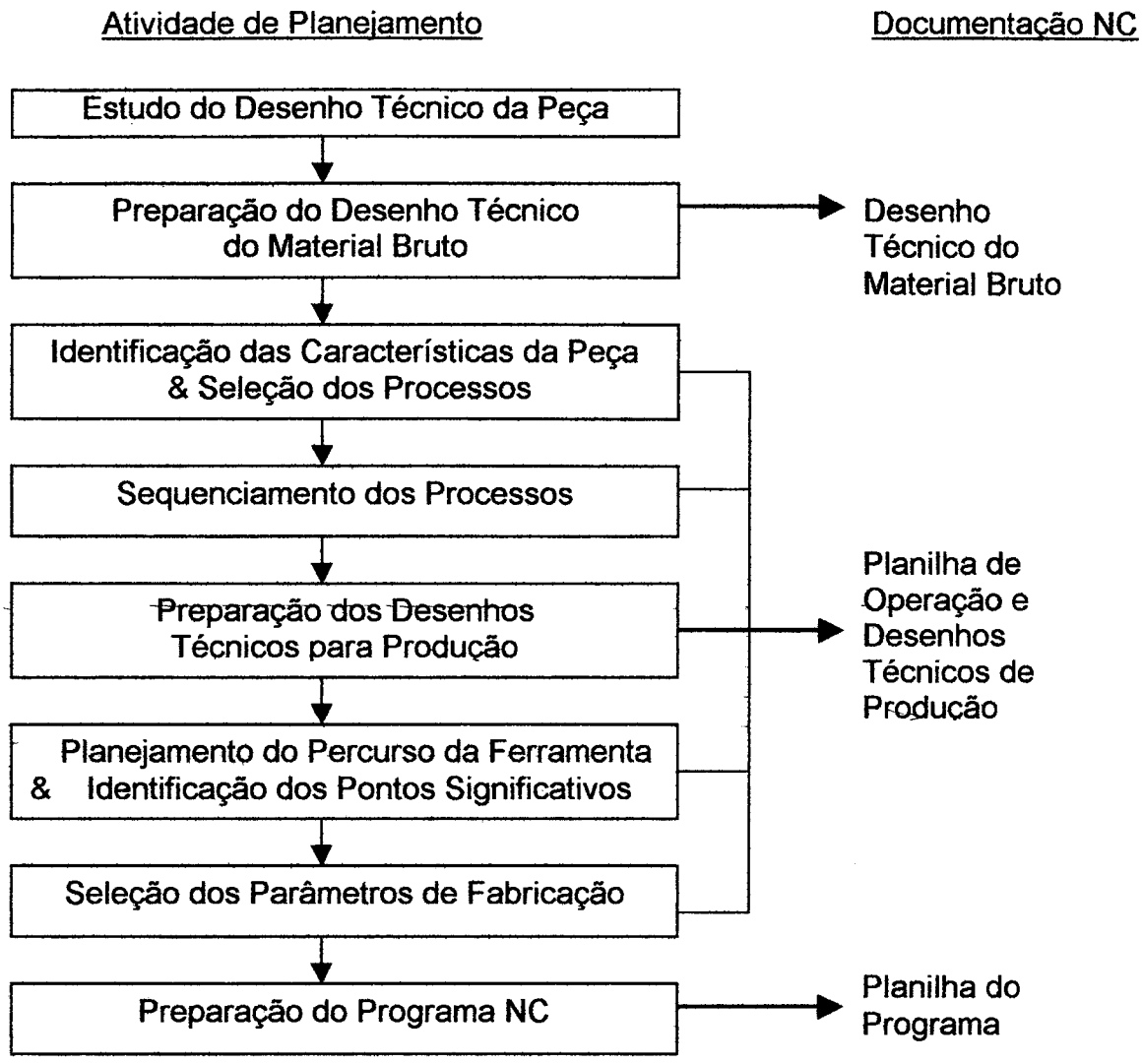

FIGURA A3.2: Plano de Fabricação NC [36]

O desenho técnico é uma definição pictorial da forma da peça de trabalho e de suas dimensões. Este é a base para a programação NC, e.que provê as seguintes definições com respeito a peça:

1. Forma básica e tamanho da peça de trabalho;
2. Envelope externo da peça;
3. Características da peça (formas) a serem produzidas;
4. Tolerâncias dimensionais e geométricas;
5. Acabamentos requeridos nas superfícies;

6. Superfícies de referência para ajuste e medição;
7. Material da peça de trabalho.

Dentre estas definições, a seleção de uma referência na peça é de extrema importância para o desenvolvimento do seu desenho técnico pois consiste na base dos trabalhos da produção. Esta referência deve ser estabelecida numa característica ou superfície da peça, podendo ser um ponto, uma linha ou uma superfície, a partir da qual as medidas são realizadas. A análise da peça e de sua função revelará quais as referências importantes.

