CALIBRAÇÃO REMOTA DE SISTEMAS ROBÓTICOS UTILIZANDO SENSORES INTERNOS E EXTERNOS

Greicy Costa Marques

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

Prof. Fernando Cesar Lizarralde, D.Sc.

Prof. Liu Hsu, Docteur d'Etat

Prof. Raul Guenther, D.Sc.

RIO DE JANEIRO, RJ - BRASIL

JULHO DE 2005

MARQUES, GREICY COSTA

Calibração Remota de Sistemas Robóticos Utilizando Sensores Internos e Externos [Rio de Janeiro] 2005

XI, 104 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia Elétrica, 2005)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Robótica
2. Método de Newton
3. Calibração

I. COPPE/UFRJ II. Título (série)

Agradecimentos

A Deus, seu filho Jesus Cristo e Nossa Senhora pela proteção e força nos momentos de alegria e de dificuldade.

Aos meus Pais, Alfeu e Graci, por todo o incentivo que sempre me foi dado, pelo amor que têm por mim e por ajudarem a desenvolver meus valores, e principalmente por tudo que sempre fizeram e fazem por mim.

Ao meu querido Junior, que sempre esteve junto comigo nesta longa caminhada, dando-me força quando as coisas pareciam não se concretizar. E principalmente pela sua paciência e companheirismo.

Aos meus irmãos, a minha cunhada pelo apoio quando precisei. Aos meus sobrinhos Léo e Carol pelos momentos de descontração e carinho.

Aos queridos amigos Ana Regina, Jorginaldo e Dona Abigail por permitirem minha estadia em sua residência e por toda a atenção a mim dada. A Dona Olga pelo seu constante incentivo.

Ao meu orientador, Professsor Fernando Lizarralde, pela paciência, compreensão e orientação ao longo do desenvolvimento deste trabalho, e minha admiração.

À SUFRAMA, pela iniciativa pioneira de formar e qualificar para a Sociedade, profissionais nas mais diferentes áreas da Engenharia Elétrica.

À Prof. Marly Costa, pela coordenação do Mestrado na UFAM, que soube conduzir da melhor maneira possível as responsabilidades a ela conferida; Ao Prof. Ramon Costa pela coordenação na COPPE do Programa de Mestrado Inter-Institucional.

Ao Prof. Rubem César, coordenador Geral do CDEAM (Centro de Desenvolvimento Energético Amazônico), pela sua confiança e apoio para a conclusão deste trabalho.

Aos colegas do Labcon (laboratório de controle) por sua atenção e ajuda.

Aos meus amigos que de alguma forma contribuíram para realização deste trabalho.

As meninas da secretaria, Ana e Renata, que sempre se mostraram prestativas e atenciosas.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

CALIBRAÇÃO REMOTA DE SISTEMAS ROBÓTICOS UTILIZANDO SENSORES INTERNOS E EXTERNOS

Greicy Costa Marques

Julho/2005

Orientador: Fernando Cesar Lizarralde

Programa: Engenharia Elétrica

Neste trabalho é abordado o problema de replanejamento de trajetória de manipuladores robóticos. O replanejamento de trajetória é necessário quando existe incerteza na localização absoluta do manipulador com relação à localização original assumida para resolver o problema de planejamento de trajetória. Considerando que a incerteza é desconhecida, o replanejamento é baseado num método de calibração utilizando um algoritmo de mínimos quadrados. A incerteza é estimada utilizando a informação da configuração do manipulador numa grade de calibração conhecida utilizando sensores internos ou externos. A estimação desta incerteza pode ser realizada no espaço operacional ou no espaço de velocidades. Desta forma, o replanejamento é baseado na estimativa da incerteza e na trajetória ideal planejada. Os métodos propostos são validados via simulação utilizando o modelo cinemático do manipulador Zebra Zero e considerando ruído de medição.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

REMOTE CALIBRATION OF ROBOTIC SYSTEMS USING INTERNAL AND EXTERNAL SENSORS

Greicy Costa Marques

July/2005

Advisor: Fernando Cesar Lizarralde

Department: Electrical Engineering

In this work the problem of trajectory replanning of a robotic manipulator is considered. The trajectory replanning is necessary when uncertainties in the absolute localization of the manipulator with respect to the original localization assumed to solve the trajectory planning problem. Considering that the uncertainty is unknown the replanning is based on a calibration method using a least square algorithm. The uncertainty is estimated using the information of the configuration manipulator in a calibration grid known through internal and external sensors. The estimation of this uncertainnty can be carried in the operational space or the space speeds. This the replanning is based on the estimative of the uncertainty and the ideal planned trajectory. The proposed methods are validated by simulation using the kinematic model of the manipulator Zebra Zero and considering measurement noise.

Sumário

Lista de Figuras **ix**

Lista de Tabelas **xi**

1 Introdução **1**
1.1 Motivação do trabalho . . . 3
1.2 Estado da arte . . . 4
1.3 Apresentação do problema . . . 6
1.3.1 Metodologia . . . 9
1.3.2 Contribuição . . . 10
1.4 Organização do trabaho . . . 10

2 Programação *Off-line* e Planejamento de Trajetória **11**
2.1 Programação *off-line* . . . 12
2.1.1 Limitações da programação off-line . . . 13
2.2 Planejamento de trajetória . . . 15
2.2.1 Transformação homogênea . . . 16
2.2.2 Modelagem cinemática de manipuladores . . . 18
2.2.2.1 Ângulos de Euler . . . 19
2.2.2.2 Quaternions unitário . . . 21
2.2.3 Parâmetros de Denavit-Hatenberg . . . 23
2.2.4 Modelagem cinemática inversa . . . 26
2.2.5 Caminho e trajetória . . . 28
2.2.6 Trajetórias no espaço das juntas . . . 30
2.2.6.1 Movimento ponto a ponto . . . 30
2.2.6.2 Movimento do caminho . . . 32

3 Calibração e o Replanejamento de Trajetória **34**
3.1 Estimação do erro de posicionamento com sensor interno . . . 36
3.2 Algoritmo de calibração - sensores internos . . . 41
3.2.1 Simulação . . . 43
3.2.1.1 Manipulador robótico . . . 43
3.2.1.2 Grade de calibração . . . 44
3.2.2 Resultados de simulação . . . 46
3.3 Identificação do erro de posicionamento com um sensor visual . . . 54
3.4 Algoritmo de calibração - sensor externo . . . 60

3.4.1 Simulação 61
3.4.1.1 Modelo matemático 61
3.4.1.2 Parâmetros de Simulação 62
3.4.2 Resultados simulação 62
3.5 Replanejamento da trajetória 69
3.5.1 Algoritmo de cinemática inversa 69

4 Calibração e Replanejamento no Espaço de Velocidades 72
4.1 Planejamento do caminho 72
4.2 Abordagem da solução iterativa 74
4.3 Calibração e replanejamento no espaço de velocidades 76
4.3.1 Método iterativo 78

5 Conclusões 81
5.1 Propostas para trabalhos futuros 82

Apêndices 83

A Otimização 83
A.1 Métodos baseados em gradiente 86
A.1.1 Método de descida máxima 87
A.1.2 Método de Newton 89
A.2 Métodos globalmente convergentes 93
A.2.1 Método de Newton globalmente convergente 95
A.2.2 Método de Levenberg-Marquardt 97

B Sensores visuais 99

C Configuração de câmeras 101

Referências Bibliográficas 103

Lista de Figuras

1.1 Manipulador na localização original percorrendo uma trajetória desejada. 7
1.2 Manipulador deslocado em relação ao sistema de coordenadas inercial. 7
1.3 Esquema do problema em estudo. 8
1.4 Diagrama em bloco das etapas do trabalho. 9

2.1 Representação de um ponto P em diferentes sistemas de coordenadas. 16
2.2 Sistema de referência. 19
2.3 Representação dos Ângulos Z-Y-Z de Euler. 20
2.4 Rotação em torno de um eixo arbitrário. 21
2.5 Representação de um sistema de coordenadas de um robô. 24
2.6 Configuração de Denavit-Hartenberg. 25
2.7 Método para solução do modelo inverso. 26
2.8 Trajetória entre dois pontos. 28

3.1 Representação dos pontos da Grade de calibração em E_b e E_{b_1} e o erro de posicionamento T_{bb_1} 36
3.2 Definição Translacional e Rotacional do erro 37
3.3 Grade de Calibração - Representação dos pontos no sistema de coordenadas E_b. 45
3.4 Grade de Calibração - Representação dos pontos no sistema de coordenadas E_{b_1}. 45
3.5 Manipulador tocando os pontos da grade de calibração - Manipulador em E_b. 45
3.6 Manipulador tocando os pontos da grade de calibração - Manipulador em E_{b_1}. 46
3.7 Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, estimação sem ruído. 47
3.8 Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, para $N = 18$ pontos da grade de calibração. 49
3.9 Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, para $N = 10$ pontos da grade de calibração. 50

3.10 Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, para $N = 5$ pontos da grade de calibração. 50
3.11 Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, para $N = 3$ pontos da grade de calibração. 51
3.12 Medindo os pontos de calibração da grade, através de um sensor visual (câmera). 54
3.13 Manipulador deslocado em relação a E_b, erro de posicionamento estimado através do sensor visual. 55
3.14 Representação dos sitemas de coordenadas usados com sensor visual, estando o robô na base b. 56
3.15 Modelo de projeção perspectiva de uma câmera. 56
3.16 Resultado de simulação: Erro de posicionamento estimado com sensor visual, considerando uma situação ideal. 63
3.17 Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensor visual, para $N = 18$ pontos da grade de calibração. 64
3.18 Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensor visual, para $N = 10$ pontos da grade de calibração. 65
3.19 Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensor visual, para $N = 5$ pontos da grade de calibração. 65
3.20 Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensor visual, para $N = 3$ pontos da grade de calibração. 66
3.21 Um ponto λ no espaço nos sistemas de coordenada E_b e E_{b_1} 69
3.22 Diagrama em bloco do algoritmo para cinemática de controle cinemático com jacobiano inverso 71
3.23 Replanejamento de Trajetória 71

4.1 Esquema de replanejamento. 79
4.2 Ponto de calibração. 80
4.3 Sinal de controle de calibração. 80
4.4 Replanejamento de Trajetória. 80

B.1 Representação esquemática de um sistema de visão. 100

C.1 Configuração câmera na mão (eye-in-hand) e eixos de coordenadas. . . 102
C.2 Configuração câmera fixa no espaço de trabalho eixos de coordenadas (câmera, efetuador, e objeto). 102

Lista de Tabelas

3.1 Parâmetros de Denavit-Hatenberg para o Manipulador Zebra Zero. . . 43
3.2 Pontos da grade de calibração. 44
3.3 Parâmetros de posição e orientação do erro de posicionamento. 47
3.4 Parâmetros do erro de posicionamento, com diferentes conjuntos de pontos. 48
3.5 Erro de posiçao e Erro relativo da posição - Sensor interno 52
3.6 Erro de posicionamento estimado utilizando o primeiro método - sensores internos para $N = 18$ pontos da grade de calibração. 52
3.7 Erro de posicionamento estimado utilizando o primeiro método - sensores internos para $N = 10$ pontos da grade de calibração. 53
3.8 Erro de posicionamento estimado utilizando o primeiro método - sensores internos para $N = 5$ pontos da grade de calibração. 53
3.9 Erro de posicionamento estimado utilizando o primeiro método - sensores internos para $N = 3$ pontos da grade de calibração. 53
3.10 Parâmetros de simulação utilizados nas simulações com sensor visual. . 62
3.11 Erro de posiçao e Erro relativo da posição - Sensor externo 67
3.12 Erro de posicionamento estimado utilizando o segundo método - sensores externos para $N = 18$ pontos da grade de calibração. 67
3.13 Erro de posicionamento estimado utilizando o segundo método - sensores externos para $N = 10$ pontos da grade de calibração. 68
3.14 Erro de posicionamento estimado utilizando o segundo método - sensores externos para $N = 5$ pontos da grade de calibração. 68
3.15 Erro de posicionamento estimado utilizando o segundo método - sensores externos para $N = 3$ pontos da grade de calibração. 68

Capítulo 1

Introdução

Nos dias atuais em que a tecnologia avança a uma taxa altíssima, a robótica atravessa uma época de contínuo crescimento, devido aos inúmeros recursos que os sistemas de microcomputadores oferecem, o que permitirá, em um curto intervalo de tempo, o desenvolvimento de robôs inteligentes. É crescente a necessidade de realizar tarefas com eficiência e precisão. Há também tarefas a serem consideradas, em lugares em que a ação do homem é difícil, arriscada e até mesmo impossível, como em unidades de sistemas submarinos de extração de petróleo em águas profundas ou em meio a imensidão do espaço. Para executá-las, faz-se necessário a presença de dispositivos mecatrônicos (robôs), que as realizam sem riscos aos seres humanos.

Fazendo um breve histórico, o termo *robô* foi originalmente utilizado em 1921, pelo dramaturgo checo Karen Capek, na peça teatral *Os robôs universais de Russum* (RUR), como referência a um autômato que acabava-se rebelando contra o ser humano. Robô deriva da palavra *robota*, de origem eslava, que significa "trabalho forçado". Nessa época a idéia de um "homem mecânico"parecia pertencer a alguma obra de ficção. O desejo de se construir robôs não é so do homem moderno, alguns fatos nos mostram que a idéia não é nova, por exemplo: são muitas as referências sobre a construção do homem mecânico por relojoeiros, que os exibiam em feiras. Há também relatos acerca de algumas animações mecânicas realizadas por Leonardo da Vinci, tais como um leão animado, e seus esforços para fazer máquinas que reproduzissem o vôo das aves. Porém, esses dispositivos eram muitos limitados, pois não podiam realizar mais

do que uma tarefa, ou um conjunto reduzido delas.

A idéia de construir robôs começou a tomar força no início do século XX com a necessidade de aumentar a produtividade industrial e melhorar a qualidade dos produtos. Nessa época o robô industrial encontrou suas primeiras aplicações. O primeiro robô industrial moderno foi desenvolvido por George Devol e Joe Engelberger no final dos anos 50. Engelberger criou a empresa Unimation Inc. que iniciou a comercialização de robôs industriais, sendo esse o motivo pelo qual é chamado de "pai da robótica".

O avanço tecnológico das últimas décadas teve reflexo direto na organização das indústrias, as quais buscam minimizar seus custos operacionais através da adoção de diversos modelos de produção. Nesse contexto, destacam-se a automação programável, relativa à fabricação em série de pequenos e médios lotes de produtos, e a automação flexível, relativa à fabricação de lotes variáveis de produtos diversos.

Os robôs industriais têm sido muito utilizados nos processos de automação programável e flexível, pois são essencialmente máquinas que realizam os mais diversos movimentos programados, adaptando-se às necessidades operacionais de determinadas tarefas pelo emprego de garras e/ou ferramentas de forma adequada.

A Associação de Robótica Industrial (RIA) define o robô industrial como um manipulador multifuncional reprogramável, projetado para movimentar materiais, partes, ferramentas ou peças especiais através de diversos movimentos programados, para o desempenho de uma variedade de tarefas (Romano 2002).

Em ambientes de difícil acesso como em unidades de sistemas submarinos de extração de petróleo em águas profundas, onde a precisão é de importância para a realização de uma tarefa especificada, deve-se considerar o problema de calibração.

A calibração de robôs em geral é o termo adotado para referênciar os procedimentos de determinação dos valores reais das dimensões geométricas e características mecânicas da estrutura de um robô. O contexto de calibração adotado neste trabalho refere-se não somente ao ajuste dos parâmetros de Denavit-Hartenberg mas ao replanejamento de uma tarefa quando existem incertezas quanto a localização absoluta da célula robótica.

A programação *off-line* pode ser definida como os processos mediante os quais é realizada a programação de robôs em ambientes de operação complexos, sem a necessidade dos dispositivos automatizados e do próprio robô.

O replanejamento de trajetória em ambientes hostis de difícil acesso é importante

devido a imprecisão da célula robótica em relação à localização adotada como original para resolver o problema de planejamento de trajetória. Portanto esses ambientes enfrentam este problema que normalmente não é considerado, porém sua relevância é importante pois a estimação deste erro de posicionamento existente minimizaria as diferenças entre o modelo original e o real. Para a estimação do erro de posicionamento são medidos no mínimo três pontos de uma grade de calibração perfeitamente conhecida através de sensores internos e externos. Contudo as medidas obtidas destes sensores apresentam ruído, sendo assim optou-se por utilizar um conjunto maior de pontos visando uma melhor exatidão na obtenção dos parâmetros do erro de posicionamento, sendo dessa forma os métodos propostos baseados num algoritmo de mínimos quadrados.

Os métodos de calibração propostos são três, os dois primeiros métodos propõe resolver o problema no espaço operacional e o terceiro método resolve o problema no espaço de velocidades cartesianas. O primeiro método utiliza informações de sensores internos (encoders), e o segundo método obtêm informações de sensores externos, onde neste se utiliza uma câmera com a função de sensor visual fixa no espaço de trabalho do manipulador, sendo este método complementar ao primeiro. O terceiro e último método propõe realizar a calibração trabalhando com as velocidades linear e angular e dessa forma considerar implicitamente a transformação transformação homogênea entre a localização original e atual, que é obtida com os dois primeiros métodos,.

1.1 Motivação do trabalho

Em ambientes de difícil acesso, tal como ocorre em unidades de sistemas submarinos de extração de pétroleo em águas profundas, onde o planejamento de trajetória a ser seguido normalmente é feito na superfície, porém quando a célula robótica é imersa apresenta incerteza em sua localização absoluta em relação à localização original assumida, dessa forma caracterizando-se em um problema de replanejamento de trajetória. Contudo, o replanejamento é realizado através da estimação do erro de posicionamento entre as localizações já citadas acima, e como também utilizando a trajetória ideal já planejada. Para isso são propostos três métodos de calibração para estimação desse

erro. Dentre os quais dois propõe resolver o problema no espaço operacional e o terceiro no espaço de velocidades cartesianas.

1.2 Estado da arte

A calibração de robôs é o termo adotado para referênciar os procedimentos de determinação dos valores reais das dimensões geométricas e características mecânicas da estrutura de um robô. Esses valores são usualmente classificados como parâmetros cinemáticos ou dinâmicos. Os parâmetros cinemáticos descrevem principalmente os comprimentos dos elos do robô e a orientação relativa as juntas, e sua determinação constitui a calibração estática de um robô. Já os parâmetros dinâmicos descrevem principalmente as massas dos elos e das juntas e o atrito interno. A determinação desses parâmetros é de importância para a melhoria e a manutenção da precisão pela qual as posições e movimento do robô são controlados.

O erro de posição de um robô pode ser atribuído a diversas causas e com isso podem ser classificados como erros sistemáticos e aleatórios. Os erros aleatórios estão associados à resolução finita dos codificadores das juntas e às folgas nas engrenagens. Já os erros sistemáticos são aqueles que independem da configuração do robô, como os erros associados ao comprimento dos elos e à posição inicial de codificadores das juntas, ou que variam previsivelmente com a posição, associados à deflexão elástica ou a erros de transmissão em engrenagens. Esses erros sistemáticos e aleatórios em robôs são usualmente chamados de erros geométricos e erros não geométricos. Em (Schoröer, Albrigth & Grenthlein 1997) reporta-se que robôs indústriais têm seu erro de posicionamento largamente influenciado pelos erros sistemáticos, e a calibração estática é suficiente para reduzir em até 95% seus erros de posicionamento. Um ponto muito importante no sucesso da determinação numérica desses parâmetros, que podem variar de 30 a 100% para robôs industriais de 6 graus de liberdade, dependendo da abrangência do modelo, é a construção de um modelo que não apresente redundâncias paramétricas, o que não é uma tarefa trivial (Schöer 1993, Motta & s. McMaster 1999).

A calibração de robôs tem apresentado importância crescente no meio industrial e em outras áreas afins e a sua aplicação tem sido feita tanto nas empresas onde

são fabricados quanto em aplicações como em sistemas de manufatura integrados por computador, tais quais aqueles utilizados para operações de montagem ou soldagem (Hidalgo & Brunn 1998). A produção, implementação e operação de robôs são áreas onde os resultados da calibração podem levar a uma melhora siginificativa de precisão e/ou economia dos recursos.

A calibração de robôs é um processo integrado de modelagem, medição, identificação numérica das características físicas de um robô, e na implantação de um novo modelo corrigido. Vários sistemas para calibração de robôs têm sido desenvolvidos para realização dos procedimentos de calibração de forma automática. Porém, os sistemas disponíveis atualmente ainda não são apropriados, pois nenhum desses sistemas combinam satisfatoriamente baixo custo e precisão, facilidade de uso e rapidez de ajuste e implementação.

Em (Meggiolaro & Dubowsky 2000) a calibração de robôs é executada utilizando um método analítico para eliminar os parâmetros redundantes existentes, que frequentemente são não-intuitivos, que podem comprometer a robustez da calibração. As expressões analíticas gerais e interpretação física dos parâmetros redundantes bem como as combinações lineares apresentadas na parametrização do erro redundante são desenvolvidas para um manipulador de elo serial, sendo expressas através dos parâmetros de Denavit-Hartenberg, que são usadas para eliminar os parâmetros redundantes do modelo de erro antes do processo de identificação. A forma não redundante da matriz Jacobiana de identificação é obtida usando estas expressões, seguindo para a calibração sistemática com melhora na exatidão.

Em (Mavroidis, Dubowsky, Drouet, Hintersteiner & Flanz 1997) uma metodologia sistemática para calcular os erros de posição e orientação no efetuador de um manipulador robótico é apresentada. Esta metodologia trata das fontes de erro físico, que podem existir, de uma forma unificada durante o projeto do sistema, sendo que o efeito que elas têm no posicionamento exato no fim do efetuador pode ser comparada e as fontes dominantes identificadas. E baseado nesta metodologia, um pacote de *software* foi desenvolvido utilizando uma combinação de computação numérica e simbolica para calcular o modelo de erro para um manipulador elo serial de seis graus de liberdade.

Um método utilizado para calibrar o robô em relação a uma estrutura de geometria conhecida consiste em dotar o efetuador do robô de uma ponta de prova. Tocando

a superfície da estrutura em pontos adequados é possível encontrar a transformação que relaciona as coordenadas do robô com as coordenadas da estrutura de interesse. A idéia inicialmente parece simples, porém sabe-se que é preciso utilizar uma escolha cuidadosa do método de estimação, por exemplo, mínimos quadrados (Hollerbach & Wampler 1996, Jennings & Mckeown 1993).