Segundo LIN [32], o desenho técnico da peça revela sua forma básica e seu tamanho. Este também fornece o envelope externo da peça, ajudando na determinação da forma ótima do material bruto necessário à produção da peça acabada. As características do material bruto têm um significativo efeito no montante de material a ser removido, bem como na facilidade de fixação deste e na eficiência de fabricação da peça. A seleção apropriada de um processo para a produção de uma peça em uma máquina ferramenta NC é uma tarefa complicada. A seleção imprópria do processo de fabricação de uma peça pode resultar em (1) custos altos de produção, (2) precisões dimensionais inaceitáveis, e (3) tempo de vida da ferramenta reduzido.

Muitos fatores devem ser considerados na seleção de um processo de fabricação, dentre estes LIN [32] sugere: (1) as características ("features") da peça, (2) a precisão e a tolerância dimensional e geométrica requerida, (3) o acabamento superficial requerido, (4) os recursos disponíveis, incluindo as máquinas NC e as ferramentas de corte, e (5) o custo.

As características da peça são de importante relevância. Uma característica constitui uma forma geométrica distinta a ser produzida a partir do material bruto, e vai determinar os tipos de processos, os tipos de ferramentas, a máquina ferramenta necessária (3, 4, ou 5 eixos) e o percurso da ferramenta. Existem características básicas, como furos, perfis e ângulos, e características compostas, que se constituem de duas ou mais características básicas.

Dentre as considerações importantes, comentadas por LIN [32], para determinação da seqüência de operações, destaca-se a necessidade de serem prevenidas interferências entre as características da peça. Isto ocorre quando a fabricação de uma característica destrói os requerimentos para a produção de outra característica, configurando-se de uma interação ou dependência entre as operações de fabricação.

Após a determinação de uma seqüência de processos para a produção de uma peça, o programador NC prossegue com o planejamento dos percursos necessários das ferramentas para cada operação. O esboço do percurso da ferramenta não apenas demonstra onde esta deve se movimentar mas também permite checar interferências entre a ferramenta e os dispositivos de fixação. O operador da máquina ferramenta NC também pode utilizar o esboço do percurso da ferramenta para checar erros no programa e nos ajustes das fixações. Segundo LIN [32], em processos NC é essencial indicar claramente as posições iniciais e finais do percurso da ferramenta para cada operação, indicar movimentos rápidos e interpolados, e especificar a profundidade e o número de passos do percurso da ferramenta de corte.

A eficiência e a precisão são as duas mais importantes considerações em qualquer operação NC. Segundo o autor [32], "a precisão dimensional obtida na peça é determinada pela precisão do programa e a precisão da máquina. A precisão do programa deve ser assegurada através da programação sem erros das coordenadas de posição, enquanto precisão da máquina depende da qualidade do sistema de servo controle, dos dispositivos de medição e do sistema de acionamento, devendo estar dentro das especificações de resolução e de repetibilidade da máquina. Quanto à eficiência, está relacionada com a taxa de produção, ou tempo gasto na produção de uma peça. A programação de três parâmetros de fabricação – velocidade de corte, taxa de alimentação e profundidade de corte – dita a eficiência do processo de fabricação. A seleção destes três parâmetros deve ser realizada considerando-se a capacidade operacional da ferramenta de corte, a potência de giro da máquina e a rigidez da peça e das fixações".

A3.4 PREPARAÇÃO DOS PROGRAMAS NC

Assim como na robótica, em geral, um programa de instruções NC pode ser gerado de três maneiras diferentes, destacadas na figura 3.11: Entrada Manual Direta dos Dados na Máquina CNC, Programação Assitida por Computador e com o uso de softwares de CAD/CAM.