Em (Lei, Jingtai, Weiwei, Shuihua & Xingbo 2004) o método de calibração do robô é baseado na geometria, utilizando um manipulador robótico de 6 graus de liberdade. Sendo o dispositivo de calibração incluíndo somente um conjunto de projeções de luz, consistindo de três lasers. O algoritmo proposto é baseado na geometria, onde as coordenadas da posição do laser na mesa, no sistema de coordenadas global é obtida primeiro pelo processamento dos dados da imagem de uma câmera CCD fixa acima da mesa. Com base nisso, o mapeamento entre o sistema de coordenadas global e o sistema de coordenadas da base do robô pode então ser utilizado para localizar o robô no seu meio pela análise geométrica. O método de calibração envolvido é simples comparado com os algoritmos de calibração baseados na álgebra tradicional, pois neste método nenhuma operação com matriz complexa é envolvida.

1.3 Apresentação do problema

Seja uma trajetória a ser percorrida no espaço da junta em relação a um sistema de coordenadas inercial E_I. Considere inicialmente que a base do manipulador coincide com o sistema de coordenadas inercial, vide Figura 1.1. Para que este siga a trajetória definida em E_I é necessário haver o planejamento desta, este planejamento é realizado através da obtenção de entradas de referência para o movimento do sistema de controle que assegure que o manipulador execute a trajetória planejada, onde estas entradas são os ângulos das juntas, os quais são obtidos através da cinemática inversa. Agora considere que a base do manipulador tenha sido deslocada em relação a localização original definida como E_b, conforme Figura 1.2. Para que este siga a mesma trajetória definida anteriormente é necessário haver o replanejamento desta trajetória. E para esse replanejamento acontecer é preciso estimar o erro de posicionamento entre as localizações original e atual, ou seja, entre E_b e E_{b_1}, assim como também utilizar a

trajetória ideal planejada em E_b, vide Figura 1.3. A Figura 1.3 ilustra como é obtido o erro de posicionamento, onde nesta λ_1 λ_2 e λ_3 são os pontos de calibração medidos através de sensores internos ou externos.

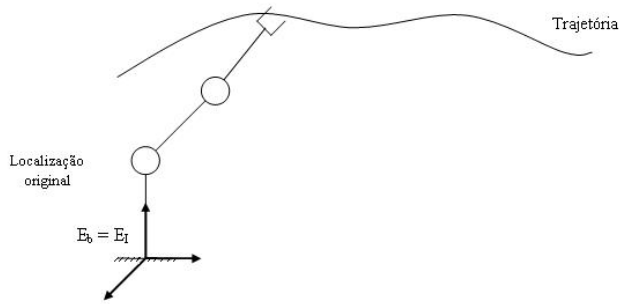

FIGURA 1.1: Manipulador na localização original percorrendo uma trajetória desejada.

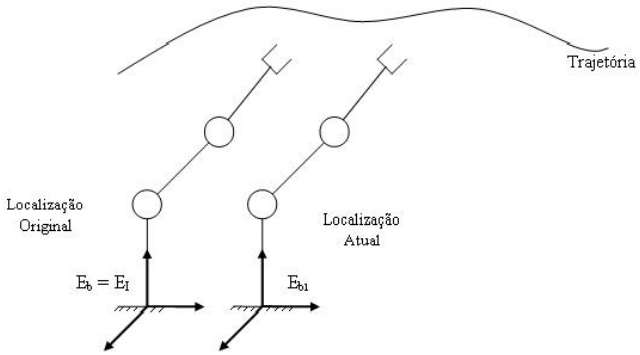

FIGURA 1.2: Manipulador deslocado em relação ao sistema de coordenadas inercial.

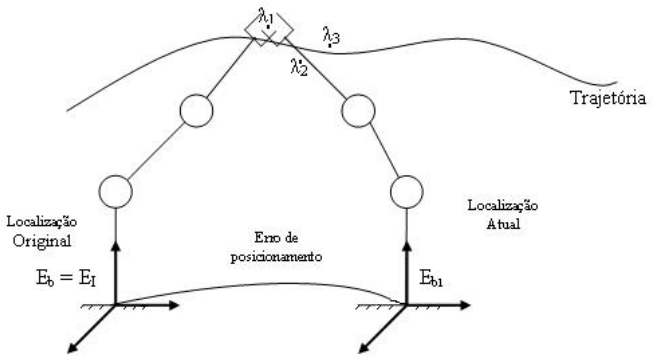

FIGURA 1.3: Esquema do problema em estudo.

1.3.1 Metodologia

Como já mencionado este estudo aborda três métodos de calibração, onde nos dois primeiros métodos o erro de posicionamento é obtido no espaço operacional e no terceiro é obtido no espaço de velocidades cartesianas. Portanto pode-se dividir o problema em três etapas, conforme ilustrado no diagrama em blocos da Figura 1.4:

- a primeira etapa consiste no planejamento ideal da trajetória, aqui o manipuldor se encontra na localização assumida como original E_b, onde a trajetória planejada é dada através dos ângulos das juntas;

- a segunda etapa consiste na calibração, onde esta é realizada no espaço operacional, nos dois primeiros métodos, e no espaço de velocidades cartesianas no terceiro método. Contudo, em ambos são baseados em mínimos quadrados;

- a terceira etapa consiste no replanejamento de trajetória já que existe incerteza na localização absoluta da célula robótica, em relação a localização adotada como original, dessa forma o replanejamento é necessário para que o manipulador execute a mesma trajetória planejada em E_b, vide Figura 1.3.

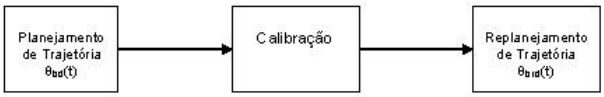

FIGURA 1.4: Diagrama em bloco das etapas do trabalho.

Portanto os três métodos propostos a serem apresentados são validados via simulação, utilizando o modelo cinemático do manipuldor Zebra Zero e o pacote de robótica do *Matlab*. Contudo, para emular uma situação real é adicionado ruído nas simulações, pois as informações obtidas dos sensores apresentam ruído. Sendo válido dizer que a grade de calibração utilizada para as simulações é perfeitamente conhecida, porém na prática os pontos para calibração são tomados dentro de uma estrutura conhecida.

1.3.2 Contribuição

A principal contribuição que se propõe com este trabalho, está nos métodos de replanejamento de trajetória apresentados. Pois a idéia consiste em fazer o robô executar a mesma trajetória planejada com ele na localização original, contudo, como existem incertezas na localização absoluta da célula robótica é necessário realizar o replanejamento de trajetória.

1.4 Organização do trabaho

O presente trabalho está dividido em cinco capítulos. O Capítulo 2 aborda o planejamento de trajetória e a programação *off-line* de um manipulador robótico. A necessidade de se abordar o planejamento de trajetória é gerar entradas de referência para o movimento do sistema de controle que assegure que o manipulador execute a trajetória planejada. O Capítulo 3 propõe dois métodos de calibração, baseados em mínimos quadrados, os quais utilizam informações de sensores externos e/ou internos para calcular a transformação homogênea entre as localizações original e real, ou seja, o erro de posicionamento. O Capítulo 4 aborda o terceiro método de calibração, a idéia neste capítulo é considerar implicitamente a transformação homogênea entre as localizações original e atual e trabalhar com as velocidades linear e angular que sintetizariam esta transformação. O Capítulo 5 apresenta as conclusões obtidas.

Capítulo 2

Programação *Off-line* e Planejamento de Trajetória

Este capítulo descreverá o planejamento de trajetória e a programação *off-line* de um manipulador robótico. A necessidade de se abordar o planejamento de trajetória é gerar entradas de referência para o movimento do sistema de controle que assegure que o manipulador execute a trajetória planejada. E a programação *off-line* pode ser definida como os processos mediante os quais são realizadas as programações de robôs em ambientes de operação complexos, sem a necessidade dos dispositivos automatizados e do próprio robô. Nesta a partir de um *software* para visualização gráfica do modelo geométrico de robôs, pode-se especificar pontos de passagem correspondentes à trajetória do robô, expressos em coordenadas angulares. Esses pontos de passagem podem ser obtidos a partir do movimento angular de cada junta ou a partir do modelo geométrico. A partir de um conjunto de pontos correspondentes à trajetória a ser realizada pelo robô, é possível implementar algoritmos *off-line* para interpolação e filtragem, levando-se em consideração aspectos dinâmicos e testes de colisão. Contudo pode-se dizer que o planejamento de trajetória faz parte da programação *off-line* (Rosário 2005).

2.1 Programação *off-line*

Programação *off-line* é um processo pelo qual a programação dos robôs é desenvolvida, de forma parcial ou completa, sem a necessidade do uso do robô. A programação *off-line* aumenta a flexibilidade e a habilidade de utilização de robôs, com uma variedade ilimitada de cenários e movimentos. Um dos problemas que impedem o rápido desenvolvimento desses sistemas de programação *off-line* é a limitada precisão estática e dinâmica no posicionamento dos mesmos em seu volume de trabalho. Como por exemplo erros cinemáticos e dinâmicos (Romano 2002).

A trajetória de um robô é definida pelo conjunto de ângulos associados ao movimento angular de cada grau de liberdade, que, utilizando um algoritmo de interpolação, servirá como sinal de referência para o controlador de posição de cada junta robótica, a qual o comparará com os sinais provenientes dos transdutores de posição das juntas.

A programação off-line apresenta certas vantagens, por exemplo: não utilizar o robô no período de programação, ela pode ser realizada em computadores, necessitando apenas de um modelo matemático. Onde esse modelo contém informações sobre a cinemática e a dinâmica do robô.

Normalmente a programação de tarefas de robôs é realizada no espaço das juntas, sem a necessidade de um modelo geométrico, e a trajetória angular de mesma natureza dos sinais provenientes do transdutor de posição serve como sinal de referência para o controlador de cada junta robótica. Entretanto, a realização de algumas tarefas relacionadas a um sistema de referência colocado na ferramenta (espaço cartesiano) exige o conhecimento completo do modelo geométrico e torna necessária a transformação de coordenadas, tendo em vista que o sinal de referência correspondente à trajetória, necessário para o controle das juntas, deve ser angular. O modelo geométrico é aquele que expressa a posição e a orientação da extremidade do efetuador em relação a um sistema de coordenadas fixo à base do robô em função de suas coordenadas generalizadas (angulares, no caso de juntas rotacionais). Essa relação é expressa matematicamente por uma matriz de transformação homogênea, que relaciona o sistema de coordenadas da base com o sistema de coordenadas da extremidade do efetuador (Rosário 2005).

O incremento na complexidade das aplicações em robótica torna as vantagens da

programação *off-line* mais atrativas, comparando com a programação *on-line*, como, por exemplo, em uma linha de produção, onde o tempo que o robô fica fora da linha de produção, gastos na programação, pode prejudicar substancialmente a sua utilidade. As vantagens da programação *off-line*, podem ser classificadas como (Romano 2002):

- Redução do tempo ocioso: O robô pode ser mantido no seu ambiente de trabalho enquanto a próxima tarefa é programada, proporcionando maior flexibilidade aos robôs;

- Ambientes potencialmentes perigosos: Redução no tempo de permanência do operador próximo ao robô, o que diminui o risco de acidentes por comportamento impróprio do equipamento;

- Sistema simplificado de programação: Pode-se usar a forma *off-line* para programar uma grande variedade de robôs sem a necessidade de conhecer as peculiaridades de cada controlador, reduzindo assim o índice de reciclagem dos programadores;

- Integração com sistemas CAD/CAM (desenho eletrônico): Habilita a interface com o banco de dados de peças, centralizando a programação de robôs com esses sistemas; possibilita o acesso a outras funcionalidades, como, por exemplo planejamento e controle;

- Sistemas de programação off-line com CAD/CAM integrados podem produzir um modelo da planta (robô + ambiente de trabalho) que pode ser usado para detectar colisões dentro do espaço de trabalho, possibilitando determinados movimentos e evitando assim danos ao equipamento.

2.1.1 Limitações da programação off-line

Como já foi mencionado a programação off-line, requer a existência de um modelo teórico do robô e do ambiente, objetivando usar esse modelo para simular o comportamento real do robô. A implementação da programação *off-line* enfrenta principalmente três problemas:

- dificuldade em desenvolver um sistema de programação generalizado que seja independente do robô e de suas aplicações;

- para reduzir a incompatibilidade entre robôs e sistemas de programação, faz-se necessário definir padrões para as interfaces;

- programas gerados *off-line* devem levar em conta os erros e imprecisões entre o modelo idealizado e o mundo real (Romano 2002).

Devido às imprecisões do modelo teórico idealizado e as variáveis inerentes ao processo no mundo real, sequências simuladas geralmente não atingem o objetivo de controlar o robô sem erros. Por outro lado, em situações práticas o robô não atinge o local calculado pelo modelo precisamente. Essas discrepâncias podem ser atribuídas aos seguintes fatores

No robô:

- falta de tolerância na montagem dos seus componentes, provocando o aumento na variação do *off-set* das juntas. Assim erros na estrutura são amplificados e produzem grandes erros de posicionamento no efetuador;

- falta de rigidez na estrutura do robô, pode causar grandes erros, quando o mesmo está sujeito a condições severas de carga;

- incompatibilidade entre robôs do mesmo modelo. Devido a diferenças na inicialização do sistema de controle de cada robô, a mesma programação *off-line* pode apresentar pequenos erros.

No controlador:

- resolução insuficiente do controlador. A resolução especifica o menor incremento de movimento atingível pelo controlador;

- precisão numérica do controlador: é afetado pelo número de parâmetros envolvidos e tamanho das palavras de comando usadas no controlador; além da eficiência do algoritmo usado para os propósitos de controle (Romano 2002).

No ambiente:

- dificuldades na determinação precisa dos objetos com relação ao sistema de coordenadas generalizadas;
- alterações no ambiente, como temperatura, podem causar efeitos adversos no desempenho do robô.

Modelo e sistema de programação:

- a precisão numérica do processador do computador;
- a qualidade dos dados do modelo real. Isso determina a precisão final do programa gerado *off-line* (Romano 2002).

A composição desses erros através de todo o sistema de programação *off-line* pode levar a discrepâncias de magnitude significante, algumas delas podem ser remediadas com a calibração. E para que a programação *off-line* se torne uma ferramenta prática, essa magnitude deve ser reduzida a um nível em que ajustes do posicionamento final possam ser executados automaticamente.

2.2 Planejamento de trajetória

A programação de tarefas de robôs na grande maioria das aplicações industriais é realizada por aprendizagem, consistindo no movimento individual de cada junta, sem que necessite, assim, de um modelo geométrico. Dessa forma, a programação de trajetória de um robô torna-se simples, e não requer o conhecimento do modelo (Rosário 2005).

Grande parte das aplicações industriais, por exemplo, exige que o robô realize tarefas em relação a sua extremidade (posicionamento e orientação). Porém como o robô é controlado através de suas variáveis articulares, para controlá-lo em relação ao sistema de coordenadas cartesianas é preciso desenvolver metodologias que transformem as coordenadas, sendo necessária a implementação de algoritmos numéricos rápidos para

resolver o modelo cinemático inverso e, assim, determinar o deslocamento de cada grau de liberdade para que o robô execute as tarefas em relação ao referêncial de trabalho.

Dessa forma, determinada trajetória deverá ser definida por um conjunto de ângulos associados ao movimento angular de cada grau de liberdade do robô, que, após algoritmo de interpolação, servirão como sinal de referência para o controlador de posição de cada junta do robô. Porém antes de se abordar o planejamento de trajetória, descreve-se a transformação homogênea, modelagem cinemática de manipuladores e a modelagem cinemática inversa.

2.2.1 Transformação homogênea

A transformação homogênea expressa a mudança de coordenadas entre dois sistemas de coordenadas. A Figura 2.1 mostra um ponto P no espaço e o vetor p_0 de coordenadas deste ponto com respeito ao sistema de coordenadas de referência E_0, bem como um outro sistema de coordenadas E_1 em que p_1 é a representação do ponto P neste sistema de coordenadas. Portanto, a posição do ponto P com respeito ao sistema de coordenadas de referência é expresso por

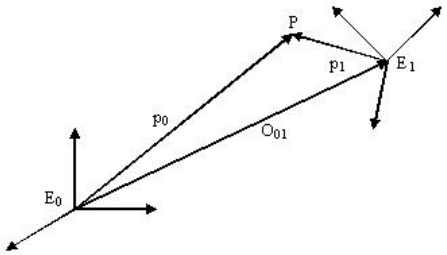

FIGURA 2.1: Representação de um ponto P em diferentes sistemas de coordenadas.

$$p_0 = o_{01} + R_{01}p_1, \tag{2.1}$$

onde o_{01} é o vetor que descreve a origem do sistema de coordenadas E_1 com respeito a E_0 e R_{01} é a matriz de rotação de E_1 com respeito a E_0.

Então (2.1) representa a transformação de coordenadas (translação + rotação) de um vetor limitado entre dois sistemas de coordenadas.

A transformação inversa pode ser obtida premultiplicando ambos os lados de (2.1) pela matriz de rotação R_{01}^T conforme

$$p_1 = -R_{01}^T o_{01} + R_{01}^T p_0, \tag{2.2}$$

ou

$$p_1 = -R_{10} o_{01} + R_{10} p_0. \tag{2.3}$$

Para uma representação compacta da relação entre os sistemas de coordenadas do mesmo ponto em dois sistemas de coordenadas diferentes, a representação homogênea de um vetor p genérico pode ser introduzido como o vetor $\tilde{p}$ formado pelo acréscimo de uma quarta componente unitária, isto é,

$$\tilde{p} = \begin{bmatrix} p \\ \\ 1 \end{bmatrix}. \tag{2.4}$$

Adotando esta representação para os vetores p_0 e p_1 em (2.1), a transformação de coordenadas pode ser escrita em termos de uma matriz (4×4),

$$T_{01} = \begin{bmatrix} R_{01} & o_{01} \\ \\ 0 & 1 \end{bmatrix}, \tag{2.5}$$

a qual de acordo com (2.4) é chamada *matriz de transformação homogênea*. Como pode ser visto em (2.5) a transformação do sistema de coordenadas E_1 para o sistema de coordenadas E_0 é expresso por uma única matriz que contém a matriz de rotação de E_1 para E_0 e o vetor posição da origem de E_0 para origem de E_1.

Portanto a transformação de coordenadas que descreve a posição e orientação de um sistema de coordenadas E_n com respeito a E_0 é dada por

$$T_{0n} = T_{01} T_{12} \dots T_{n-1,n}. \tag{2.6}$$

2.2.2 Modelagem cinemática de manipuladores

Um manipulador pode ser representado esquematicamente por uma cadeia cinemática de corpos rígidos (elos), conectados por juntas, prismáticas ou rotacionais. Cada par junta-elo constitui um grau de liberdade. Assim, para um manipulador com n graus de liberdade, temos n pares junta-elo onde o primeiro elo é a base de sustentação do robô (sistema de coordenadas inerciais fixo) e o último elo é o que incorpora a ferramenta de trabalho. Portanto a estrutura mecânica de um manipulador é caracterizado por um número de graus de mobilidade que determinam sua configuração. Cada grau de mobilidade é tipicamente associado com uma articulação da junta e constitui uma variável da junta. Então a cinemática direta calcula a posição e orientação da extremidade do manipulador como uma função das variáveis da junta (Sciavicco & Siciliano 1996).

O conhecimento completo das variáveis das juntas de um robô (θ_i), determina o posicionamento de sua ferramenta no sistema de coordenadas de trabalho. De um modo geral, os três primeiros graus de liberdade de um robô são responsáveis pelo posicionamento de sua ferramenta no espaço operacional, e os restantes pela sua orientação.

Para representar a situação relativa dos vários corpos da cadeia, é fixado a cada elemento $C_i(i = 1, 2, ..., n)$, n corpos móveis, e um sistema de coordenadas E. Pode-se relacionar um determinado sistema de coordenadas E_{i+1} $(O_{i+1}, X_{i+1}, Y_{i+1}, Z_{i+1})$ com o seu anterior E_i (O_i, X_i, Y_i, Z_i), bem como o sistema de coordenadas da origem da base (vide Figura 2.2) através de (2.7), onde $R_{i,i+1}$ representa as matrizes de rotação e P_i representa o vetor de translação de uma origem a outra. A matriz $R_{i,i+1}$ pode ser interpretada pela composição de rotações sucessivas, conforme (2.8) (Romano 2002).

$$O_{i+1} = O_i + R_{i,i+1}P_i, \tag{2.7}$$

$$R_{i,i+1} = R_{12}R_{23}...R_{i-1,i}. \tag{2.8}$$

Qualquer rotação no espaço pode ser decomposta em um grupo de rotações elemen-

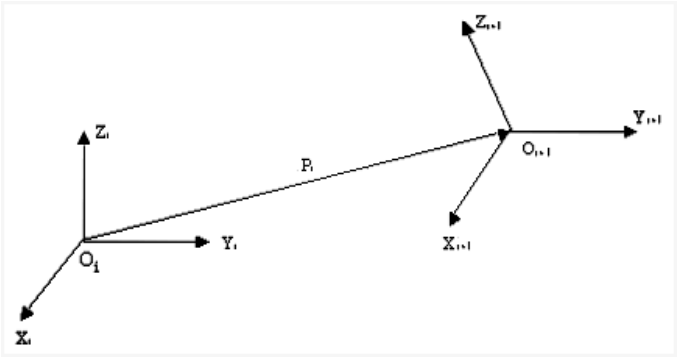

FIGURA 2.2: Sistema de referência.

tares ao longo dos eixos X, Y e Z. A matriz de rotação elementar usada na equação de transformação é associada com a rotação elementar do referencial correspondente em relação ao seu anterior.

Consequentemente, o posicionamento completo de um corpo rígido no espaço poderá ser obtido através da Equação (2.7), que fornece a sua posição e orientação. A orientação expressa através de componentes angulares associadas às três direções de rotação correspondentes aos eixos de referência do sistema de coordenadas, como por exemplo os ângulos de Euler ou através da representação quaternions unitário.

2.2.2.1 Ângulos de Euler

Uma representação da orientação em termos de três parâmetros independentes - constituem uma representação mínima. A representação mínima da orientação pode ser obtida usando um conjunto de três ângulos $\Phi = [\varphi, \vartheta, \psi]^T$, considerando que a matriz de rotação é expressa por rotações elementares em torno de cada eixo do sistema de coordenadas como uma função de um único ângulo.

A rotação descrita pelos ângulos de Euler ZYZ é obtida de uma composição das seguintes rotações elementares (vide Figura 2.3): inicialmente é realizada uma rotação em torno do eixo z; em seguida é realizada a rotação em torno do eixo y após a aplicação da primeira rotação; por fim é realizada a rotação novamente em torno do eixo z após terem sido aplicadas as duas outras rotações.

Sendo a orientação resultante obtida da forma

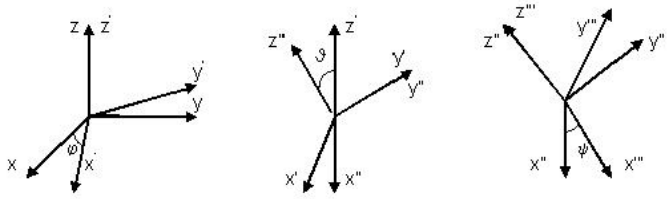

FIGURA 2.3: Representação dos Ângulos Z-Y-Z de Euler.

$$R(\Phi) = R_z(\varphi)R_y(\vartheta)R_z(\psi), \tag{2.9}$$

ou

$$R(\Phi) = \begin{bmatrix} \cos\varphi\cos\vartheta\cos\psi - \text{sen}\,\varphi\text{sen}\,\psi & -\cos\varphi\cos\vartheta\text{sen}\,\psi - \text{sen}\,\varphi\cos\psi & \cos\varphi\text{sen}\,\vartheta \\ \text{sen}\,\varphi\cos\vartheta\cos\psi + \cos\varphi\text{sen}\,\psi & -\text{sen}\,\varphi\cos\vartheta\text{sen}\,\psi & \text{sen}\,\varphi\text{sen}\,\vartheta \\ -\text{sen}\,\vartheta\cos\varphi & \text{sen}\,\vartheta\text{sen}\,\psi & \cos\vartheta \end{bmatrix}. \tag{2.10}$$

Representando de forma simplificada

$$R(\Phi) = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}. \tag{2.11}$$

Os ângulos de rotação podem ser obtidos a partir de $R(\Phi)$ aplicando-se as seguintes equações, porém se $r_{23} = r_{13} = 0$ há presenças de singularidades:

$$\varphi = atan2(r_{23}, r_{13}) \tag{2.12}$$

$$\vartheta = atan2(\sqrt{r_{13}^2 + r_{23}^2}, r_{33}) \tag{2.13}$$

$$\psi = atan2(r_{32}, -r_{31}) \tag{2.14}$$

2.2.2.2 Quaternions unitário

Outra forma bastante utilizada para orientação é a representação quaternion unitário. Onde a matriz de orientação do efetuador de um manipulador em relação ao sistema de coordenadas da base, é expressa a partir da utilização de quatro parâmetros. A principal vantagem dessa representação é que a utilização de quatro parâmetros permitirá a obtenção de soluções únicas, implicando assim num número menor de manipulações computacionais.

Seja $v = [v_x\ v_y\ v_z]^T$ o vetor unitário de um eixo de rotação com respeito ao sistema de coordenadas de referência $O - xyz$,

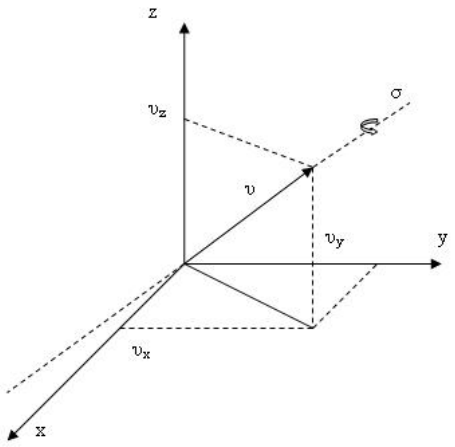

FIGURA 2.4: Rotação em torno de um eixo arbitrário.

Onde σ é o ângulo de rotação sobre o eixo v. O quaternion unitário é definido como $\mathcal{Q} = (q_0, q_v)$ sendo

$$q_0 = \cos\frac{\sigma}{2}, \tag{2.15}$$

$$q_v = \text{sen}\,\frac{\sigma}{2}v, \tag{2.16}$$

onde q_0 é denominada a parte *escalar* do quaternion, enquanto $q_v = [q_{v_x}\ q_{v_y}\ q_{v_z}]^T$ é chamada a parte *vetorial* do quaternion, estando sujeitas à seguinte restrição

$$q_0^2 + q_{v_x}^2 + q_{v_y}^2 + q_{v_z}^2 = 1. \tag{2.17}$$

Em vista (2.15), (2.16) e (2.17), a matriz de rotação correspondente ao quaternion dado assume a seguinte forma:

$$R(q_0, q_v) = \begin{bmatrix} 2(q_0^2 + q_{v_x}^2) - 1 & 2(q_{v_x}q_{v_y} - q_0 q_{v_z}) & 2(q_{v_x}q_{v_z} + q_0 q_{v_y}) \\ 2(q_{v_x}q_{v_y} + q_0 q_{v_z}) & 2(q_0^2 + q_{v_y}^2) - 1 & 2(q_{v_y}q_{v_z} + q_0 q_{v_x}) \\ 2(q_{v_x}q_{v_z} + q_0 q_{v_y}) & 2(q_{v_y}q_{v_z} + q_0 q_{v_x}) & 2(q_0^2 + q_{v_z}^2) - 1 \end{bmatrix}. \tag{2.18}$$

Entretanto, quando é desejado resolver o problema inverso, para computar o quaternion correspondente a uma matriz de rotação dada

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \tag{2.19}$$

o seguinte resultado é útil

$$q_0 = \frac{1}{2}\sqrt{r_{11} + r_{12} + r_{13} + 1}, \tag{2.20}$$

$$q_v = \begin{bmatrix} sgn(r_{32} - r_{23})\sqrt{r_{11} - r_{22} - r_{33} + 1} \\ sgn(r_{13} - r_{31})\sqrt{r_{22} - r_{33} - r_{11} + 1} \\ sgn(r_{21} - r_{12})\sqrt{r_{33} - r_{11} - r_{22} + 1} \end{bmatrix}, \tag{2.21}$$

onde $sgn(x) = 1$ para $x \geq 0$ e $sgn(x) = 0$ para $x < 0$. Note que em (2.20) é implicitamente assumido $q_0 \geq 0$; isto corresponde a um ângulo $\sigma \in [-\pi, \pi]$, e portanto

qualquer rotação pode ser descrita. Além disso, as soluções para q_0 e q_v são livres de singularidades.

O quaternion extraído a partir de $R^{-1} = R^T$ é representado por $\mathcal{Q}^{-1}$, e pode ser computado como

$$\mathcal{Q}^{-1} = (q_0, -q_v). \tag{2.22}$$

Sejam $\mathcal{Q}_1 = (q_1, q_{v_1})$ e $\mathcal{Q}_2 = (q_2, q_{v_2})$ os quaternions correspondentes às matrizes de rotações R_1 e R_2 respectivamente. O quaternion correspondente ao produto $R_1 R_2$ é dado por

$$\mathcal{Q}_1 * \mathcal{Q}_2 = (q_1 q_2 - q_{v_1}^T 1 q_{v_2}, q_1 q_{v_2} + q_2 q_{v_1} + q_{v_1} \times \ q_{v_2}), \tag{2.23}$$

onde * denota o operador produto de quaternion. Note que, se $\mathcal{Q}_2 = \mathcal{Q}_1^{-1}$ então de acordo com (2.23) vem

$$\mathcal{Q}_1 * \mathcal{Q}_1^{-1} = (1, \ [0\ 0\ 0]). \tag{2.24}$$

2.2.3 Parâmetros de Denavit-Hatenberg

O modelo cinemático de um robô expressa a posição e orientação de seu elemento terminal em relação a um sistema de coordenadas fixo à base do robô, vide Figura 2.5, em função de suas coordenadas generalizadas (coordenadas angulares, no caso de juntas rotacionais).

Sendo o modelo cinemático representado pela expressão:

$$x = k(\theta), \tag{2.25}$$

onde $\theta = (\theta_1, \theta_2, ..., \theta_n)$ são as posições angulares das juntas; e $x = (p_x, p_y, p_z, \varphi, \vartheta, \psi)$: é a configuração onde os três primeiros termos denotam a posição cartesiana e os três últimos, a representação da orientação do elemento terminal.

Como foi dito anteriormente um manipulador pode ser modelado como um sistema de elos articulados através de juntas no espaço tridimensional. O estudo de seu movi-

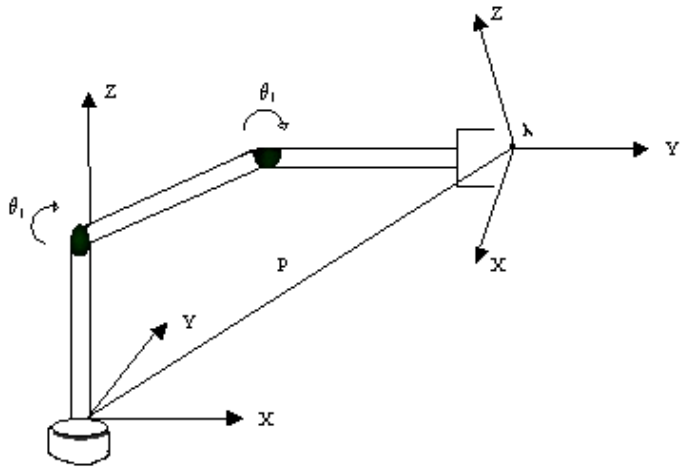

FIGURA 2.5: Representação de um sistema de coordenadas de um robô.

mento, está relacionado a descrição de variáveis de posição das juntas e ao modo como essas variáveis variam em função do tempo (velocidade e aceleração). A notação de Denavit- Hartenberg (DH) é uma ferramenta utilizada para sistematizar a descrição cinemática de sistemas mecânicos articulados com n graus de liberdade (Sciavicco & Siciliano 1996).

Denavit e Hartenberg propuseram um método matricial para estabelecimento sistemático de um sistema de coordenadas fixo para cada elo de uma cadeia cinemática articulada.

A representação de Denavit e Hartenberg resulta na obtenção de uma matriz de transformação homogênea, que expressa a transformação de coordenadas entre dois sistemas de coordenadas. Contudo representa cada sistema de coordenadas do elo na junta, em relação ao sistema de coordenadas do elo anterior. Dessa forma, a partir de transformações sucessivas, podem ser obtidas as coordenadas do elemento terminal de um manipulador, expressas matematicamente no sistema de coordenadas fixo à base.

Na Figura (2.6) (Sciavicco & Siciliano 1996) pode ser visualizado dois elos conectados por uma junta que tem duas superfícies deslizantes, uma sobre a outra remanescente, em contato. Um eixo de uma junta estabelece a conexão de dois elos.

Esses eixos das juntas devem ter dois vetores normais conectadas a eles, uma para cada elo. A posição relativa desses dois elos conectados (elo $i-1$ e elo i) é dada por d_i, que é a distância medida ao longo do eixo da junta entre suas normais. O ângulo da junta θ_i entre as normais é medido em um plano normal ao eixo da junta. Assim, d_i e θ_i

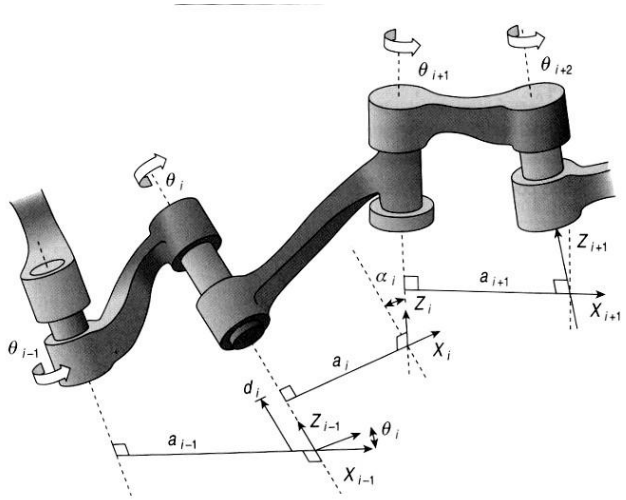

FIGURA 2.6: Configuração de Denavit-Hartenberg.

podem ser chamados, respectivamente, de distância e de ângulo entre elos adjacentes. Eles determinam a posição relativa de elos vizinhos.

Um elo i poderá estar conectado, no máximo, a dois outros elos (elo $i-1$ e elo $i+1$). Assim, dois eixos de junta são estabelecidos em ambos os terminais de conexão. O significado dos elos, do ponto de vista cinemático, é que eles mantêm uma configuração fixa entre suas juntas que podem ser caracterizadas por dois parâmetros: a_i e α_i. O parâmetro a_i é a menor distância medida ao longo da normal comum entre os eixos de junta, isto é, os eixos Z_{i-1} e Z_i para a junta i e a junta $i+1$, respectivamente. Dessa forma a_i e α_i, podem ser chamados, respectivamente, de comprimento e ângulo de torção do elo i. E juntos eles determinam a estrutura do elo i.

Potanto quatro parâmetros que são: a_i, α_i, d_i e θ_i , que são associados com cada elo do manipulador. Estes parâmetros constituem um conjunto suficiente para determinar a configuração cinemática de cada elo do manipulador.

2.2.4 Modelagem cinemática inversa

A cinemática inversa consiste na determinação das variáveis das juntas correspondendo as tarefas definidas no espaço cartesiano, é expressa matematicamente pela inversão do modelo geométrico

$$\theta = k^{-1}(x), \tag{2.26}$$

onde a função k é não-linear e composta da soma de produtos de senos e cossenos das coordenadas generalizadas (translações ou rotações). Por isso, a sua inversão é em geral não-trivial. Portanto como k é não-linear, não é possível garantir a existência e/ou a unicidade da função k^{-1}. No caso geral, pode-se apenas determinar o número máximo de prováveis soluções. Os métodos de solução do problema de inversão do modelo geométrico são analíticos e numéricos iterativos. Sendo os analíticos métodos não gerais, isto é, a inversão analítica não é trivial e, além disso, não há garantia de que seja possível fazê-la para um robô qualquer (Romano 2002). Esses métodos são mais adequados para manipuladores simples, ou seja, aqueles que possuem parâmetros de Denavit Hartenberg (D-H) nulos. E os métodos numéricos iterativos são de caráter geral e convergem para uma solução possível entre todas as existentes.

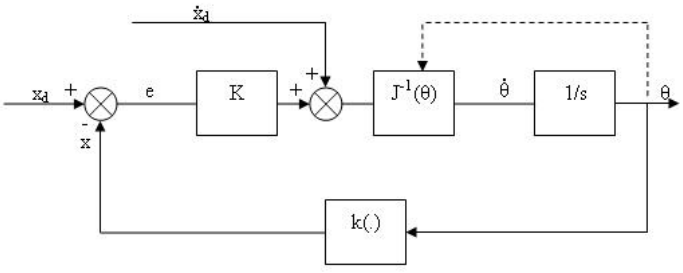

FIGURA 2.7: Método para solução do modelo inverso.

A Figura 2.7 mostra o esquema em bloco do algoritmo de cinemática inversa com Jacobiano inverso. Esse esquema após algumas modificações e considerações é utilizado

no replanejamento de trajetória. Através da Figura 2.7 pode-se ver que o erro entre a posição atual e desejada do efetuador é dada por

$$e = x_d - x = x_d - k(\theta). \tag{2.27}$$

De acordo com a cinemática diferencial (2.27) pode ser escrita como

$$\dot{e} = \dot{x}_d - J(\theta)\dot{\theta}. \tag{2.28}$$

Para (2.28) conduzir um algoritmo de cinemática inversa, é válido relacionar o vetor da velocidade da junta calculado, $\dot{\theta}$, com o erro e de modo que (2.28) forneça uma equação diferencial descrevendo a evolução do erro com o passar do tempo. No entanto, é necessário escolher uma relação entre $\dot{\theta}$ e e que assegure convergência do erro para zero.

Após ter sido formulado o problema de cinemática inversa numa forma algoritmica implica que a variável da junta θ correspondente a posição desejada atribuída a x_d são exatamente obtidas somente quando o erro e está abaixo do mínimo tolerado. A escolha da relação entre θ e e permite encontrar algoritmos de cinemática inversa com diferente execução.

Contudo a transformação cinemática direta de coordenadas não apresenta dificuldades em sua resolução, o mesmo não acontece com a transformação cinemática inversa, que é complexa, não apresentando uma solução única. Para eliminar as indeterminações que aparecem no problema inverso utiliza-se geralmente a matriz Jacobiana.

Supondo que a matriz J é não-singular, a escolha

$$\dot{\theta} = J^{-1}(\theta)(\dot{x}_d + Ke), \tag{2.29}$$

conduz para o sistema linear equivalente

$$\dot{e} + Ke = 0. \tag{2.30}$$

Se K é uma matriz positiva definida, o sistema (2.30) é assintoticamente estavel, o erro tende para zero ao longo da trajetória com uma razão de convergência que depende dos autovalores da matriz K: quanto maior os autovalores mais rápida é a

convergencia.

2.2.5 Caminho e trajetória

O planejamento de trajetória cuja meta é gerar entradas de referência para o sistema de controle do movimento assegurar que o manipulador execute a trajetoria planejada. O usuário deve especificar um número de parâmetros para descrever a trajetória desejada. A seguir são apresentadas algumas técnicas para geração de trajetória, no caso quando o ponto inicial e final do caminho são especificados, denominada movimento *ponto a ponto*, e no caso quando uma sequência finita de pontos são especificadas ao longo do caminho, denominada *caminho do movimento*. Porém, antes é apresentada a diferença entre caminho e trajetória.

A exigência mínima para um manipulador é a capacidade de mover-se de uma configuração inicial para uma configuração final (vide Figura 2.8). A transição deve ser caracterizada por leis de movimento que requerem que os atuadores exerçam forças generalizadas na junta, que não violem os limites de saturação e não excite os modos ressonantes não-modelados da estrutura. Então é necessário projetar algoritmos de planejamento que gerem trajetórias convenientemente suaves.

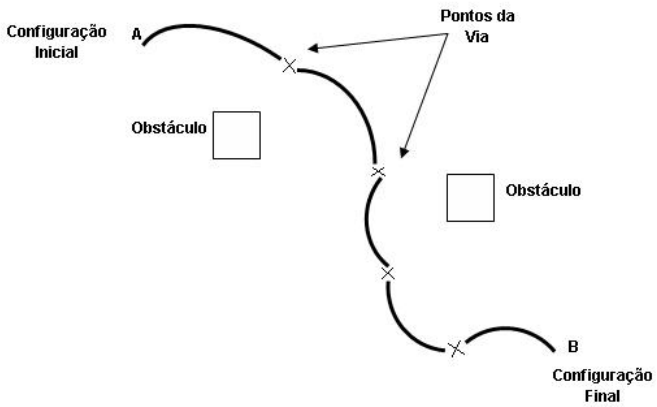

FIGURA 2.8: Trajetória entre dois pontos.

Para evitar confusão que geralmente é feita entre termos usados como sinônimo, a diferença entre *caminho* e *trajetória* é explicada. Um caminho denota o local dos pontos no espaço da junta, ou no espaço operacional, que o manipulador tem que seguir

na execução do movimento especificado. Por outro lado, uma trajetória é um caminho em que uma lei é especificada no tempo, como por exemplo em termos de velocidade e/ou aceleração em cada ponto (Sciavicco & Siciliano 1996).

Em princípio, pode ser estabelecido que as entradas para um algoritmo de planejamento de trajetória são: a descrição do caminho, as restrições do caminho, e as restrições impostas pela dinâmica do manipulador, enquanto as saídas são as trajetórias da junta (extremidade do efetuador) em termos de uma sequência de valores atingidos pela posição, velocidade e aceleração. Um caminho pode ser definido também no espaço da junta ou no espaço operacional.

Um caminho geométrico não pode ser especificado completamente pelo usuário por razões de complexidade. Geralmente, um número reduzido de parâmetros é especificado tais como pontos externos, pontos intermediários possíveis e primitivas geométricas de interpolação de pontos. Também a lei de movimento no tempo não é geralmente especificada em cada ponto do caminho geométrico, exceto antes de considerar a trajetória total no tempo, as restrições nas velocidades e acelerações máximas, e eventualmente a velocidade e aceleração especificada nos pontos de interesse particular. Com base no que foi dito acima, o algoritmo de planejamento de trajetória gera uma sequência de variáveis no tempo que descrevem a posição e orientação do efetuador em respeito as restrições impostas. Desde que a ação de controle no manipulador é executada no espaço da junta, um algoritmo de cinemática inversa é usado para reconstruir a sequência no tempo de variáveis da junta correspondendo a sequência no espaço operacional (Sciavicco & Siciliano 1996).

O planejamento de trajetória no espaço operacional geralmente permite considerar a presença de restrições no caminho; estas são devido as regiões do espaço de trabalho que não são permitidas para o manipulador, ou seja, devido a presença de obstáculos. De fato, tais restrições são geralmente melhores descritas no espaço operacional, desde que os pontos correspondentes no espaço da junta são difíceis de calcular.

Em consideração ao movimento na vizinhança de configurações singulares e presença de graus de liberdade redundantes, o planejamento de trajetória pode envolver problemas difíceis de resolver. Em tais casos, é aconselhável especificar o caminho no espaço da junta, ainda em termos de números de parâmetros redundantes. Portanto, uma sequência no tempo de variáveis da junta têm que ser garantido que satisfaça as

restrições impostas na trajetória (Sciavicco & Siciliano 1996).

2.2.6 Trajetórias no espaço das juntas

O movimento de um manipulador é geralmente descrito no espaço operacional em termos dos parâmetros de trajetória tais como a localização inicial e final do efetuador, localizações intermediárias, e a história no tempo ao longo dos caminhos geométricos particulares. Se é desejado planejar uma trajetória no espaço das juntas, os valores das variáveis das juntas têm que ser determinado primeiro da posição e orientação do efetuador. Sendo então necessário recorrer a um algoritmo de cinemática inversa, se o planejamento é feito off-line, ou diretamente medir as variáveis das juntas, se o planejamento é feito pela técnica de *teaching-by-showing*.

O algoritmo de planejamento gera uma função $\theta(t)$ (ângulos das juntas) interpolando os vetores das variáveis das juntas em cada ponto, em respeito as restrições impostas.

Em resumo neste caso é determinada a história no tempo dos ângulos das juntas, $\theta(t)$. Para isto deve-se considerar alguns pontos:

- a geração da trajetória não deve demandar uma carga computacional muito grande;
- posições e velocidades das juntas devem ser funções suaves no tempo;
- efeitos não-desejados deven ser minimizados, ou seja interpolação de pontos da trajetória com funções não suaves.

2.2.6.1 Movimento ponto a ponto

No movimento ponto a ponto, o manipulador se move de uma configuração inicial da junta para uma configuração final em um dado tempo, t_f. Neste caso, o caminho atual do efetuador é de interesse. O algoritmo deve gerar uma trajetória de acordo com às exigências descritas acima, e ser também capaz de otimizar alguns índices de desempenho quando a junta é movida para um outra posição (Sciavicco & Siciliano 1996).

Uma solução para escolher o movimento primitivo pode ser inspirado na análise de um problema de movimento incremental. Considerando I o momento de inércia de um corpo rígido em relação ao seu eixo de rotação. É necessário levar o ângulo θ de um valor inicial θ_i para um valor final θ_f em um tempo t_f. Existem infinitas soluções para este problema. Supondo que um elo é modelado pela equação:

$$Iw = \tau, \tag{2.31}$$

onde $\dot{\theta} = w$ e τ é o torque.