O primeiro método é eficiente para pequenos programas, mas torna-se não prático em programas extensos. No segundo método, o usuário utiliza linguagens como o APT e o COMPACT II na criação de programas NC para a fabricação de geometrias complexas usando algumas definições matemáticas. Já no terceiro método, uma ligação direta entre o projeto do produto e a fabricação é estabelecida utilizando um software de CAD/CAM. Os engenheiros de projeto de produto utilizam o sistema CAD para definir a geometria da peça, suas dimensões e tolerâncias. Os dados de projeto podem ser transferidos para o sistema CAM, permitindo aos programadores desenvolver os códigos para fabricar a peça.

Quando a programação de uma máquina ferramenta CNC é feita manualmente, a confiabilidade da usinagem somente é alcançada às custas de longos e caros procedimentos de tentativa e erro. Os atuais sistemas CAM realizam a programação de todos os tipos de usinagem CNC com mais rapidez e precisão, utilizando um modelo gerado em CAD e otimizando a trajetória da ferramenta. Os recursos de simulação gráfica da usinagem permitem a verificação de todas as operações envolvidas, através de uma animação tridimensional envolvendo a peça e as ferramentas de corte.

Em um sistema que utiliza pacotes separados de sistemas CAD e CAM, a transferência de dados da geometria do produto deve ser realizada. Nestes, a geometria é criada num sistema CAD e então transferida para o sistema CAM. Depois que a geometria da peça é criada, o programador especifica no programa CAM a ferramenta que deve ser utilizada. A informação detalhada da ferramenta deve ser dada, através de dados como o tipo de material, o

comprimento, o diâmetro, etc., devendo também ser providas as informações para o processo de fabricação e o percurso da ferramenta [32].

Já um sistema CAD/CAM integrado é um sistema dedicado que permite ao usuário criar a geometria do produto e gerar os percursos das ferramentas no mesmo pacote. A interface de dados entre os sistemas CAD e CAM não é necessária, o que faz com que não haja problemas de compatibilidade. Esta característica é importante já que assegura dados com mais precisão e confiabilidade. Como desvantagem deste sistema integrado, pode-se citar o fato de que este pode não ser flexível o bastante para modelar um produto com geometria complexa [32].

Este é um método relativamente novo desenvolvido pelos vendedores de sistemas CAD/CAM e está provado que constitui um jeito prático de gerar os programas NC. A maioria dos sistemas CAD/CAM dispõe da capacidade de verificar os programas gerados antes da execução dos mesmos, o que é possibilitado pelo uso de animações gráficas interativas para verificar os programas, ajudando na determinação do processo apropriado de fabricação e a prevenção de problemas de fabricação; tornando este um meio custo/efetivo de automatizar o processo de criação dos programas para máquinas ferramentas numericamente controladas [32].

As figuras 3.13 e 3.14, a seguir, ilustram algumas atividades de fabricação que podem ser planejadas e programadas diretamente a partir de modelos geométricos 2D e 3D, respectivamente. Neste método já consagrado de programação, tem-se então como partida inicial a geometria dos produtos e, com o uso de um programa gráfico interativo, auxilia-se o planejamento do processo e a definição do plano de processo em si, brevemente comentados anteriormente, de modo a definir as trajetórias das ferramentas a partir de alguns parâmetros disponíveis pré-definidos.

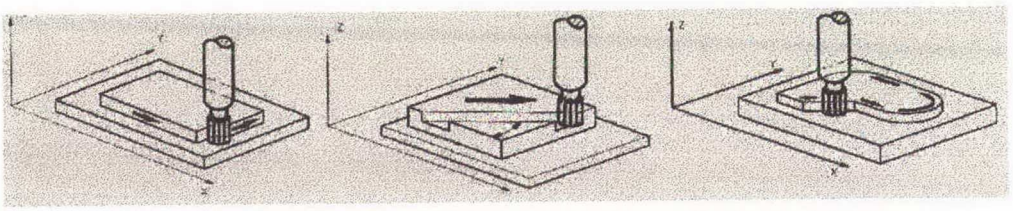

FIGURA A3.3: Percursos definidos em 2D para desbaste de peças.