Sujeito à condição

$$\int_0^{t_f} w(t)dt = \theta_f - \theta_i, \tag{2.32}$$

de forma a minimizar a função de custo

$$\int_0^{t_f} \tau^2(t)dt. \tag{2.33}$$

Pode ser mostrado que a solução deste problema é do tipo:

$$w(t) = at^2 + bt + c. \tag{2.34}$$

Embora o modelo de um manipulador seja mais complicado, temos que esta é uma solução válida. Portanto a trajetória será gerada por:

$$\theta(t) = a_3t^3 + a_2t^2 + a_1t + a_0, \tag{2.35}$$

resultando em

$$\dot{\theta}(t) = 3a_3t^2 + 2a_2t + a_1, \tag{2.36}$$

$$\ddot{\theta}(t) = 6a_3t + 2a_2. \tag{2.37}$$

Então supondo que seja desejável velocidade inicial e final nulas tem-se que:

$$a_0 = \theta_i$$

$$
\begin{aligned}
a_1 &= \dot{\theta}_i \\
a_3 t_f^3 + a_2 t_f^2 + a_1 t_f + a_0 &= \theta_f \\
3a_3 t_f^2 + 2a_2 t_f + a_1 &= \dot{\theta}_f.
\end{aligned}
$$

Neste caso a velocidade tem um perfil parabólico e a aceleração tem um perfil linear com descontinuidades em $t = 0$ e $t = t_f$.

Caso seja desejado especificar também os valores iniciais e finais da aceleração será necessário considerar a trajetória como sendo descrita por um polinômio de $5a$ ordem tendo a forma:

$$
\theta(t) = a_5 t^5 + a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0. \tag{2.38}
$$

O perfil de velocidade trapezoidal é a opção mais desejada na prática poque permite verificar se a trajetória desejada se encontra dentro dos limites de velocidade e aceleração do equipamento.

2.2.6.2 Movimento do caminho

Em várias aplicações o caminho é descrito em termos de um número de pontos maior que dois. Por exemplo, para um simples movimento ponto a ponto de uma tarefa *pick-and-place*, pode ser válido especificar dois pontos intermediários entre os ponto inicial e final. Para aplicações mais complexas, pode ser conveniente especificar uma sequência de pontos, para garantir melhor monitoramento nas trajetórias executadas; os pontos são especificados mais densamente em segmentos do caminho onde os obstáculos têm que ser evitados ou se uma alta curvatura no caminho é esperada. Não deve ser esquecido que as variáveis das juntas correspondentes têm que ser calculadas das localizações do espaço operacional.

Portanto, o problema é gerar uma trajetória com N pontos, chamada de pontos do caminho, que são especificadas e têm que ser alcançadas pelo manipulador em certos instantes de tempo. Para cada variável da junta existem N restrições, e então uma pode usar um polinômio de ordem $(N-1)$. Esta escolha, de qualquer maneira, tem as seguintes desvantagens:

- não é possível especificar as velocidades inicial e final;
- quando a ordem de um polinômio aumenta, seu comportamento oscilatório aumenta, e isto pode levar a trajetórias que não são naturais para o manipulador;
- precisão numérica para o cálculo de coeficientes polinomiais diminuem com o aumento da ordem;
- o sistema resultante de equações de restrição é difícil de resolver;
- coeficientes polinomiais dependem de todos os pontos especificados, portanto, se é desejado mudar um ponto, todos eles têm que ser recalculados.

Estas desvantagens podem ser superadas se um número conveniente de interpolações polinomiais de menor ordem, contínuas nos pontos do caminho, são consideradas em lugar de uma única e alta ordem polinomial.

Para simplificar ainda mais o problema, é possível também encontrar interpolações polinomiais de ordem ainda mais baixas, que por exemplo três (cúbica), que determinem as trajetórias próximas aos pontos do caminho nos instantes de tempo dados.

Capítulo 3

Calibração e o Replanejamento de Trajetória

Este capítulo abordará a calibração e o replanejamento de trajetória, a necessidade de se abordar a calibração e o replanejamento é devido a incerteza na localização absoluta do manipulador em relação à localização original assumida para resolver o problema de planejamento de trajetória. Considerando que a incerteza é desconhecida, sua estimação é realizada através de um algoritmo baseado em mínimos quadrados que utiliza informações de sensores internos e externos. Contudo são abordados dois métodos de calibração para obtenção do erro de posicionamento entre E_b e E_{b_1} para assim resolver o problema de replanejamento da trajetória.

Para a obtenção do erro de posicionamento são propostos dois métodos de calibração. O primeiro método obtêm informações de sensores internos (encoders), que consiste em estando o manipulador na localização assumida como original medir os pontos da grade de calibração nesta localização, e assumindo que houve um deslocamento em relação ao sistema de coordenadas inercial, medir os pontos da grade de calibração em E_{b_1}. Após a determinação dos pontos de calibração da grade em respeito a E_b e E_{b_1}, estima-se o erro de posicionamento através de um algoritmo de mínimos quadrados. O segundo método utiliza informações de sensores externos, no caso uma câmera fixa no espaço de trabalho do manipulador com a função de sensor visual. Na verdade este método é complementar ao método utilizando sensores internos, porém os

pontos de calibração medidos com respeito as localizações original e atual são obtidos do primeiro método, contudo, esses pontos agora são vistos no ambiente da câmera. Entretanto devido a necessidade que existe em ambientes remotos onde a ação do homem é difícil ou até mesmo impossível o uso desse método se torna importante, já que se tem uma câmera olhando o manipulador executando uma determinada trajetória. É importante ressaltar que em ambos métodos considera-se apenas a posição, sendo por isso necessário um conjunto mínimo de três pontos para realizar a calibração.

Portanto conforme mencionado no capítulo 1 o problema é dividido em três etapas:

- a primeira etapa consiste no planejamento de trajetória, nesta o manipulador se encontra na localização original e para este percorrer uma determinada trajetória definida no espaço das juntas em relação a um sistema de coordenadas inercial tem que se realizar o planejamento desta trajetória, gerando entradas de referência para o movimento do sistema de controle que assegure que o manipulador execute a trajetória planejada;

- a segunda etapa consiste na estimação do erro de posicionamento entre E_b e E_{b_1} que é realizada utilizando um algoritmo baseado em mínimos quadrados, sendo essa etapa chamada de calibração;

- a terceira etapa consiste no replanejamento da trajetória, já que há incerteza na localização absoluta do manipulador em relação a localização original assumida para resolver o problema do planejamento de trajetória. Contudo com o robô encontrando-se deslocado em relação ao sistema de coordenadas inercial, e para que este execute a mesma trajetória percorrida com ele em E_b é necessário haver o replanejamento de trajetória, sendo então realizado através da obtenção do erro de posicionamento e da trajetória planejada com o manipulador em E_b.

3.1 Estimação do erro de posicionamento com sensor interno

Como já mencionado, a estimação do erro de posicionamento com sensor interno é dado medindo-se os pontos da grade de calibração em respeito a E_b (localização original) e E_{b_1} (localização atual), vide Figura 3.1, após a obtenção dessas medidas e através de um algoritmo baseado em mínimos quadrados o erro de posicionamento representado pela matriz de transformação homogênea T_{bb_1} é estimado. O número mínimo de pontos necessários para realizar a calibração são três, pois estamos levando em consideração apenas posição. Porém, como as medidas dos sensores apresentam ruídos, é medido um número de pontos de calibração superior ao número mínimo para uma melhor exatidão na obtenção dos parâmetros do erro de posicionamento.

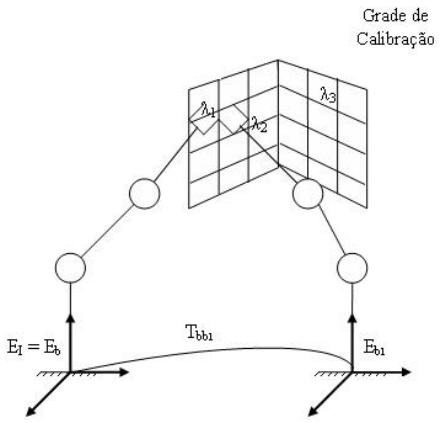

FIGURA 3.1: Representação dos pontos da Grade de calibração em E_b e E_{b_1} e o erro de posicionamento T_{bb_1}

Através da Figura 3.2, é vista a posição e orientação do sistema de coordenadas E_b, em respeito ao sistema de coordenadas E_{b_1}, representadas pela matriz de transformação homogênea T_{bb_1}. A parte translacional dessa matriz é composta pelo vetor posição: $[P_{bb_{1_x}}\ P_{bb_{1_y}}\ P_{bb_{1_z}}]^T$.

A convenção utilizada para a matriz de rotação é a representação XYZ dos ângulos de Euler, podendo também ser utilizada a representação de quaternions unitário. Dessa

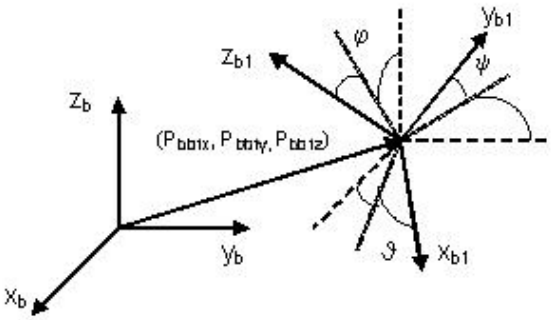

FIGURA 3.2: Definição Translacional e Rotacional do erro

forma a parte rotacional de T_{bb_1} é dada por uma sequência de três rotações em torno dos eixos z, y, e z novamente, sendo assim expressa pela sequência de produtos, conforme Equação (2.9), porém substituíndo as rotações elementares em (2.9) esta passa a ser dada de acordo com (2.10).

Contudo, os parâmetros do erro de posicionamento $P_{bb_{1_x}}, P_{bb_{1_y}}, P_{bb_{1_z}}, \varphi, \vartheta, \psi$ são função do sistema geométrico, representados por

$$x_{bb_1} = [P_{bb_1}\ \Phi_{bb_1}]^T, \tag{3.1}$$

onde

$$P_{bb_1} = \begin{bmatrix} P_{bb_{1_x}} \\ P_{bb_{1_y}} \\ P_{bb_{1_z}} \end{bmatrix}$$

e

$$\Phi_{bb_1} = \begin{bmatrix} \varphi \\ \vartheta \\ \psi \end{bmatrix}.$$

Portanto para se calcular x_{bb_1} recorre-se a um conjunto de três pontos de calibração, sendo determinado da seguinte forma:

- Considerando λ_1, λ_2 e λ_3 os pontos de calibração, conforme Figura 3.1. E estando o manipulador em E_b (localização adotada como original) o qual tocará esses pontos e os medirá em respeito a esta base. Em seguida, após um deslocamento o manipulador se encontra agora em E_{b_1} (localização atual), então da mesma forma tocará os referidos pontos e os medirá em respeito a nova base. Dessa forma o erro de posicionamento pode ser calculado resolvendo o sistema de equações para P_{bb_1} e R_{bb_1} :

$$P_{b\lambda_1} = P_{bb_1} + R_{bb_1} P_{b_1\lambda_1}, \tag{3.2}$$

$$P_{b\lambda_2} = P_{bb_1} + R_{bb_1} P_{b_1\lambda_2}, \tag{3.3}$$

$$P_{b\lambda_3} = P_{bb_1} + R_{bb_1} P_{b_1\lambda_3}, \tag{3.4}$$

onde $P_{b\lambda_1}, P_{b\lambda_2}$ e $P_{b\lambda_3} \in I\!R^3$ são a representação de cada ponto em E_b; $P_{b_1\lambda_1}, P_{b_1\lambda_2}$ e $P_{b_1\lambda_3} \in I\!R^3$ são a representação de cada ponto em E_{b_1}.

Contudo, como as medidas obtidas dos sensores internos (encoders) apresentam ruído, utiliza-se mínimos quadrados para se estimar os parâmetros de x_{bb_1}, pois se não existisse ruído seria suficiente utilizar três pontos de calibração (três pontos de calibração é o número mínimo de pontos) para se estimar x_{bb_1}, conforme mostrado anteriormente. A seguir pode-se vê o procedimento para se estimar o erro de posicionamento, baseado em mínimos quadrados:

- Com o manipulador na localização adotada como original, este tocará os pontos $\lambda_1, \lambda_2, \ldots, \lambda_n$ da grade de calibração e dessa forma se obtêm a posição desses pontos em relação a E_b, que podem ser representados por: $P_{b\lambda_1}, P_{b\lambda_2}, \ldots, P_{b\lambda_n}$.

- Em seguida considerando que o manipulador se encontra deslocado em relação a localização original, os pontos da grade de calibração serão agora medidos em relação a E_{b_1}, sendo expressos por: $P_{b_1\lambda_1}, P_{b_1\lambda_2}, \ldots, P_{b_1\lambda_n}$.

A representação desses pontos em E_b e E_{b_1} de acordo com a transformação homogênea de coordenadas ficam respectivamente

$$T_{b\lambda_n} = \begin{bmatrix} R_{b\lambda_n} & P_{b\lambda_n} \\ 0 & 1 \end{bmatrix} \tag{3.5}$$

e

$$T_{b_1\lambda_n} = \begin{bmatrix} R_{b_1\lambda_n} & P_{b_1\lambda_n} \\ 0 & 1 \end{bmatrix}. \tag{3.6}$$

Sendo finalmente T_{bb_1} expressa por:

$$T_{bb_1} = T_{b\lambda_n} T_{b_1\lambda_n}^{-1}. \tag{3.7}$$

Como é desejado encontrar T_{bb_1} utilizando um algoritmo baseado em mínimos quadrados, é necessário para isto definir uma função objetivo $f(x)$ que descreva o sistema em termos de seus parâmetros. Sendo T_{bb_1} representada conforme (3.1), o problema de minimização consiste em minimizar a função objetivo f de forma a obter seu mínimo. A função objetivo $f \in I\!R^{3n}$, onde n se refere ao número de pontos de calibração utilizados, vem da representação dos pontos de calibração na localização original, E_b, apresentando a seguinte forma

$$f = \begin{bmatrix} P_{b\lambda_1} - P_{bb_1} - R_{bb_1} P_{b_1\lambda_1} \\ \vdots \\ P_{b\lambda_n} - P_{bb_1} - R_{bb_1} P_{b_1\lambda_n} \end{bmatrix}, \tag{3.8}$$

onde se tem representado as posições dos pontos de calibração em E_b e E_{b_1} os quais $\in I\!R^3$; $R_{bb_1} \in I\!R^{3\times 3}$ é a matriz de rotação entre E_b e E_{b_1}, que pode ser representada através dos ângulos ZYZ de Euler.

Para este método de minimização é necessário também o uso do gradiente, o Jacobiano da função f, que são as derivadas parciais de primeira ordem com respeito aos

parâmetros P_{bb_1} e Φ_{bb_1}. O Jacobiano é dado por

$$J = \nabla f = \left[\frac{\partial f}{\partial P_{bb_1}} \; \frac{\partial f}{\partial \Phi_{bb_1}} \right]^T . \tag{3.9}$$

O jacobiano da função objetivo f, apresenta uma forma que facilita bastante os cálculos, note que $\frac{\partial f}{\partial P_{bb_1}} = I$. Então com essa facilidade apenas calcula-se o jacobiano de f em relação a Φ_{bb_1} da forma

$$\frac{\partial f}{\partial \Phi_{bb_1}} = \begin{bmatrix} \frac{\partial R_{bb_1}}{\partial \varphi} P_{b_1\lambda_1} & \frac{\partial R_{bb_1}}{\partial \vartheta} P_{b_1\lambda_1} & \frac{\partial R_{bb_1}}{\partial \psi} P_{b_1\lambda_1} \\ \vdots & \vdots & \vdots \\ \frac{\partial R_{bb_1}}{\partial \varphi} P_{b_1\lambda_n} & \frac{\partial R_{bb_1}}{\partial \vartheta} P_{b_1\lambda_n} & \frac{\partial R_{bb_1}}{\partial \psi} P_{b_1\lambda_n} \end{bmatrix}, \tag{3.10}$$

onde $\frac{\partial R_{bb_1}}{\partial \varphi}, \frac{\partial R_{bb_1}}{\partial \vartheta}, \frac{\partial R_{bb_1}}{\partial \psi} \in I\!R^{3\times 3}$ e a dimensão de (3.10) $\in I\!R^{3n\times 3}$.

Finalmente o Jacobiano assume a forma

$$J = \begin{bmatrix} -I_{(3\times 3)} & -\frac{\partial R_{bb_1}}{\partial \varphi} P_{b_1\lambda_1} & -\frac{\partial R_{bb_1}}{\partial \vartheta} P_{b_1\lambda_1} & -\frac{\partial R_{bb_1}}{\partial \psi} P_{b_1\lambda_1} \\ \vdots & \vdots & \vdots & \vdots \\ -I_{(3\times 3)} & -\frac{\partial R_{bb_1}}{\partial \varphi} P_{b_1\lambda_n} & -\frac{\partial R_{bb_1}}{\partial \vartheta} P_{b_1\lambda_n} & -\frac{\partial R_{bb_1}}{\partial \psi} P_{b_1\lambda_n} \end{bmatrix}. \tag{3.11}$$

Nota-se que o Jacobiano é uma matriz $\in I\!R^{3n\times 6}$ que será invertida para se encontrar x_{bb_1}, para sua inversão utiliza-se a pseudoinversa a esquerda pois o posto é dado por coluna. Portanto a correção nos valores de x_{bb_1} a cada iteração k é então realizada fazendo

$$x_{k+1} = x_k - \alpha[J(k)^T J(k)]^{-1} J(k)^T f(k). \tag{3.12}$$

Dessa forma a equação (3.12) pode ser considerada como um problema de mínimos quadrados não-linear e assim resolvida com base no método de Newton.

3.2 Algoritmo de calibração - sensores internos

Nesta seção é mostrado o algoritmo utilizado para encontrar os parâmetros de posição e orientação de x_{bb_1}. Os principais passos são:

1. Selecionar um conjunto de pontos de entrada;
2. Medir os pontos de calibração nos sistemas de coordenadas E_b e E_{b_1}, e dessa forma obter os pontos $P_{b\lambda_n}$ e $P_{b_1\lambda_n}$ relativo aos respectivos sistemas de coordenadas;
3. Selecionar uma solução inicial x_0 e um critério de término, que neste caso é o número de iterações. Calcular $J(0)$ e $f(0)$. Se o erro relativo entre cada iteração for menor até chegar a um nível em que a diferença entre os erros atual e anterior forem muito pequenas ir para o passo 6, caso contrário fazer $k = 0$ e ir para o próximo passo;
4. Calular a matriz do jacobiano $J(k)$ e a função principal $f(k)$;
5. Calcular o vetor de ajustes $x_{k+1} = x_k - \alpha[J(k)^T J(k)]^{-1} J(k)^T f(k)$. Caso tenha sido atingido $x_k = x_{bb_1}$, pular para o passo 6, caso contrário faz-se $k = k + 1$ e voltar ao passo 4;
6. Os três parâmetros de posição e os três parâmetros de orientação ($P_{bb_{1_x}}, P_{bb_{1_y}}, P_{bb_{1_z}}$, φ, ϑ, ψ) são finalmente determinados.

A computação de x_{bb_1} requer o conhecimento de no mínimo, três pontos, a escolha adequada desses pontos é de grande importância, uma vez que uma escolha inadequada, como um ponto fora da área de trabalho do manipulador pode levar a divergências, sendo por isso utilizada uma grade de calibração perfeitamente conhecida.

A determinação de x_{bb_1} faz uso de um método de mínimos quadrados iterativo partindo de uma solução inicial, onde esta estimativa deve garantir a convergência no caso para o mínimo global. Este método descrito apresenta limitações na determinação de x_{bb_1}, tais como a escolha de pontos de forma aleatória ou randômica podem levar a divergências, pois estes podem está fora da área de trabalho do manipulador, e a tendência ao mal-condicionamento da matriz do jacobiano.

Como já foi dito anteriormente, o processo de determinação de x_{bb_1} faz uso de um método não-linear. Este método consiste em um algoritmo de mínimos quadrados resolvido iterativamente a partir de uma estimativa inicial. A cada iteração k do algoritmo é computado x_{bb_1}, mediante a solução de (3.12).

Para se evitar o mal-condicionamento da matriz $J^T J$, que podem levar a falhas no processo de inversão dessa matriz existem algumas estratégias que podem ser adotadas para se lidar com esse mal-condicionamento da matriz $J^T J$. Uma estratégia seria efetuar pequenas modificações em alguns elementos dessa matriz, de forma a evitar que seja atingido o estado de quase-singularidade. O método de Levenber-Marquardt (vide apêndice A) apresenta uma estratégia para contornar esse problema, consiste em modificar os valores dos elementos da diagonal principal $J^T J$ em função do posto dessa matriz. Somando-se a $J^T J$ um fator μ, o valor do determinante dessa matriz é alterado. Portanto escolhendo-se um valor apropriado para μ, evita-se que $J^T J$ se torne mal-condicionada.

Tipicamente, um valor razoável para μ, se fixo, situa-se na ordem de 10^{-4}. Para o caso de μ ser modificado a cada iteração, adota-se a estratégia referida no anexo A, para calcular μ, analisando-se para isso o valor da função objetivo f.

Outras modificações podem ser realizadas no algoritmo, como incluir uma forma de controlar ou pelo menos limitar o avanço do algoritmo a cada iteração, ou seja, controlar a forma como é feita a atualização dos parâmetros. Podendo esse controle ser realizado incluíndo um fator α_k. O valor desse fator deve variar entre 0 e 1, e pode ser mantido fixo para todas as iterações, sendo escolhido uma valor apropriado. Contudo, a experiência tem demonstrado que melhores resultados são obtidos adotando-se α_k variável, de forma a garantir a melhor atualização dos parâmetros a cada iteração. Então o fator α_k pode ser escolhido de acordo com a regra de Armijo.

Na determinação dos parâmetros de T_{bb_1}, os seis parâmetros que são estimados pertencem a dois grupos distintos com diferentes características que são: os parâmetros de translação (posição) e os parâmetros de orientação. A diferença de unidades de medida entre ambos os grupos é óbvia (metros versus radianos), porém pode ocorrer também uma diferença de escala entre os grupos: enquanto os parâmetros de rotação estão limitados a pequenos valores (devido principalmente ao fato de os ângulos não excederem 2π), os parâmetros de posição podem apresentar valores bem maiores, pois

teoricamente não há limites para a variação em metros (exceto as limitações físicas do robô).

3.2.1 Simulação

Nesta seção apresenta-se os modelos cinemáticos utilizados e os resultados de simulações obtidos com a implementação do primeiro método apresentado neste capítulo.

As simulações foram realizadas em *Matlab* (Mathworks, Inc) utilizando o toolbox de robótica, onde tentou-se recriar nas simulações o problema em estudo.

3.2.1.1 Manipulador robótico

Para as simulações do erro de posicionamento com sensor interno, o modelo do robô manipulador utilizado baseia-se no modelo cinemático de um manipulador robótico Zebra Zero (IMI Inc.). O Zebra Zero é um robô articulado de seis graus de liberdade, onde suas juntas são todas rotacionais. Este robô está equipado com uma plataforma móvel de três graus de liberdade, a qual em conjunto com o manipulador produzem nove graus de liberdade. O seu modelo cinemático é dado através dos parâmetros α_i, a_i, θ_i e d_i, segundo a convenção standard de Denavit-Hatenberg, vide Tabela 3.1.

Junta i	α_i [rad]	a_i [mm]	θ_i [rad]	d_i [mm]
1	$\frac{\pi}{2}$	0	0	0
2	0	l_1	0	0
3	$-\frac{\pi}{2}$	0	0	0
4	$\frac{\pi}{2}$	0	l_2	0
5	$-\frac{\pi}{2}$	0	0	0
6	0	0	0	0

TABELA 3.1: Parâmetros de Denavit-Hatenberg para o Manipulador Zebra Zero.

3.2.1.2 Grade de calibração

Os pontos de calibração utilizados são mostrados na Tabela 3.2, cuja grade de calibração pode ser vista através das Figuras 3.3 e 3.4, onde os pontos desta grade se referem ao sistemas de coordenadas E_b e E_{b_1}. O manipulador tocando os pontos desta grade podem ser vistos nas Figuras 3.5 e 3.6 com o robô nas respectivas localizações original e atual. É importante ressaltar que o termo "tocar"os pontos se refere a medir a posição desses pontos em respeito ao sistema de coordenadas especificado.

Ponto [mm]	x	y	z
λ_1	228.600	0	379.4000
λ_2	211.2352	98.4808	379.4000
λ_3	193.8704	196.9616	379.4000
λ_4	228.6000	0	279.4000
λ_5	211.2352	98.4808	279.4000
λ_6	193.8704	196.9616	279.4000
λ_7	228.6000	0	179.4000
λ_8	211.2352	98.4808	196.9616
λ_9	193.8704	196.9616	179.4000
λ_{10}	193.8704	-196.9616	379.4000
λ_{11}	211.2352	-98.4808	379.4000
λ_{12}	228.6000	0	379.4000
λ_{13}	193.8704	-196.9616	279.4000
λ_{14}	211.2352	-98.4808	279.4000
λ_{15}	228.6000	0	279.4000
λ_{16}	193.8704	-196.9616	179.4000
λ_{17}	211.2352	-98.4808	179.4000
λ_{18}	228.6000	0	179.4000

Tabela 3.2: Pontos da grade de calibração.

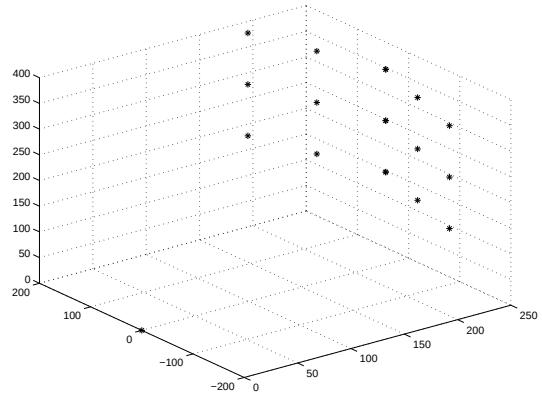

FIGURA 3.3: Grade de Calibração - Representação dos pontos no sistema de coordenadas E_b.

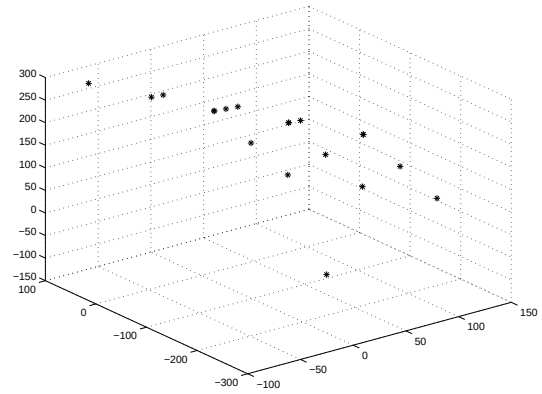

FIGURA 3.4: Grade de Calibração - Representação dos pontos no sistema de coordenadas E_{b_1}.

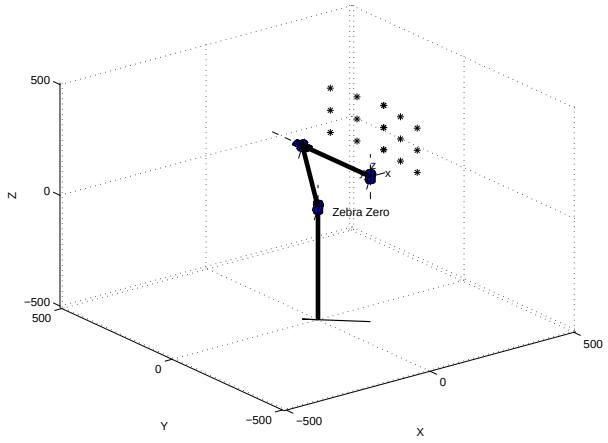

FIGURA 3.5: Manipulador tocando os pontos da grade de calibração - Manipulador em E_b.

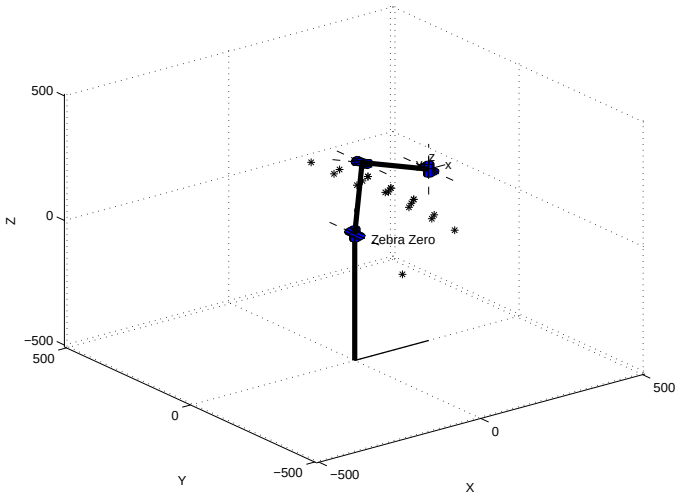

FIGURA 3.6: Manipulador tocando os pontos da grade de calibração - Manipulador em E_{b_1}.

3.2.2 Resultados de simulação

Nesta seção, apresenta-se resultados de simulação para ilustrar o desempenho do método proposto.

O processo de determinação dos três parâmetros de translação e dos três parâmetros de orientação determinam o posicionamento do sistema de coordenadas E_{b_1} em relação ao sistema de coordenadas E_b, em outras palavras determinam o erro de posicionamento da localização assumida como original. Usando as fórmulas apresentadas na seção 1.1, a grade de calibração e valores apropriados para α, obtêm-se os parâmetros do erro de posicionamento. A Tabela 3.3 apresenta os valores ideais, ou seja, sem a presença de ruído, onde os parâmetros de posição são expressos em metros, e os parâmetros de orientação em radianos. A Figura 3.7 ilustra esses parâmetros, utilizando os pontos da grade de calibração.

Parâmetros: Posição/Orientação	Valor Ideal
$P_{bb_{1_x}}$	50
$P_{bb_{1_y}}$	100
$P_{bb_{1_z}}$	150
φ	-0.7835
ϑ	0.1233
ψ	0.7873

TABELA 3.3: Parâmetros de posição e orientação do erro de posicionamento.

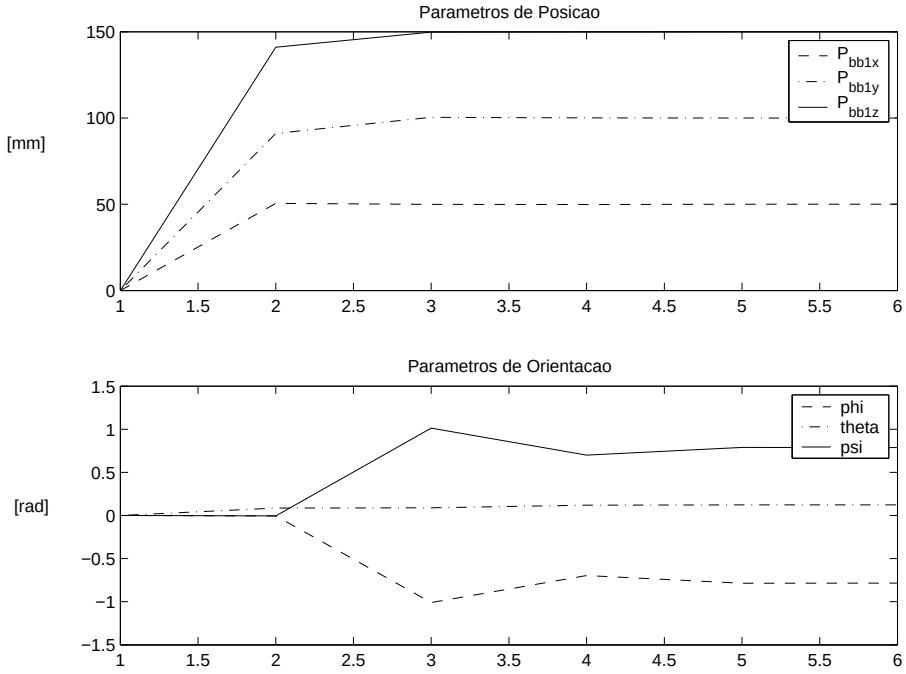

FIGURA 3.7: Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, estimação sem ruído.

Como na prática as medidas obtidas dos sensores internos (encoders) apresentam ruído, acrescentou-se um ruído nos ângulos das juntas para emular uma situação real. O ruído adicionado nos ângulos das juntas é um ruído randômico gerado pela função *rand* do *Matlab* que possui uma distribuição uniforme em um intervalo de 0 a 1, de amplitude $\frac{2\pi}{600}$. A simulação apresentada na Figura 3.7 foi realizada com o conjunto completo de pontos da grade de calibração, vide Tabela 3.2, porém nesta simulação não há presença de ruído, sendo assim uma situação ideal, não acontecendo na prática com as medidas obtidas dos sensores internos. Apesar de se estar considerando apenas posição desses pontos em respeito a E_b e E_{b_1} o número mínimo de pontos de calibração necessários para se estimar x_{bb_1} são três. Contudo devido ao ruído presente nas medidas, recorreu-se a um conjunto de pontos maior com o intuito de ter-se melhor conhecimento da trajetória a ser seguida.

De forma a verificar o comportamento do método proposto na estimação do erro de posicionamento, foram realizadas simulações com diferentes número de pontos de calibração. Os pontos utilizados, considerando os rótulos indicados na Tabela 3.2, foram os seguintes: 18 pontos (todos); 10 pontos (λ_1 a λ_{10}); 5 pontos (λ_1 a λ_5) e 3 pontos (λ_1 a λ_3). Atravé da Tabela 3.4 pode-se observar os parâmetros do erro de posicionamento para cada conjunto de pontos.

Parâmetro	$N = 18$	$N = 10$	$N = 5$	$N = 3$
$P_{bb_{1_x}}$	51.6591	50.5235	44.8775	49.5036
$P_{bb_{1_x}}$	99.7184	99.1428	95.1136	99.6308
$P_{bb_{1_x}}$	147.6440	152.2172	153.0788	150.1124
φ	-0.8749	-0.7467	-0.6363	-0.7607
ϑ	0.1069	0.1235	0.1351	0.1145
ψ	0.8781	0.7472	0.6570	0.7547

TABELA 3.4: Parâmetros do erro de posicionamento, com diferentes conjuntos de pontos.

As Figuras 3.8 a 3.11 ilustram os parâmetros do erro de posicionamento do manipulador estimados entre E_b e E_{b_1}, apartir do algoritmo apresentado, e considerando o ruído em questão. Através das simulações pode ser observado que apesar das diferenças

que há entre as unidades de medidas de ambos os grupos, não houve problema quanto aos parâmetros de orientação influenciarem os parâmetros de posição, e vice-versa, pois as diferenças de escala podem conduzir a singularidades. Sendo importante ressaltar que os pontos destinados a calibração devem ser tomados dentro de uma estrutura conhecida, sendo por isso utilizada uma grade de calibração, pois uma escolha aleatória desses pontos poderia levar a divergência se um ou mais pontos estiverem fora do espaço de trabalho do manipulador. Assim como os parâmetros de orientação do erro podem não ser bem definidos com um número de pontos inferior a três, e como se está utilizando os ângulos de Euler para parametrizar a orientação deve-se tomar cuidado com a escolha do ponto de partida, pois os ângulos de Euler apresentam singularidade com o ângulo ϑ sendo um valor nulo. Outra limitação desse método é a respeito também dos ângulos de Euler, cujo valores acima de 2π levam a singularidades.

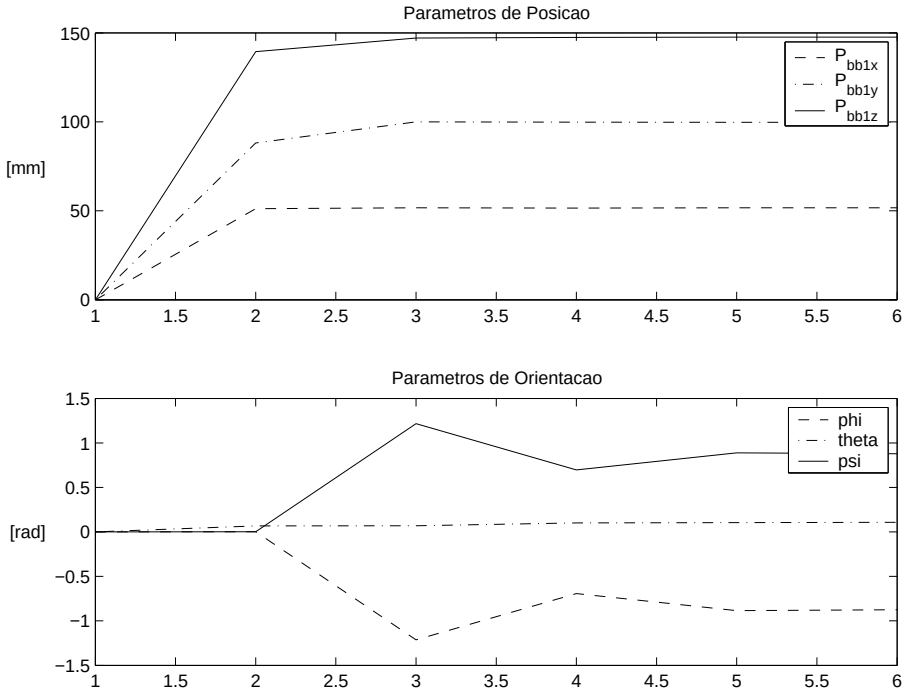

FIGURA 3.8: Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, para $N = 18$ pontos da grade de calibração.

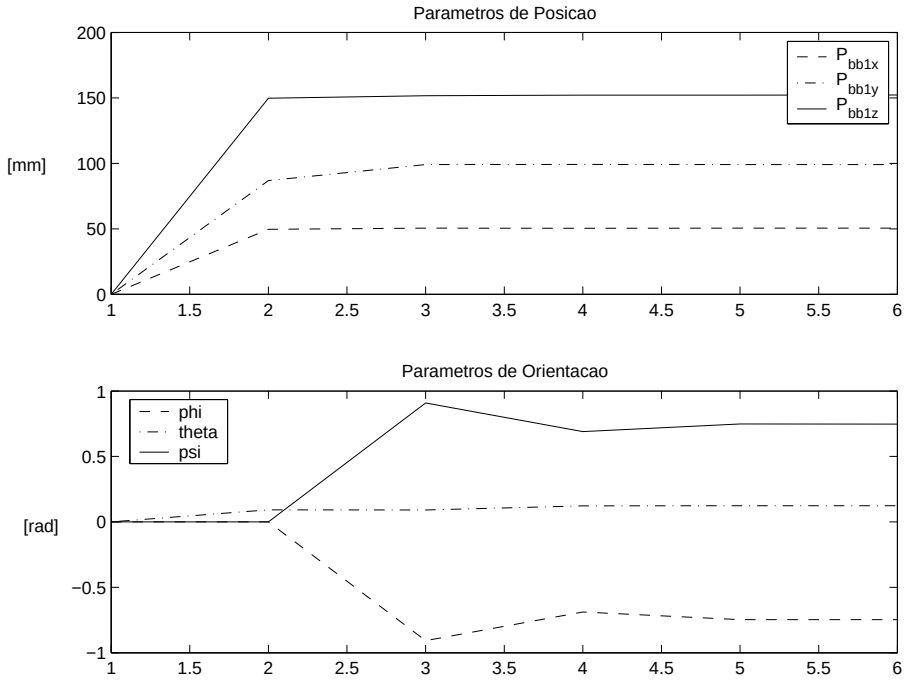

FIGURA 3.9: Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, para $N = 10$ pontos da grade de calibração.

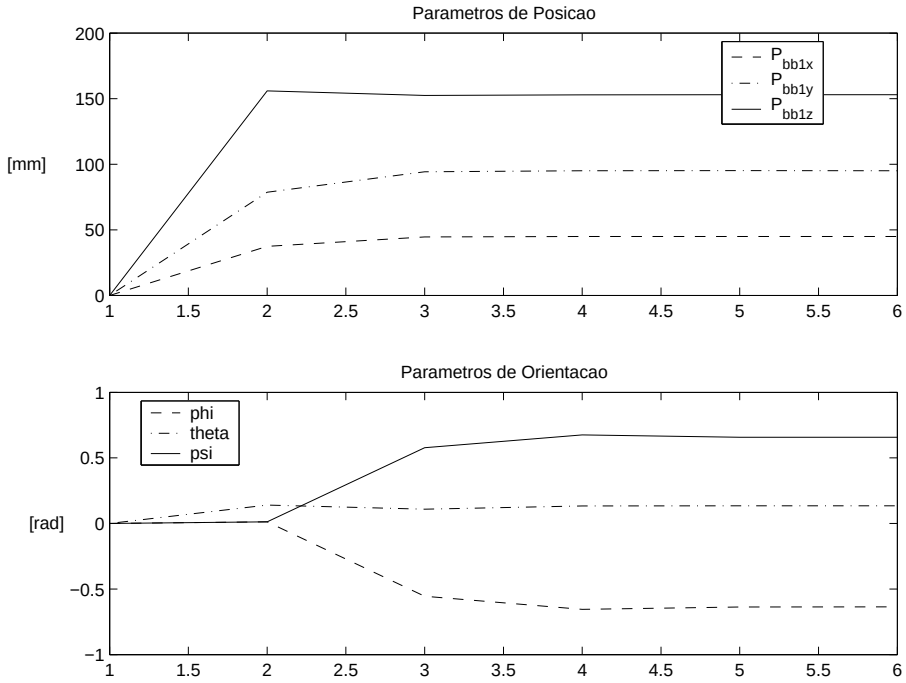

FIGURA 3.10: Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, para $N = 5$ pontos da grade de calibração.

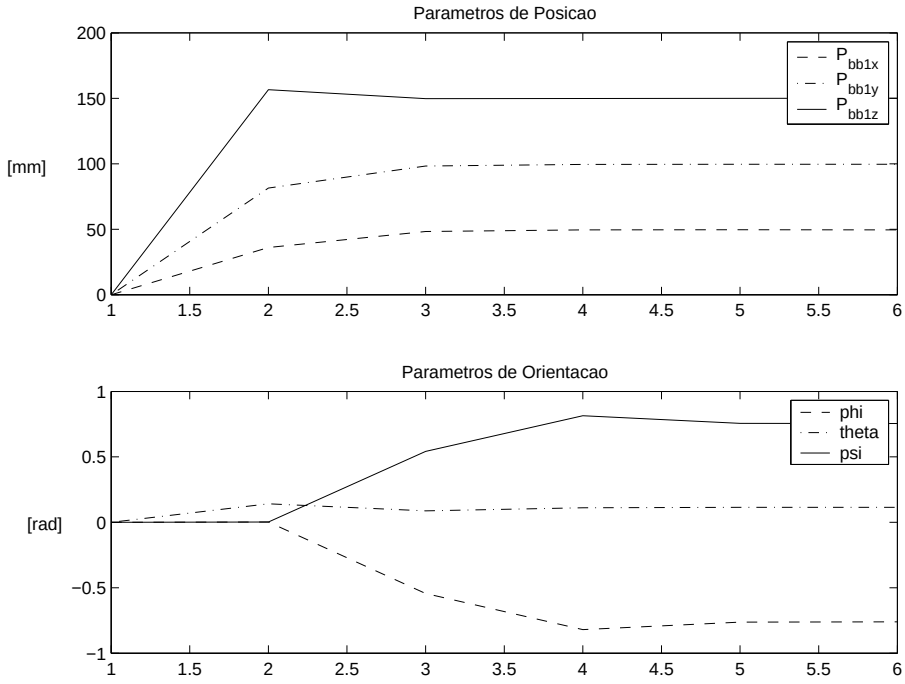

FIGURA 3.11: Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensores internos, para $N = 3$ pontos da grade de calibração.

Através das Tabelas 3.6 a 3.9 pode ser observado os parâmetros estimados do erro de posicionamento para um conjunto diferente de pontos de calibração em comparação com o valor ideal do erro, como também mostram o erro absoluto. Apesar dos resultados de simulações mostrados nas Figuras 3.8 a 3.11 estarem pertubados por ruído, os parâmetros estimados se mostraram dentro do esperado, devido ao ruído utilizado não exercer grandes influências nestes.

A Tabela 3.5 mostra o erro de posição do erro de posicionamento e o erro relativo da posição para cada conjunto de pontos já especificados, sendo o erro de posição dado da seguinte forma:

$$|P_{bb_1}(Valor\ ideal) - P_{bb_1}(Valor\ estimado)|$$

Erro de Posição	Erro Relativo da Posição	N pontos de calibraçao
2.8950	1.5474%	$N = 18$
2.4341	1.3011%	$N = 10$
7.7198	4.1264%	$N = 5$
0.6288	0.3361%	$N = 3$

TABELA 3.5: Erro de posiçao e Erro relativo da posição - Sensor interno

Portanto com a estimação do erro de posicionamento nesta etapa de calibração, pode-se realizar o replanejamento de trajetória e assim percorrer a mesma trajetória já planejada com o robô na localização original.

Parâmetro	Valor Real	Valor Estimado: $N = 18$	Erro Absoluto
$P_{bb_{1_x}}$	50	51.6591	1.6591
$P_{bb_{1_y}}$	100	95.1136	0.2816
$P_{bb_{1_z}}$	150	147.6440	2.3560
φ	-0.7835	-0.8749	0.0914
ϑ	0.1233	0.1069	0.0164
ψ	0.7873	0.8781	0.0908

TABELA 3.6: Erro de posicionamento estimado utilizando o primeiro método - sensores internos para $N = 18$ pontos da grade de calibração.