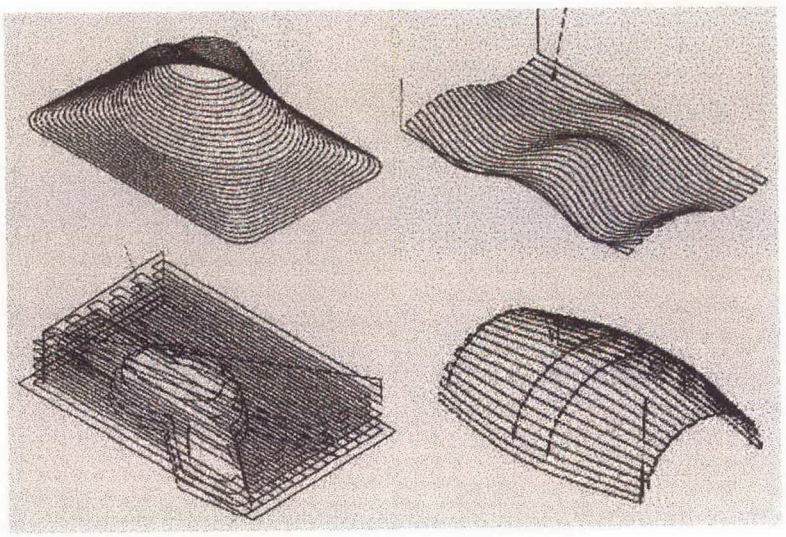

FIGURA A3.4: Percursos definidos em 3D para realização de atividades diversas.

A integração de sistemas resulta na diminuição do tempo total despendido desde a concepção até a sua fabricação. O conceito de protótipo virtual simulado em computador reduz o número de protótipos físicos a serem construídos para efeito de visualização, testes funcionais, verificação e validação do projeto. O custo e o tempo total de desenvolvimento do produto são drasticamente diminuídos.

Como destaca MOTTA [34], "de certo, a experiência acumulada, o raciocínio seletivo, a capacidade de julgamento e os conhecimentos técnicos e humanísticos do engenheiro jamais serão substituídos pelos mais avançados sistemas de "Engenharia Virtual" – pois uma simulação digital somente terá valor prático se os fenômenos reais estiverem bem representados e se os resultados forem interpretados adequadamente. Entretanto, a disponibilidade de poderosos

sistemas CAD/CAM torna obrigatória a utilização desta importante ferramenta por todas as empresas que pretendem ser competitivas dentro de um mercado onde produtividade, qualidade e redução de custos são exigências fundamentais".

APÊNDICE 4

O PLANEJAMENTO DA IMPLEMENTAÇÃO DE APLICAÇÕES DOS ROBÔS INDUSTRIAIS

ASFAHAL [04] apresenta um diagrama onde constam cinco fases que geralmente podem ser identificadas na implementação de uma aplicação de robótica, ilustradas na figura A4.1, e que ocorrem de uma maneira seqüencial.

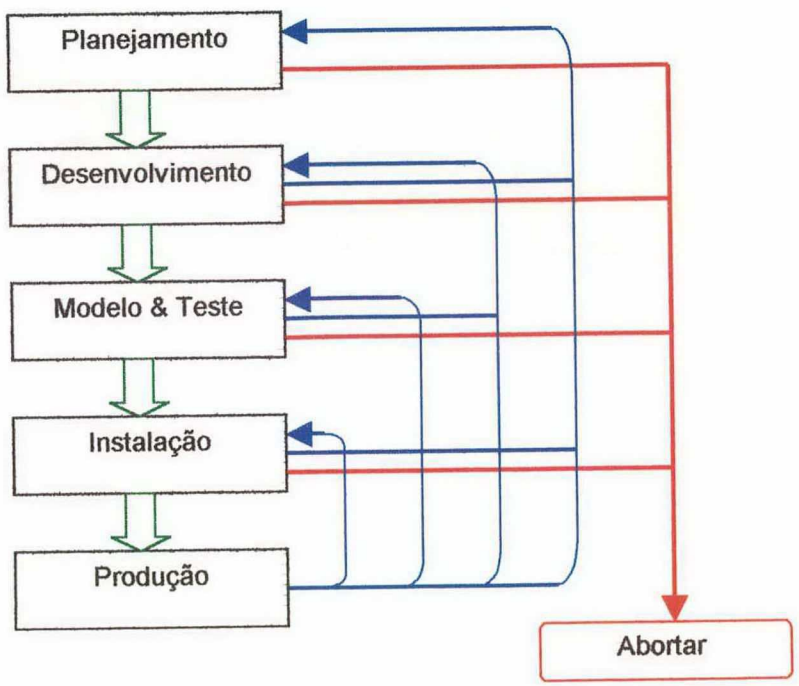

FIGURA A4.1: As fases identificáveis na implementação de aplicações industriais de robótica.