Parâmetro	Valor Real	Valor Estimado: $N = 10$	Erro Absoluto
$P_{bb_{1_x}}$	50	50.5235	0.5235
$P_{bb_{1_y}}$	100	99.7184	0.8572
$P_{bb_{1_z}}$	150	152.2172	2.2172
φ	-0.7835	-0.7467	0.0368
ϑ	0.1233	0.1235	0.0002
ψ	0.7873	0.7472	0.0401

TABELA 3.7: Erro de posicionamento estimado utilizando o primeiro método - sensores internos para $N = 10$ pontos da grade de calibração.

Parâmetro	Valor Real	Valor Estimado: $N = 5$	Erro Absoluto
$P_{bb_{1_x}}$	50	44.8775	5.1225
$P_{bb_{1_y}}$	100	99.7184	4.8864
$P_{bb_{1_z}}$	150	153.0788	3.0788
φ	-0.7835	-0.6363	0.1472
ϑ	0.1233	0.1351	0.0118
ψ	0.7873	0.6570	0.1303

TABELA 3.8: Erro de posicionamento estimado utilizando o primeiro método - sensores internos para $N = 5$ pontos da grade de calibração.

Parâmetro	Valor Real	Valor Estimado: $N = 3$	Erro Absoluto
$P_{bb_{1_x}}$	50	49.5036	0.4964
$P_{bb_{1_y}}$	100	99.6308	0.3692
$P_{bb_{1_z}}$	150	150.1124	0.1124
φ	-0.7835	-0.7607	0.0266
ϑ	0.1233	0.1145	0.0093
ψ	0.7873	0.7547	0.0326

TABELA 3.9: Erro de posicionamento estimado utilizando o primeiro método - sensores internos para $N = 3$ pontos da grade de calibração.

3.3 Identificação do erro de posicionamento com um sensor visual

Neste método o erro de posicionamento é estimado utilizando uma câmera fixa no espaço de trabalho do manipulador com a função de sensor visual, vide Figura 3.13. A utilização de câmeras como sensor visual são importantes para que o robô opere em ambientes desconhecidos ou remotos, e compartilhem o espaço de trabalho com operadores humanos ou até mesmo com outros robôs. Sendo esta a finalidade deste método que complementa o método proposto na seção 3.1.

Como já mencionado é necessário para estimação do erro de posicionamento x_{bb_1} uma quantidade mínima de três pontos da grade de calibração, pois está sendo considerado apenas posição. Dessa forma o manipulador tocará esses pontos de calibração que serão medidos primeiramente em relação a localização original E_b, vide Figura 3.12. Em seguida esses pontos serão medidos em relação a localização atual E_{b_1}. Porém, após as medidas desses pontos,e como pode ser observado são obtidos conforme o método utilizando sensores internos, serão agora vistos no ambiente da câmera. Após se obter os novos pontos de calibração, vistos pela câmera, o procedimento para estimar x_{bb_1} segue o mesmo raciocínio do método anteriormente abordado. A Figura 3.13 ilustra a situação descrita.

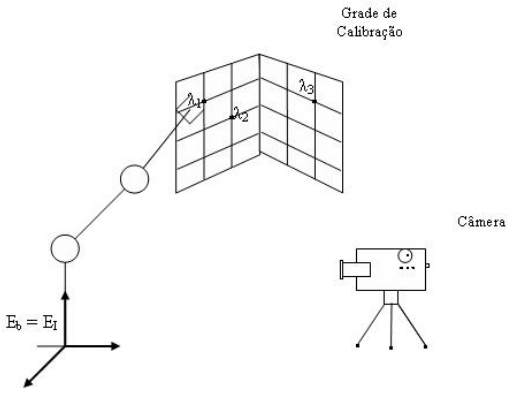

FIGURA 3.12: Medindo os pontos de calibração da grade, através de um sensor visual (câmera).

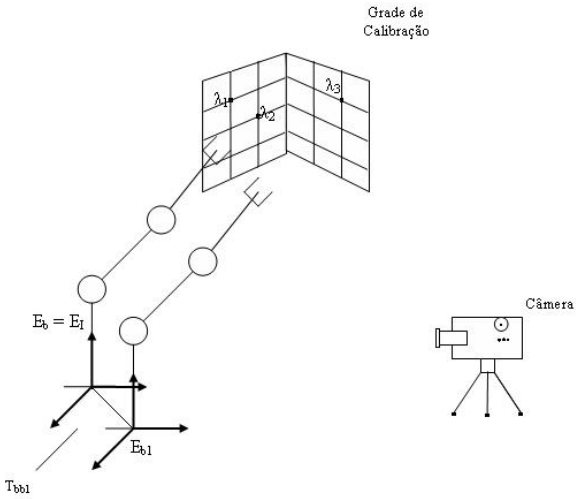

FIGURA 3.13: Manipulador deslocado em relação a E_b, erro de posicionamento estimado através do sensor visual.

O primeiro método já abordado nas seções iniciais deste capítulo e o segundo ainda a ser abordado não são concorrentes mais sim complementares, pois as medidas dos pontos de calibração da grade seja em relação a E_b e E_{b_1} são medidos através de sensores internos, ou seja, os encoders, que estão localizados nas juntas do manipulador. Portanto o procedimento para obtenção dos novos pontos de calibração agora visualizados pela câmera seguem o procedimento descrito ao longo do texto.

De acordo com a Figura 3.14 o ponto λ no plano de trabalho é expresso em relação ao sistema de coordenadas da câmera por

$$P_{c\lambda} = R_{cb}(P_{b\lambda} - P_{bc}), \tag{3.13}$$

onde $P_{b\lambda}$ representa o vetor posição de λ no sistema de coordenadas na base; P_{bc} vetor posição ou translação entre o sistema de coordenada do robô e o sistema de coordenadas da câmera; R_{cb} matriz de rotação. Contudo a equação (3.13) pode ser representada por:

$$\begin{bmatrix} x_{c\lambda} \\ y_{c\lambda} \\ z_{c\lambda} \end{bmatrix} = R_{cb} \left(\begin{bmatrix} x_{b\lambda} \\ y_{b\lambda} \\ z_{b\lambda} \end{bmatrix} - \begin{bmatrix} x_{bc} \\ y_{bc} \\ z_{bc} \end{bmatrix} \right). \tag{3.14}$$

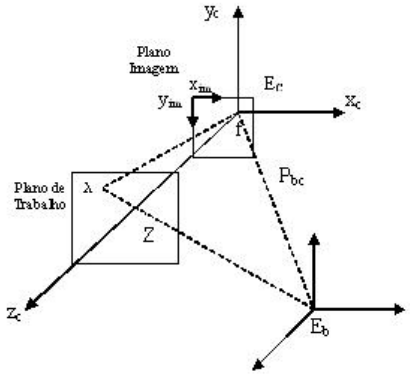

FIGURA 3.14: Representação dos sitemas de coordenadas usados com sensor visual, estando o robô na base b.

A partir desta representação para o ponto 3D no referencial da câmera, e devido ao fenômeno de refração, o ponto no sistema de coordenadas da câmera é transformado em um ponto no plano imagem via projeção em perspectiva (vide Figura 3.15).

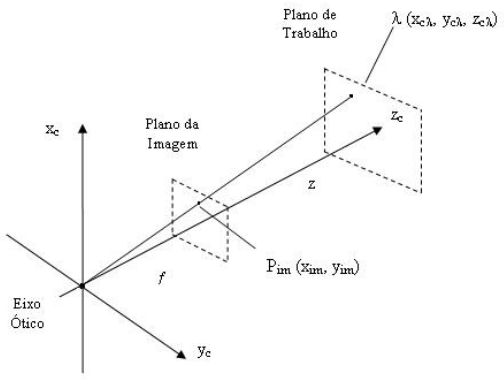

FIGURA 3.15: Modelo de projeção perspectiva de uma câmera.

De acordo com a Figura 3.15 as coordenadas de projeção são dadas por

$$\begin{bmatrix} x_{im} \\ \\ y_{im} \end{bmatrix} = \frac{f}{z_{c\lambda}} \begin{bmatrix} x_{c\lambda} \\ \\ y_{c\lambda} \end{bmatrix}. \tag{3.15}$$

Considerando que uma câmera CCD digital é utilizada para extrair as coordenadas do ponto λ de calibração no plano imagem, a relação (3.15) assume a forma

$$\begin{bmatrix} x_{im} \\ \\ y_{im} \end{bmatrix} = \frac{f}{f+z} \begin{bmatrix} \alpha_1 & 0 \\ \\ 0 & \alpha_2 \end{bmatrix} \begin{bmatrix} x_{c\lambda} \\ \\ y_{c\lambda} \end{bmatrix}, \tag{3.16}$$

onde f é a distância focal da câmera; α_i $(i = 1, 2)$ são os fatores de escala da câmera [pixels/mm]; $[x_{im} \;\; y_{im}]^T$ são as coordenadas de projeção do ponto λ no plano imagem da câmera.

Com um conjunto de N pontos da grade de calibração $[x_{\lambda_n} \; y_{\lambda_n} \; z_{\lambda_n}]^T$ e o correspondente conjunto de pontos 2D $[x_{im_n} \; y_{im_n}]^T$ no referencial da imagem, com $n = 1, \ldots, N$, são usados para a estimação da orientação e da posição do erro de posicionamento entre a localização original e atual. Na verdade com um conjunto suficiente pontos de calibração no caso três e suas respectivas projeções no plano imagem são suficientes para se estimar o erro, porém utiliza-se também uma quantidade acima de três para melhor exatidão do erro de posicionamento.

Seja a translação ou posição entre o referencial da base do manipulador e o referencial da câmera indicada por $P_{bc} = [x_{bc} \; y_{bc} \; z_{bc}]^T$. O ponto de calibração $\lambda_n = [x_{\lambda_n} \; y_{\lambda_n} \; z_{\lambda_n}]^T$ da grade no referencial da câmera se torna,

$$P_{c\lambda_n} = R_{cb}(P_{b\lambda} - P_{bc}), \tag{3.17}$$

ou em termos de suas coordenadas

$$\begin{bmatrix} x_{c\lambda_n} \\ y_{c\lambda_n} \\ z_{c\lambda_n} \end{bmatrix} = R_{cb} \left(\begin{bmatrix} x_{b\lambda_n} \\ y_{b\lambda_n} \\ z_{b\lambda_n} \end{bmatrix} - \begin{bmatrix} x_{bc} \\ y_{bc} \\ z_{bc} \end{bmatrix} \right). \tag{3.18}$$

Com isso a projeção em perspectiva do ponto λ de calibração no plano imagem é dada por

$$\begin{bmatrix} x_{im_n} \\ y_{im_n} \end{bmatrix} = \frac{f}{z_{c\lambda_n}} \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} \begin{bmatrix} x_{c\lambda_n} \\ y_{c\lambda_n} \end{bmatrix}, \tag{3.19}$$

onde $z_{c\lambda_n}$ é dado por

$$z_{c\lambda_n} = \left(R_{cb} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right)^T (P_{b\lambda_n} - P_{bc}), \tag{3.20}$$

ou através da Figura (3.15) $z_{c\lambda}$ é

$$z_{c\lambda_n} = f + z_n, \tag{3.21}$$

contudo

$$z_n = z_{c\lambda_n} - f, \tag{3.22}$$

onde z_n é a profundidade relativa medida entre o plano da imagem e o espaço de trabalho do robô.

Reescrevendo (3.19) da forma

$$P_{im_n} = \frac{f}{z_{c\lambda_n}} \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} P_{c\lambda_n}, \tag{3.23}$$

e substituíndo (3.17) em (3.23)

$$P_{im_n} = \frac{f}{z_{c\lambda_n}} \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} R_{cb}(P_{b\lambda_n} - P_{bc}), \tag{3.24}$$

$$P_{im_n} = \frac{f}{z_{c\lambda_n}} \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} R_{cb}P_{b\lambda_n} - \frac{f}{z_{c\lambda_n}} \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} R_{cb}P_{bc}, \tag{3.25}$$

finalmente chega-se a

$$P_{im_n} = K_{p_n}(P_{b\lambda_n})_{2D} - P_{c0}. \tag{3.26}$$

Em geral, a transformação de coordenadas P_{bc} é não- homogênea. Entretanto, sem perda de generalidade, pode-se assumir que as origens dos sistemas de coordenadas da câmera e da base do robô são coincidentes fazendo $P_{c0} = 0$, ficando (3.26)

$$P_{im_n} = K_{p_n}(P_{b\lambda_n})_{2D}, \tag{3.27}$$

sendo

$$K_{p_n} = \frac{f}{z_{c\lambda_n}} \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} R_{cb}, \tag{3.28}$$

onde P_{im} representa os pontos medidos no plano da imagem; P_{c0} é um termo constante que depende da posição do sistema de coordenadas da câmera em relação ao sistema de coordenadas do robô; K_{p_n} é a matriz de transformação do espaço operacional para o espaço da câmera, com $n = 1, \ldots, N$.

Contudo, através de (3.27) pode-se obter os $(P_{b\lambda_n})_{2D}$ que é um representação bidimensional dos pontos de calibração em E_b. Porém, sabe-se que o plano da imagem é bidimensional e então para representar esses pontos em 3D, acrescenta-se a este vetor a terceira coordenada $z_{c\lambda_n}$. E dessa forma volta-se a base do robô através de

$$P_{b\lambda_n novo} = R_{cb}^T (P_{b\lambda_n})_{3D} + P_{bc}. \tag{3.29}$$

O procedimento para se obter os pontos de calibração em E_{b_1} é análogo ao manipulador em E_b, sendo portanto

$$P_{b_1\lambda_n novo} = R_{cb}^T (P_{b_1\lambda_n})_{3D} + P_{bc}. \tag{3.30}$$

Portanto com $P_{b\lambda_n novo}$ e $P_{b_1\lambda_n novo}$ pode-se utilizar o mesmo algoritmo da seção (3.2) e obter finalmente os parâmetros de T_{bb_1}.

3.4 Algoritmo de calibração - sensor externo

Para o algoritmo da seção (3.2) ser utilizado é necessário que os pontos de calibração $P_{b\lambda_n}$ e $P_{b_1\lambda_n}$, com $n = 1, \ldots, N$, sejam representados no sistema de coordenadas da câmera, E_c. A seguir são indicados os principais passos.

1. Selecionar um conjunto de pontos de entrada;
2. Representar os pontos $P_{b\lambda_n}$ e $P_{b_1\lambda_n}$ no sistema de coordenada da câmera E_c, ficando ambos $(P_{c\lambda_n})_b$ e $(P_{c\lambda_n})_{b_1}$;
3. Após representar os pontos em E_c, representa-os no plano imagem, $(P_{im_n})_b$ e $(P_{im_n})_{b_1}$;
4. Calcular $(K_{p_n})_b$ e $(K_{p_n})_{b_1}$;
5. Calcular $(P_{b\lambda_n})_{2D}$ e $(P_{b_1\lambda_n})_{2D}$ que são os pontos de calibração em duas dimesões;
6. Com $(P_{b\lambda_n})_{2D}$, $(P_{b_1\lambda_n})_{2D}$, $(z_{c\lambda_n})_b$ e $(z_{c\lambda_n})_{b_1}$ são calculados $P_{b\lambda_n novo}$ e $P_{b_1\lambda_n novo}$
7. Selecionar uma solução inicial x_0 e um critério de término, que neste caso é o número de iterações. Calcular $J(0)$ e $f(0)$. Se o erro relativo entre cada iteração for menor até chegar a um nível em que a diferença entre os erros atual e anterior forem muito pequenas ir para o passo 10, caso contrário fazer $k = 0$ e ir para o próximo passo;

8. Calular a matriz do Jacobiano $J(k)$ e a função principal $f(k)$;

9. Calcular o vetor de ajustes $x_{k+1} = x_k - \alpha[J(k)^T J(k)]^{-1} J(k)^T f(k)$. Caso tenha sido atingido $x_k = x_{bb_1}$, pular para o passo 10, caso contrário faz-se $k = k + 1$ e voltar ao passo 8;

10. Os três parâmetros de posição e os três parâmetros de orientação ($P_{bb_{1_x}}, P_{bb_{1_y}}, P_{bb_{1_z}}$, φ, ϑ, ψ) são finalmente determinados.

Os subindices b e b_1 referem-se a posição do robô nos sistemas de coordenadas E_b e E_{b_1}.

3.4.1 Simulação

Para as simulações utilizou-se novamente o modelo cinemático do manipulador robótico Zebra Zero, segundo a convernção standard de Denavit-Hatenberg, conforme apresentado na Tabela 3.1 e a grade de calibração mostrada na Tabela 3.2. É apresentado também os resultados de simulações obtidos com a implementação do segundo método proposto neste capítulo.

Da mesma forma as simulações foram realizadas em *Matlab* (The Mathworks, Inc.) utilizando o toolbox de robótica, onde tentou-se recriar nas simulações o problema em estudo.

3.4.1.1 Modelo matemático

Para as simulações realizadas com este método, adotou-se como modelo matemático, a transformação de coordenadas entre o espaço de trabalho e a imagem da câmera obtida em (3.27) dada por

$$P_{im_n} = K_{p_n}(P_{b\lambda_n})_{2D}, \tag{3.31}$$

com

$$K_{p_n} = \frac{f}{z_{c\lambda_n}} \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} R_{cb}, \tag{3.32}$$

e

$$z_{c\lambda_n} = f + z_n, \tag{3.33}$$

onde P_{im} [pixels] é a posição do efetuador no sistema de coordenadas da imagem da câmera; $(P_{b\lambda_n})_{2D}$ [mm] é a posição do efetuador no espaço de trabalho do efetuador; R_{cb} [rad] é a matriz de rotação entre a câmera e a base do robô; f [mm] é a distância focal da câmera; z_n [mm] é a profundidade relativa medida entre o plano da imagem e o espaço de trabalho do robô; α_1, α_2 [pixels/mm] são fatores de escala da câmera e $n = 1, \ldots, N$ é o número de pontos destinados a calibração.

3.4.1.2 Parâmetros de Simulação

Os parâmetros utilizados nas simulações com sensor visual são apresentados na Tabela 3.10:

Variável	Valor	Unidade
l_1	$279,4$	mm
l_2	$228,6$	mm
f	6	mm
α_1	100	mm
α_2	100	mm

TABELA 3.10: Parâmetros de simulação utilizados nas simulações com sensor visual.

3.4.2 Resultados simulação

Nesta seção, apresenta-se os resultados de simulação obtidos com o segundo método proposto neste capítulo. Os resultados obtidos com este método são similares aos re-

sultados apresentados no primeiro método, os quais utilizam sensores internos para estimação do erro de posicionamento. As situações consideradas para obter os parâmetros de T_{bb_1} foram: ideal e ruído na posição dos pontos-imagem.

A Figura 3.16 apresenta o erro de posicionamento estimado com os pontos da grade de calibração, vide Tabela 3.2, considerando uma situação ideal, ou seja, sem a presença de ruído, porém pode-se observar que os parâmetros desse erro são os mesmos obtidos com o primeiro método.

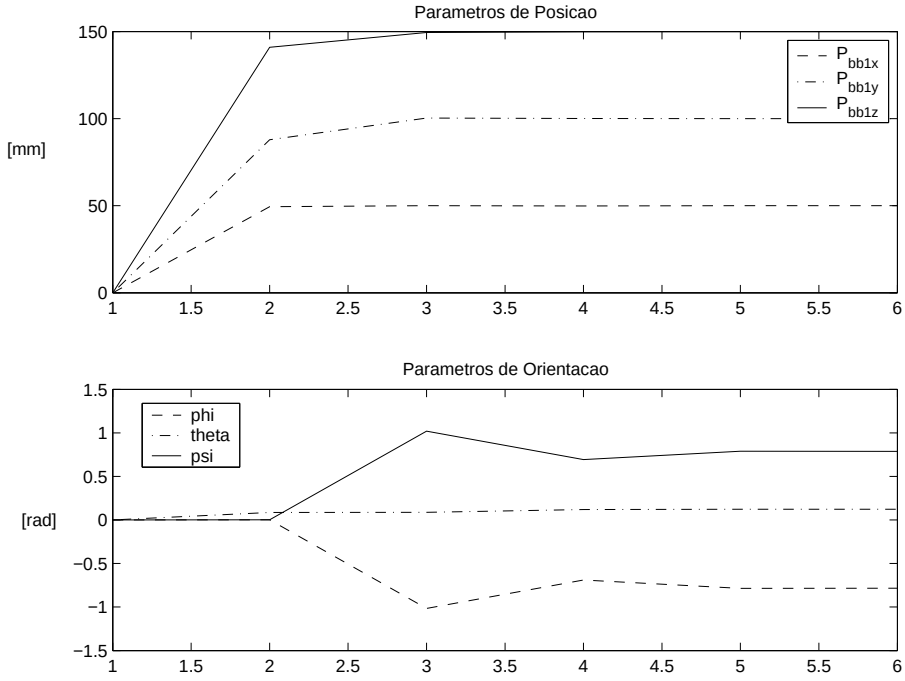

FIGURA 3.16: Resultado de simulação: Erro de posicionamento estimado com sensor visual, considerando uma situação ideal.

Neste método com sensor visual, para emular uma situação real é adicionado ruído, o qual é randômico e gerado pela função *rand* do *matlab* que possui uma distribuição uniforme em um intervalo de 0 a 1.

A grade de calibração apresentada na Tabela 3.2 é utilizada também neste método, sendo válido dizer que esta tem que ser perfeitamente conhecida para evitar pontos fora da área de trabalho do robô que levariam a resultados divergentes. Da mesma forma que no primeiro método são realizadas simulações com diferentes conjuntos de pontos de calibração da grade.

As Figuras 3.17 a 3.20 apresentam os resultados de simulação para os diferentes

conjuntos de pontos de calibração: 18, 10, 5 e 3, respectivamente. As limitações encontradas no primeiro método são válidas também para este que utiliza sensor visual. É válido lembrar também que este método e o já apresentado anteriormente são complementares, sendo destacado a utilidade deste em ambientes remotos ou de difícil acesso. Assim como no primeiro método os resultados de simulações pertubados por ruído não exerceram grandes influências nos parâmetros estimados.