Neste diagrama tem-se como fase inicial o planejamento da aplicação. A demarcação entre o planejamento e o desenvolvimento não é clara, mas normalmente um sinal geral é a aparição de um protótipo de hardware. Isto se deve ao fato de que qualquer categoria de aplicação dos robôs industriais requer algum desenvolvimento em hardware. Segundo ASFAHAL [04], as mais importantes áreas de desenvolvimento de hardware consistem nas seguintes interfaces do robô: a interface do robô com o produto, a interface do robô com equipamentos, e a interface do robô com outras máquinas de processamento.

Para implementação de uma célula de trabalho, qualquer que seja a aplicação, estas cinco fases devem ser realizadas e coordenadas numa relação seqüencial. Em qualquer momento pode-se ter um retorno para uma fase anterior, ou mesmo reiniciar-se toda a seqüência. Segundo ASFAHAL [04], é impossível desprezar qualquer uma destas fases e obter um resultado satisfatório numa nova implementação de robótica. Estamos particularmente interessados na primeira fase de implementação de uma nova aplicação de robótica. Para melhor entendimento desta fase, considere o diagrama, ilustrado a seguir na figura 3.16, apresentado por ASFAHAL [04], onde consta um diagrama de blocos que representa as atividades que devem ser realizadas para o planejamento.

A primeira etapa da fase de planejamento para a implementação de um projeto de robótica constitui-se no ***Isolamento da Aplicação Potencial***. Com uso do conhecimento básico de robótica, pode-se usualmente identificar um bom candidato para implementação.

A fase de planejamento, então, prossegue com a ***definição dos objetivos*** em termos dos benefícios a serem atingidos. Objetivos robustos podem ser considerados, como o incremento de produtividade; a redução dos custos de uma tarefa ou a redução do ciclo de tempo de uma operação crítica; ou ainda, a substituição de um trabalho indesejável, ou perigoso, ou o aumento da qualidade de um produto.

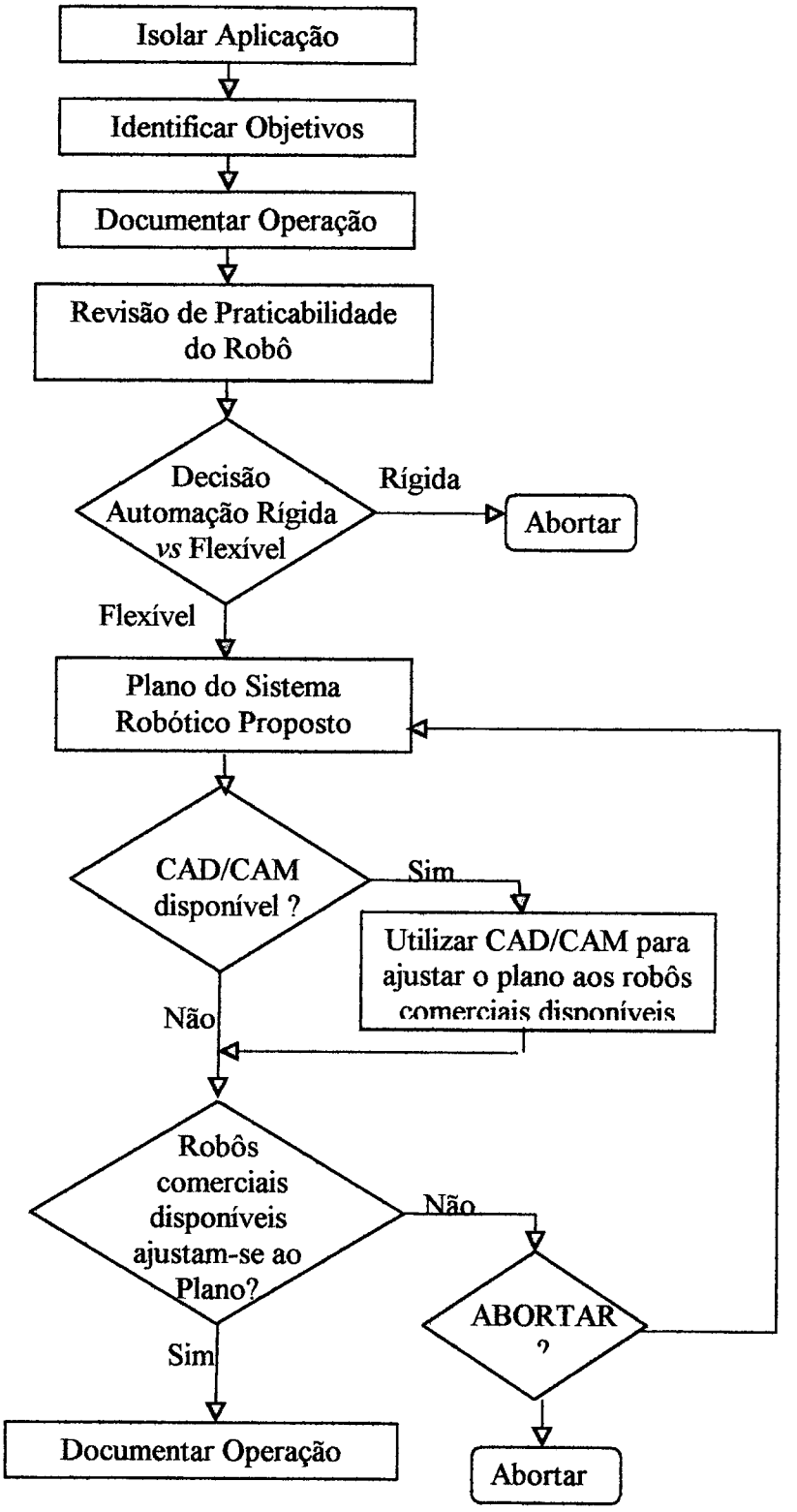

FIGURA A4.2: Diagrama de Blocos das Etapas de Planejamento de um Projeto para Implementação de Sistema Robótico.

Nesta etapa de definição do problema também devem ser considerados possíveis obstáculos à implementação. Por exemplo, a implementação de uma aplicação de robótica pode causar impacto nos trabalhadores, desde criar resistência nestes à nova realidade até tirar seus empregos. Considerações de segurança também devem ser feitas. Em muitas aplicações, a implementação de robótica pode ocasionar maior segurança, mas o caso contrário também é possível. Muitas outras considerações podem ser feitas, sendo obtida uma apreciação mais detalhada desta etapa em [04].

A próxima etapa consiste na ***documentação da operação atual***. Esta deve ser documentada cuidadosamente para que seja modificada. Estes dados serão valiosos para o detalhamento do projeto proposto para o sistema de automação e para sua eventual avaliação. Dentre estes dados, ASFAHAL [04] ressalta que devem ser tratados: a planta, os ciclos de tempo das máquinas, os tempos de operações manuais, bem como, os padrões de qualidade do produto, como por exemplo as tolerâncias dimensionais.

Uma vez completada a documentação da configuração existente da operação candidata, chega-se à etapa de se examinar as especificações de robôs comerciais existentes para determinar se estes podem ser utilizados no tipo de trabalho proposto na aplicação. Consiste, portanto, em um ***estudo de "praticabilidade" do robô,*** em termos técnicos, e não econômicos. Devem ser observadas, por exemplo, as cargas, as velocidades, a repetibilidade, e o controle do movimento.

Neste momento cumpre analisar as alternativas para os robôs. Estes são projetados com propósitos gerais e sistemas de controle reprogramáveis, havendo diversas possibilidades de projeto devido à sua flexibilidade. Contudo, a aplicação dos robôs industriais nem sempre é a solução menos onerosa. Deve ser decidida a opção entre a ***automação rígida ou flexível***.

Uma vez feita a decisão pela automação flexível, o próximo passo da fase de planejamento consiste em um estudo preliminar da planta do sistema, incluindo todos os equipamentos que irão interagir com o robô. O planejamento da planta de um sistema robótico

proposto consiste em um processo interativo, onde devem ser feitas tentativas para comparar as capacidades e envelopes de trabalho dos diferentes robôs comerciais. Neste momento, os sistemas de planejamento e simulação de células de trabalho têm um importante valor no planejamento, estando estes abordados de forma mais detalhada na seção 3.4.