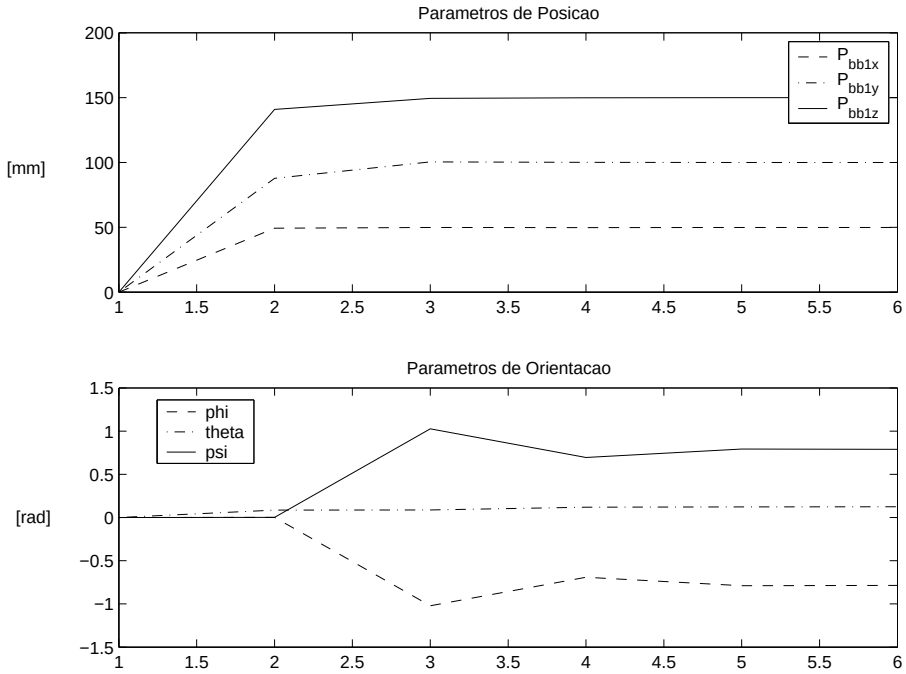

FIGURA 3.17: Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensor visual, para $N = 18$ pontos da grade de calibração.

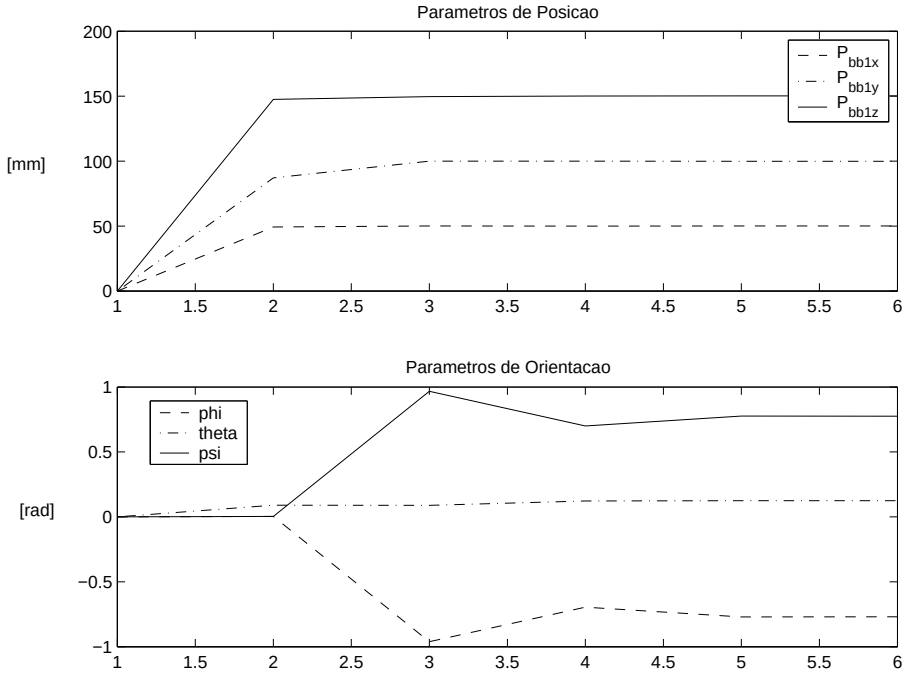

FIGURA 3.18: Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensor visual, para $N = 10$ pontos da grade de calibração.

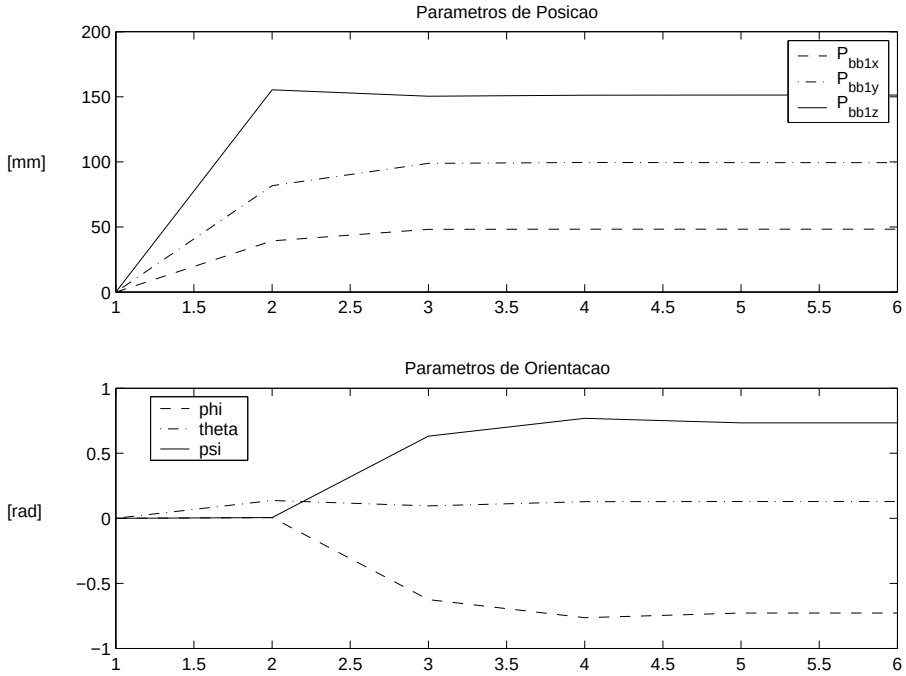

FIGURA 3.19: Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensor visual, para $N = 5$ pontos da grade de calibração.

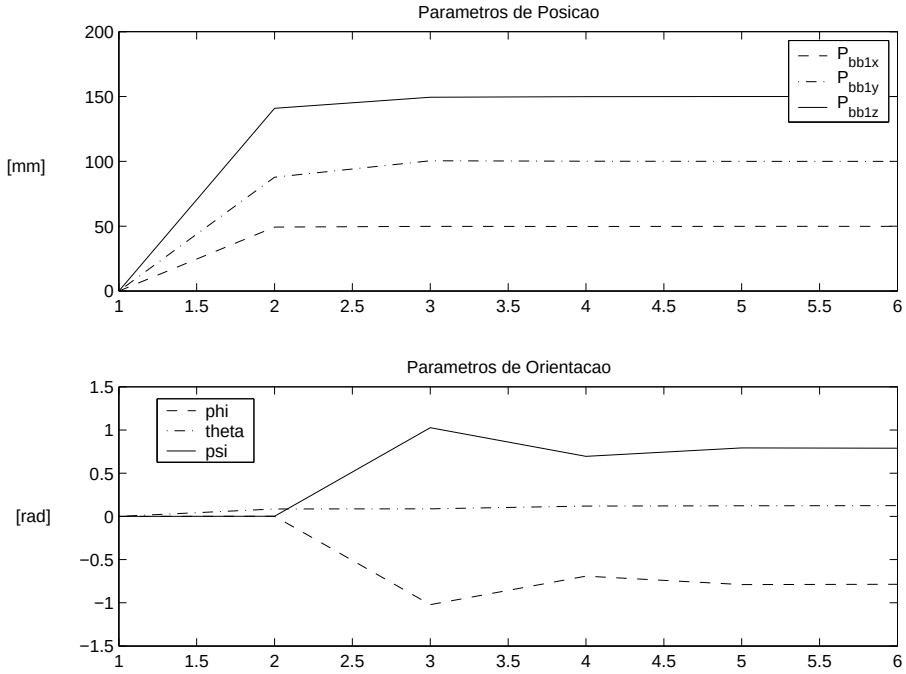

FIGURA 3.20: Resultado de simulação: Parâmetros do erro de posicionamento estimados utilizando sensor visual, para $N = 3$ pontos da grade de calibração.

As Tabelas 3.12 a 3.15 mostram o erro de posicionamento obtido com cada conjunto de pontos de calibração apresentados nas simulações, em comparação com a situação ideal. Na Tabela 3.11 pode ser visto o erro de posição entre o valor ideal do erro de posicionamento e o valor estimado para cada conjunto de pontos de calibração, assim como o erro relativo da posição.

Erro de Posição	Erro Relativo da Posição	N pontos de calibraçao
0.1118	0.0598%	$N = 18$
0.2922	0.1562%	$N = 10$
2.2555	1.2056%	$N = 5$
1.3341	0.7131%	$N = 3$

Tabela 3.11: Erro de posiçao e Erro relativo da posição - Sensor externo

Parâmetro	Valor Real	Valor Estimado: $N = 18$	Erro Absoluto
$P_{bb_{1_x}}$	50	49.9081	0.0919
$P_{bb_{1_y}}$	100	100.0346	0.0346
$P_{bb_{1_z}}$	150	150.0535	0.0535
φ	-0.7835	-0.7862	0.0027
ϑ	0.1233	0.1237	0.0004
ψ	0.7873	0.7899	0.0026

Tabela 3.12: Erro de posicionamento estimado utilizando o segundo método - sensores externos para $N = 18$ pontos da grade de calibração.

Parâmetro	Valor Real	Valor Estimado: $N = 10$	Erro Absoluto
$P_{bb_{1_x}}$	50	50.0531	0.0537
$P_{bb_{1_y}}$	100	99.9395	0.0605
$P_{bb_{1_z}}$	150	150.2809	0.2809
φ	-0.7835	-0.7693	0.0142
ϑ	0.1233	0.1245	0.0012
ψ	0.7873	0.7747	0.0126

TABELA 3.13: Erro de posicionamento estimado utilizando o segundo método - sensores externos para $N = 10$ pontos da grade de calibração.

Parâmetro	Valor Real	Valor Estimado: $N = 5$	Erro Absoluto
$P_{bb_{1_x}}$	50	48.3242	1.6758
$P_{bb_{1_y}}$	100	99.4497	0.5503
$P_{bb_{1_z}}$	150	151.4058	1.4058
φ	-0.7835	-0.7284	0.0551
ϑ	0.1233	0.1289	0.0056
ψ	0.7873	0.7332	0.0541

TABELA 3.14: Erro de posicionamento estimado utilizando o segundo método - sensores externos para $N = 5$ pontos da grade de calibração.

Parâmetro	Valor Real	Valor Estimado: $N = 3$	Erro Absoluto
$P_{bb_{1_x}}$	50	49.2029	0.7971
$P_{bb_{1_y}}$	100	98.9637	1.0363
$P_{bb_{1_z}}$	150	150.2655	0.2655
φ	-0.7835	-0.7726	0.0109
ϑ	0.1233	0.1247	0.0014
ψ	0.7873	0.7805	0.0068

TABELA 3.15: Erro de posicionamento estimado utilizando o segundo método - sensores externos para $N = 3$ pontos da grade de calibração.

3.5 Replanejamento da trajetória

Replanejar a trajetória é necessária quando existe incerteza na localização absoluta do manipulador em relação a localização original assumida. Como o manipulador se encontra deslocado em relação ao sistema de coordenada inercial e para que este execute a mesma trajetória definida com ele em E_b é necessário fazer o replanejamento da trajetória, o qual é realizado utilizando o erro de posicionamento estimado na etapa de calibração, e com a trajetória ideal planejada em E_b.

3.5.1 Algoritmo de cinemática inversa

Com o erro de posicionamento, T_{bb_1}, obtido na etapa de calibração, a nova trajetória desejada é estabelecida. Os procedimentos para obtenção desta nova trajetória são mostrados a seguir. A Figura 3.21 mostra o sistema de coordenadas do robô em E_b e E_{b_1}, para um ponto λ de calibração. Os cálculos envolvendo n pontos segue o mesmo procedimento.

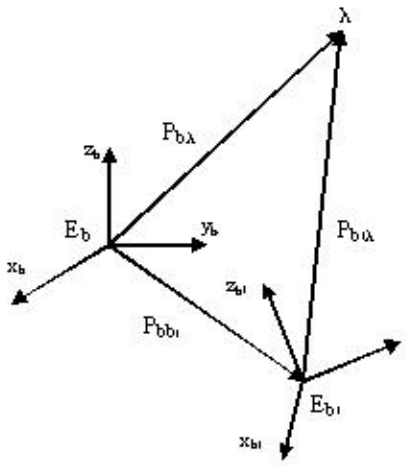

FIGURA 3.21: Um ponto λ no espaço nos sistemas de coordenada E_b e E_{b_1}

Com o manipulador na localização original, E_b, a trajetória ideal desejada é dada

$$x_{bd} = k(\theta_{bd}), \tag{3.34}$$

de acordo com a cinemática diferencial, (3.34) pode ser escrita como

$$\dot{x}_{bd} = J(\theta_{bd})\dot{\theta}_{bd}. \tag{3.35}$$

Agora considerando o manipulador deslocado em relação a localização original, ou seja, encontrando-se em E_{b_1}, a nova trajetória desejada é dada por:

$$x_{b_1 d} = T_{bb_1}^{-1} x_{bd}, \tag{3.36}$$

e

$$\theta_{b_1 d} = k^{-1}(x_{b_1 d}). \tag{3.37}$$

Derivando (3.36) fica

$$\dot{x}_{b_1 d} = T_{bb_1}^{-1} \dot{x}_{bd}. \tag{3.38}$$

Substituíndo (3.35) em (3.38)

$$\dot{x}_{b_1 d} = T_{bb_1}^{-1} J(\theta_{bd})\dot{\theta}_{bd}, \tag{3.39}$$

$$J(\theta_{b_1 d})\dot{\theta}_{b_1 d} = T_{bb_1}^{-1} J(\theta_{bd})\dot{\theta}_{bd}, \tag{3.40}$$

finalmente o esquema em malha aberta fica

$$\dot{\theta}_{b_1 d} = J^{-1}(\theta_{b_1 d}) T_{bb_1}^{-1} J(\theta_{bd})\dot{\theta}_{bd}. \tag{3.41}$$

O diagrama em bloco corresponde ao algoritmo para controle cinemático em malha fechada é mostrada na Figura 3.22, onde $k(.)$ indica a função cinemática direta.

Dessa forma conforme o diagrama em bloco da Figura 3.22 a nova trajetória desejada a ser seguida pelo manipulador robótico que se encontra deslocado em relação ao sistema de coordenadas original é obtida. Sendo assim o manipulador robótico realiza a mesma tarefa que faria quando estava localizado em E_b. Em seguida é mostrado um exemplo para validar o enfoque apresentado, considerando novamente o manipulador Ze-

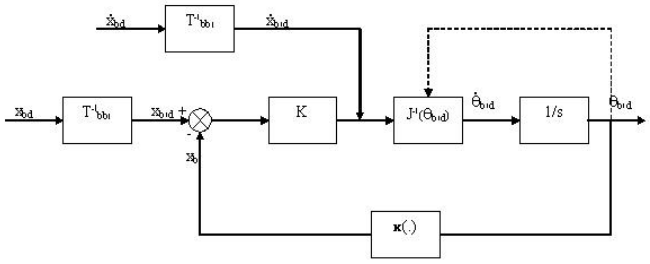

FIGURA 3.22: Diagrama em bloco do algoritmo para cinemática de controle cinemático com jacobiano inverso

bra Zero. Inicialmente considerou-se que a base do manipulador coincide com o sistema de coordenadas inercial. Para este foi planejado uma trajetória circular no plano $x-z$. Supondo que a base do robô tenha sido deslocada de $P_{bb_1} = [50\ 100\ 150]^T$, cujas medidas são expressas em mm. E considerando um ponto de calibração λ que em E_b é dado por $P_{b\lambda} = [228.6\ 0\ 379.4]^T$ que corresponde ao ponto $P_{b_1\lambda} = [157.24\ -79.626\ 251.91]^T$ em E_{b_1}. Com o erro de posicionamento obtido na etapa de calibração a tarefa é replanejada conforme o esquema da Figura 3.22. A Figura 3.23 apresenta a trajetória do efetuador com a base do manipulador na posição original e na nova posição.

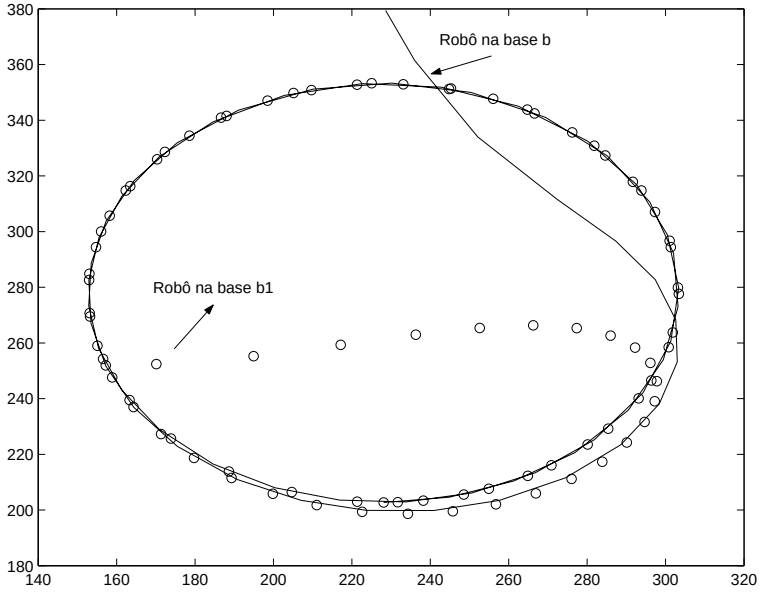

FIGURA 3.23: Replanejamento de Trajetória

Capítulo 4

Calibração e Replanejamento no Espaço de Velocidades

Este capítulo aborda um método de calibração e replanejamento considerando o problema no espaço de velocidades cartesianas. Este método assim como os abordados nos capítulos anteriores consistêm em caracterizar a incerteza na localização absoluta do manipulador, porém, a idéia proposta nesta abordagem é considerar implicitamente a transformação homogênea T_{bb_1}, estimada nos métodos apresentados no Capítulo 3, e trabalhar com as velocidades no espaço cartesiano que sintetizariam esta transformação. Sendo dessa forma o conhecimento de T_{bb_1} equivalente ao conhecimento das velocidades linear e angular.

4.1 Planejamento do caminho

Uma abordagem diferente para o planejamento de trajetória é usar o modelo cinemático diferencial do manipulador. Além de resolver o problema de planejamento é também possível a resolução de problemas de redundâncias.

Dada a cinemática direta do manipulador por:

$$x(t) = k(\theta(t)), \tag{4.1}$$

onde $x(t) \in I\!R^m$ é a coordenada no espaço operacional e $\theta(t) \in I\!R^n$ é a coordenada da junta, a derivada em relação ao tempo é

$$\dot{x}(t) = J(\theta(t))\dot{\theta}(t). \tag{4.2}$$

A matriz $J(\theta(t))$: $I\!R^n \mapsto I\!R^m$ é chamada de matriz jacobiana, é uma função não linear dos ângulos das juntas. Neste contexto, o planejamento do caminho é resumido a achar soluções de (4.2), sujeita as restrições de igualdade (tarefa desejada). Sendo o objetivo principal encontrar um caminho ótimo que satisfaça um conjunto de restrições de igualdade do espaço da junta e da tarefa.

É importante dizer que as restrições de desigualdade do espaço da junta e da tarefa, têm sido pouco abordada na literatura de planejamento do caminho via técnicas de otimização, sendo exceção a utilização de campos de nível potenciais. Um método que engloba limites no espaço da junta em um procedimento compacto de programação quadrática é mostrado em (Chen, Cheng & Sun 1992), entretanto este método não engloba as restrições de desigualdade no espaço das tarefas.

O problema do planejamento do caminho é colocado como um problema de controle não-linear de tempo finito, que é convertido em um problema não-linear de encontrar raíz com um grande espaço de busca e um espaço comparativamente menor da restrição. Um algoritmo iterativo de Newton-Raphson pode então ser aplicado que garanta a convergência sob suposições razoavelmente suaves. Restrições de desigualdade, no espaço da junta e no espaço da tarefa, podem ser incorporadas.

Com a ênfase colocada de encontrar um caminho prático e melhor que um caminho ótimo, o problema de convergência na abordagem de otimização global é evitado, constituindo um problema de valor-inicial ao em vês do problema limite de dois pontos mencionado anteriormente associado com a abordagem de otimização global, onde as exigências computacionais são moderadas.

Esta aproximação é aplicada a ambos braços redundantes e planejamento não-holomônico do caminho nas manobras do veículo com resultados muito promissores (Divelbiss & Wen 1994).

Baseado em exemplos, pode-se indicar as seguintes características desejadas desse algoritmo.

- O planejamento pode ser usado para encontrar uma sequência de juntas viáveis de uma configuração inicial fixa para um espaço cartesiano especificado ou espaço da junta objetivo, problema do planejamento do caminho;

- A natureza global do planejamento evita o problema de singularidade no jacobiano inerente em métodos locais. Enquanto a controlabilidade local numa dada configuração é perdida na singularidade, no algoritmo proposto a controlabilidade é considerada ao longo da trajetória que é uma condição mais fácil de ser atendida;

- A formulação proposta acomoda restrições de igualdade no espaço da junta e espaço da tarefa. Em conseqüência, pode gerar o movimento no espaço da junta para um caminho cartesiano cíclico especificado.

4.2 Abordagem da solução iterativa

Nesta abordagem é considerado o problema de encontrar um caminho contínuo no espaço das juntas que satisfaça uma trajetória especificada no espaço da tarefa.

Considerando $\dot{\theta}(t)$ como a variável de controle,

$$u(t) = \dot{\theta}(t), \tag{4.3}$$

o problema pode ser colocado da seguinte forma

Problema: Dado o sistema descrito pela equação (4.2) encontrar $\bar{u} = \{u(t), t \in [0, T]\}$ tal que $x(T) = x_d$.

Onde $T < \infty$ é a janela de tempo em que se quer resolver o problema.

A equação (4.2) pode ser escrita como uma equação não linear algébrica

$$\bar{x} = F(\bar{u}), \tag{4.4}$$

a forma analítica de $F(.)$ é em geral difícil de encontrar, e não será necessário demonstrar.

O problema é iniciado fazendo o erro igual a zero:

$$e = F(\bar{u}) - x_d = 0, \tag{4.5}$$

onde e representa o erro entre a posição atual e a posição desejada. A posição desejada são os pontos de calibração na base b_1.

A solução é encontrada considerando a diferenciação de (4.5) com respeito a variável de iteração τ:

$$\frac{de}{d\tau} = \nabla_{\bar{u}} F(\bar{u}) \frac{d\bar{u}}{d\tau}, \tag{4.6}$$

onde o mapeamento $\nabla_{\bar{u}} F(\bar{u})$ é a derivada de Fréchet de F com respeito a $\bar{u}$. Desta forma, supondo que $\nabla_{\bar{u}} F$ seja sobrejetivo, uma escolha adequada para lei de atualização de $\bar{u}(\tau)$ é dada por:

$$\frac{d\bar{u}(\tau)}{d\tau} = -\alpha \left[\nabla_{\bar{u}} F(\bar{u})\right]^{\dagger} e(\tau), \tag{4.7}$$

onde $\alpha > 0$ e $[\,\cdot\,]^{\dagger}$ denota a pseudo-inversa de Moore-Penrose. Tal lei é essencialmente a versão contínua do método de Newton. A equação diferencial (4.7) define um problema de condição inicial (IVP) para um dado $u(0)$, desta forma a solução $\bar{u}(\tau)$ pode ser obtida resolvendo (4.7) através de algum programa de integração numérica. No entanto, a integração numérica de (4.7) não leva em consideração as fortes propriedades de contração local inerentes a métodos iterativos tipo Newton. Este tipo de método numérico também é conhecido como *método de Newton Global*.

Uma condição suficiente para a convergência do algoritmo é que $\nabla_{\bar{u}} F(\bar{u})$ seja sobrejetivo para todo τ. Sob esta condição tem-se que, substituindo (4.7) em (4.6):

$$\frac{de}{d\tau} = -\alpha e, \tag{4.8}$$

o que claramente implica na convergência exponencial de $e(\tau)$:

$$\|e(\tau)\| \leq K \|e(0)\| \exp^{-\alpha\tau},$$

para algum $K > 0$ (por exemplo, se $\|\ \|$ é a norma-2 então $K = \sqrt{n}$).

4.3 Calibração e replanejamento no espaço de velocidades

Nos capítulos anteriores a base do replanejamento da trajetória foi a estimação da transformação homogênea T_{bb_1} através de medidas de uma grade de calibração de dimensões conhecidas.

A idéia aqui é ao invés de considerar explicitamente a transformação homogênea T_{bb_1}, é trabalhar com as velocidade no espaço cartesiano que sintetizariam esta transformação. Desta forma o problema pode ser colocado da seguinte forma: o conhecimento de T_{bb_1} é equivalente ao conhecimento da velocidade linear $v_c(t)$ e angular $\omega_c(t)$ que resolvem a seguinte equação diferencial:

$$\dot{p} = v; \qquad \text{com } p(0) = 0; \quad p(T) = p_{bb_1} \tag{4.9}$$

$$\dot{R} = \omega \times R; \qquad \text{com } R(0) = I; \quad R(T) = R_{bb_1} \tag{4.10}$$

No caso que a orientação seja parametrizada por quaternions e desta forma q_{bb_1} seja o quaternion associado com a matriz de orientação R_{bb_1}, tem-se

$$\dot{p} = v; \quad \text{com } p(0) = 0; \quad p(T) = p_{bb_1} \tag{4.11}$$

$$\dot{q} = J_r(q)\omega; \quad \text{com } q(0) = I; \quad q(T) = q_{bb_1} \tag{4.12}$$

onde o jacobiano da representação $J_r(q)$ é dado por

$$J_r(q) = \frac{1}{2} \begin{bmatrix} -q_v^T \\ \\ q_0 I - \hat{q}_v \end{bmatrix},$$

com q_0 e q_v sendo a parte escalar e vetorial do quaternion, respectivamente.

Desta forma considerando que os pontos de calibração $\lambda_1, \cdots, \lambda_m$ sejam determinados por $x_1, \cdots, x_m$ no sistema de coordenadas E_b e por $x'_1, \cdots, x'_m$ em E_{b_1}; onde $x^T = [p^T, q^T]$.

Então a idéia é determinar $v(t)$ e $\omega(t)$ que leve simultaneamente todos os x_i para

seu respectivo x_i', e desta forma estimar implicitamente a transformação homogênea T_{bb_1}.

Para isto empilhando os x_i:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}.$$

Tem-se que

$$\dot{X} = \bar{J} \begin{bmatrix} v \\ \omega \end{bmatrix} = \bar{J}u,$$

onde

$$\bar{J} = \begin{bmatrix} I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & J_R(q_1) \\ \vdots & \vdots \\ I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & J_R(q_m) \end{bmatrix}.$$

A derivada $\nabla_{\bar{u}}F$ é obtida do sistema linearizado ao redor de uma trajetória (x, u):

$$\partial\dot{x} = A(t)\partial x + B(t)\partial u,$$

onde $B(t) = \bar{J}(x(t))$ e $A(t) = \left[\frac{\partial\bar{J}}{\partial p_1}u(t)\ \frac{\partial\bar{J}}{\partial q_1}u(t)\cdots\frac{\partial\bar{J}}{\partial p_m}u(t)\ \frac{\partial\bar{J}}{\partial q_m}u(t)\right]$. No nosso caso particular tem-se que $\frac{\partial\bar{J}}{\partial p_i} = 0_{3\times3}$.

A versão discreta do sistema linearizado é dada por:

$$\delta x(k+1) = \Phi(k)\delta x(k) + \Gamma(k)\delta u(k); \qquad \delta x(0) = 0; \tag{4.13}$$

onde $\Phi(k) = e^{A(kh)h}$ e $\Gamma(k) = \int_{kh}^{(k+1)h} e^{A(kh)s}dsB(kh)$.

Resolvendo-se (4.13) para um horizonte M, tem-se:

$$\delta x(M) = D\ \delta u, \tag{4.14}$$

onde

$$D = [\prod_{j=1}^{M-1} \Phi(j)\Gamma(0), \prod_{j=2}^{M-1} \Phi(j)\Gamma(1), \cdots, \Phi(M-1)\Gamma(M-2), \Gamma(M-1)]. \tag{4.15}$$

Portanto, $\nabla_{\bar{u}} F = D$ dado que D relaciona mudanças infinitesimais em $\bar{u}$ com mudanças infinitesimais em $x(M)$.

4.3.1 Método iterativo

O algoritmo utilizado no método iterativo para se estimar $u = [v_c\ w_c]^T$ que leva os pontos de calibração da grade no sistema de coordenadas E_b para o sistema de coordenadas E_{b_1}, estimando implicitamente a transformação homogênea T_{bb_1} é mostrado a seguir:

Algoritmo Iterativo

1. Dados os vetores X e $X^{'}$ que representam os pontos de calibração $\lambda_1, \cdots, \lambda_m$ da grade nos sistemas de coordenadas E_b e E_{b_1} respectivamente;
2. Dado $\bar{u}$ e um critério de término ε;
3. Resolver $\dot{z} = \bar{J}u$ para $\bar{u}$ e $z(0) = X$;
4. Calcular $e = z(T) - X^{'}$;
5. Se $e < \varepsilon$ ir para o passo 9;
6. Calcular $\nabla_{\bar{u}} F$ utilizando a Equação (4.15);
7. Calcular $\bar{u} = \bar{u} - \alpha(\nabla_{\bar{u}} F)^{\dagger} e$;
8. Voltar ao passo 3;
9. $u_c = \bar{u}$.

As velocidades linear v_c e angular ω_c estimadas pelo algoritmo iterativo anterior podem ser utilizadas diretamente para corregir o erro na localização do robô.

O esquema completo de replanejamento é ilustrado na Figura 4.1.

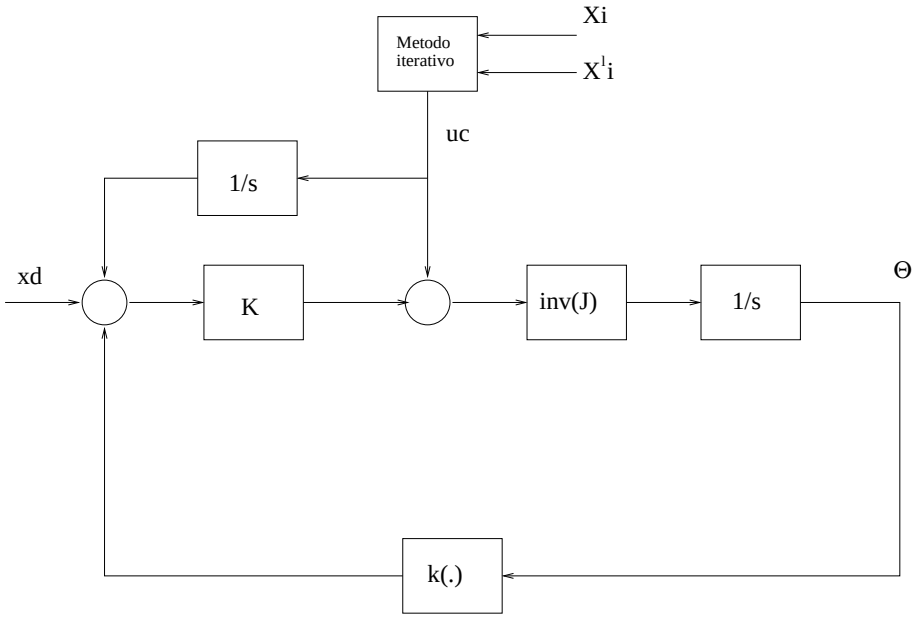

FIGURA 4.1: Esquema de replanejamento.

Para validar este enfoque considera-se novamente o manipulador Zebra Zero. Considera-se inicialmente que a base do robô coincide com o sistema de coordenadas inercial. Neste caso foi planejada uma trajetória circular no plano $x - z$. Agora suponha que a base do robô foi deslocada na direção z uns $100mm$. Considere que tem-se somente um ponto de calibração dado por $p_1 = [228.6\ 0\ 279.4]^T$ que corresponde ao ponto $p_1' = [228.6\ 0\ 179.4]^T$, vide Figura 4.2.

O sinal de controle de calibração calculado pelo algoritmo iterativo é apresentado na Figura 4.3 onde foi considerado $\alpha = 0.75$ e $u(0) = 0$.

Mediante a utilização do sinal de controle de calibração a tarefa é replanejada segundo o esquema da Figura 4.1. A Figura 4.4 apresenta a trajetória do efetuador quando a base do manipulador foi deslocada para a nova posição.

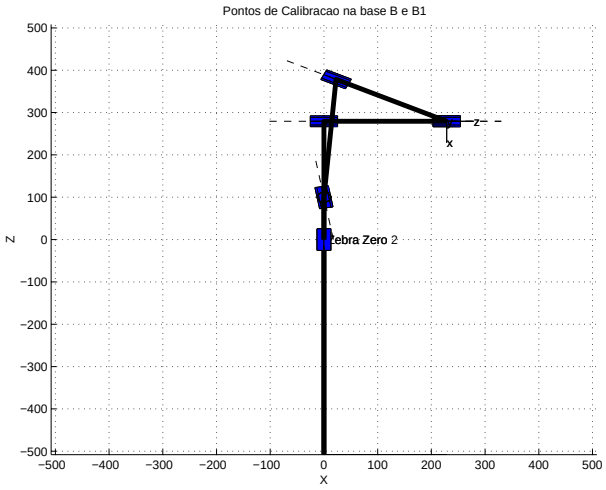

FIGURA 4.2: Ponto de calibração.

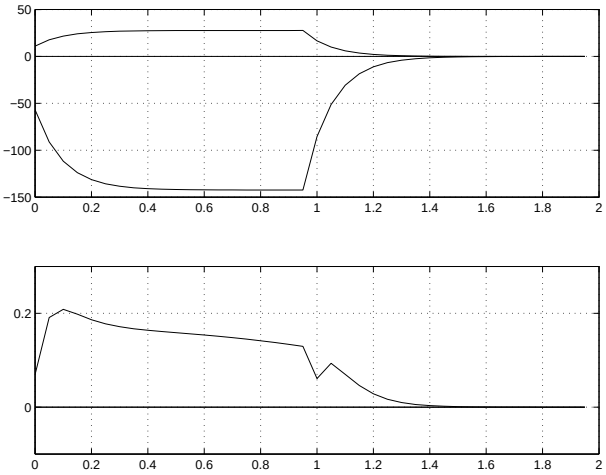

FIGURA 4.3: Sinal de controle de calibração.

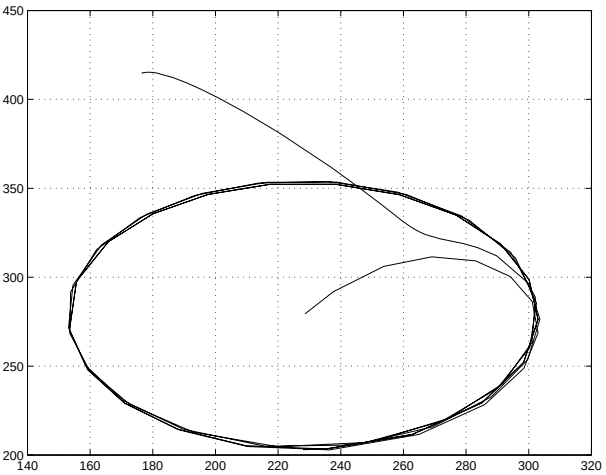

FIGURA 4.4: Replanejamento de Trajetória.

Capítulo 5

Conclusões

O presente trabalho propôs métodos de calibração baseados em um algoritmo de mínimos quadrados que utilizam informações da configuração do manipulador numa grade de calibração através de sensores internos e externos para obtenção do erro de posicionamento. A estimação deste erro de posicionamento ou incerteza pode ser realizada no espaço operacional ou no espaço de velocidades. É importante ressaltar que este trabalho foi dividido em três etapas:

- Planejamento de Trajetória;
- Calibração;
- Replanejamento da Trajetória.

A primeira etapa abordou de forma genérica a programação off-line. A necessidade de se abordar o planejamento de trajetória é gerar entradas de referência para o movimento do sistema de controle que assegure que o manipulador execute a trajetória planejada.

A fim de solucionar o problema da incerteza da localização da célular robótica, são apresentados na segunda etapa deste trabalho os métodos para estimação do erro de posicionamento entre a localização assumida como original e a localização atual que se encontra o robô.

Foi apresentado também como terceiro método de calibração o problema no espaço de velocidades cartesianas, a abordagem consistiu em caracterizar a incerteza na loca-

lização absoluta do manipulador em função das velocidades cartesianas. A idéia deste método foi considerar implicitamente a transformação homogênea T_{bb_1}, e dessa forma trabalhar com as velocidades linear e angular as quais sintetizam esta transformação.

Contudo, os métodos para replanejamento de trajetória, tanto quanto os no espaço operacional e no espaço de velocidades cartesianas, as principais contribuições deste trabalho.

Simulações em um sistema robótico real comprovaram a eficácia dos métodos apresentados. Os resultados das simulações obtidas ilustram o desempenho do algoritmo proposto. Assim como mostram os parâmetros do erro de posicionamento estimados com ruído.

5.1 Propostas para trabalhos futuros

Com o objetivo de incentivar a continuação da pesquisa apresentada neste trabalho, seguem algumas propostas para serem desenvolvidas em trabalhos futuros:

- Utilizar como algoritmo de estimação do erro de posicionamento o algoritmo de Levenberg-Marquardt, pois como a matriz $J^T J$, que precisa ser invertida a cada iteração, pode apresentar problemas de mal-condicionamento, podendo assim levar a resultados divergentes, contudo com o método de Levenberg-Marquardt, modifica-se a matriz a inverter de forma a evitar o mal-condicionamento;

- Considerar nos dois primeiros métodos a orientação, pois dessa forma, apenas um ponto de calibração é necessário para realizar a calibração, ao invés de três, como foi apresentado neste trabalho;

- Considerar como parametrização da orientação nos dois primeiros métodos quaternions, pois estes estão livres de singularidades.

Apêndice A

Otimização

Quando se fala em otimizar, pensa-se logo em minimizar ou maximizar algo. Em outras palavras otimização é o estudo de situações especiais, o que são, quanto valem e como atingí-las, ou seja, é a busca de maximizadores e minimizadores. Neste enfoque minimização e maximização são problemas relacionados, uma vez que ao se encontrar o mínimo de uma função (f) significa o mesmo que encontrar o máximo de ($-f$).

Então o problema de otimização como é comumente conhecido, se refere a sistemas ou funções que podem não ser completamente descritos matematicamente, o que conduz à necessidade de se estimar os parâmetros que melhor descrevem determinado sistema ou função bem como suas condições ótimas de operação, ou seja, que representem a operação do sistema a um custo mínimo.

Portanto sistemas impossíveis aparecem muitas vezes em aplicações, embora nem sempre com uma matriz de coeficientes tão grande. Quando se precisa de uma solução que não existe, o melhor que podemos fazer é encontrar um $\mathbf{x}$, vetor de parâmetros do sistema, que faça com que $A\mathbf{x}$ fique o mais próximo possível de $\mathbf{b}$ (Belegundu & Chandrupatla 1999). Em outras palavras o problema de otimização pode ser formulado como um problema de minimização (ou maximização), o processo de otimização consiste em dada uma função $f(x)$ representando um determinado sistema, sendo x o vetor de parâmetros desse sistema, encontrar o valor mínimo (ou máximo) de f, ou melhor ainda o valor ótimo de f.

O problema de minimização consiste então em minimizar a função f de forma a

obter seu valor mínimo.

Os métodos de otimização tratados neste apêndice necessitam de uma função suave, ou seja onde haja ausência de mudanças bruscas, e contínua, pois são baseados em cálculo de derivadas.

A necessidade de se abordar determinados métodos de otimização se deve ao fato de termos que estimar os parâmetros de por exemplo de uma transformação linear ou até mesmo obter as componentes para calibração. Portanto estimação é um processo de ajuste de um modelo matemático para dados experimentais a fim de determinar parâmetros desconhecidos no modelo. O processo de estimação é frequentemente não-linear porque os dados observados não variam em proporção direta aos parâmetros em questão.

Este apêndice descreve genericamente o problema de minimização, assim como apresenta alguns métodos de otimização tais como: método baseado em gradiente e métodos globalmente convergentes.

Sabe-se que a maioria dos problemas de engenharia envolvem minimizações com restrições, isto é, minimizar uma função sujeita a restrições. Contudo, o uso de restrições torna mais difícil o processo de otimização, se comparado com métodos sem restrições.

O problema de minimização consiste então em minimizar a função f de forma a obter seu valor mínimo, ou seja:

$$minF = f(x_{(min)}), \tag{A.1}$$

sendo $f(x)$ uma função que descreve um determinado sistema em termos de seus parâmetros (variáveis), onde estes parâmetros são representados por um vetor coluna de n variáveis como

$$x = [x_1, x_2, ..., x_N]^T, \tag{A.2}$$

e f, a função a minimizar ($F = f(x)$), é chamada de função objetivo.

O processo de minimização consiste em um processo iterativo, onde os valores dos parâmetros são ajustados até uma condição pré-determinada ter sido atingida. Sendo o valor ótimo final indicado por $F_{min} = f(x_{min})$. Então os ajustes dos parâmetros x são representados pelo vetor

$$\Delta x = [\Delta x_1, \Delta x_2, ..., \Delta x_N]^T. \tag{A.3}$$

Sendo o processo iniciado a partir de uma solução inicial x_0 correspondendo ao valor inicial de $F_0 = f(x_0)$ da função objetivo. O valor x_0 é escolhido de forma que represente uma boa solução inicial, ou seja, uma escolha inadequada pode alterar substancialmente o tempo para a convergência do método iterativo, ou mesmo levar à divergência.

Contudo o mínimo obtido, x_{min}, pode ser um mínimo local ou global, sendo que no primeiro caso, representa o menor valor de f em um intervalo ou espaço de busca finito. Já no segundo caso, representa o mínimo de f para qualquer variação de seus parâmetros x_i. A busca do mínimo global é um problema de difícil solução, dado que a função f pode ter vários mínimos locais, sendo que um destes corresponde ao mínimo global (pode haver mais de um mínimo global, não necessita ser único), a menos que todos os mínimos locais de f sejam encontrados, o que pode ser usualmente impossível em muitos casos.

Alguns métodos de minimização requerem o uso do gradiente da função objetivo, obtido na forma das derivadas parciais de f com respeito aos parâmetros x_i. Portanto o uso de derivadas pode auxiliar na solução do problema de otimização; contudo, em certos casos seu uso pode não ser possível, devido à dificuldade de serem computadas as derivadas (Gill, Murray & Wright 1986) (Jennings & Mckeown 1993).

Portanto as derivadas de primeira ordem formam o vetor do gradiente do Jacobiano, lembrando que o gradiente armazena informações sobre a direção e a quantidade de crescimento de uma função em cada ponto. Nota-se aqui que o gradiente será indicado por g, somente para efeito de simplificação. Embora a notação ∇ é mais usual, o uso de g é mais simples, desde claro que não exista a possibilidade de confusão com outras funções. Então g é dado por

$$g = \nabla f = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, ..., \frac{\partial f}{\partial x_N}\right]^T. \tag{A.4}$$

As derivadas parciais de segunda ordem, por sua vez, formam a matriz hessiana denominada por H, dada por

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 x_N} \\ \frac{\partial^2 f}{\partial x_2 x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_N x_1} & \frac{\partial^2 f}{\partial x_N x_2} & \cdots & \frac{\partial^2 f}{\partial x_N x_2} \end{bmatrix}. \tag{A.5}$$

Há dois pontos importantes a considerar a respeito da convergência do processo iterativo em um problema de minimização que são: a velocidade de convergência e a necessidade de saber se a função objetivo F convergiu para um mínimo e se este é local ou global. A convergência pode ser verificada pelo progresso a cada iteração do valor de F. Pode-se considerar que a convergência foi atingida quando F não reduz seu valor após um certo número de iterações realizadas. A velocidade de convergência é em geral associada ao número de iterações é necessária para se obter a convergência. ou mais precisamente ao número de verificações do valor de F (Belegundu & Chandrupatla 1999).

A.1 Métodos baseados em gradiente

Os métodos de otimização que utilizam gradientes são baseados na série de Taylor

$$f(x + \Delta x) = f(x) + g^T \Delta x + \frac{1}{2} \Delta x^T H \Delta x + \ldots, \tag{A.6}$$

onde os termos superiores (ou seja a partir da terceira ordem) são considerados desprezíveis. E caso os termos de segunda ordem sejam desprezíveis, a equação (A.6) fica

$$f(x + \Delta x) \approx f(x) + g^T \Delta x = F + \Delta F. \tag{A.7}$$

Os métodos que são baseados em gradientes são mais eficientes quando a função f tem derivadas contínuas C^2 (derivadas de segunda ordem), ou seja uma função é contínua num determinado intervalo quando é possível traçar o seu gráfico, sem

descontinuidades, em outras palavras, é contínua quando não apresenta saltos. Essas derivadas podem ser obtidas analiticamente, mas em certos casos, as derivadas somente podem ser obtidas por métodos numéricos (Belegundu & Chandrupatla 1999) (Dennis & Snabil 1983).

A.1.1 Método de descida máxima

Método de maior descida, ou ainda de descida íngreme, ou, se preferir Steepest-Descent. O mecanismo básico deste método é simples e intuitivo: a direção de avanço em cada iteração deve causar o máximo decréscimo possível na função f. Assim este método (Belegundu & Chandrupatla 1999) (Jennings & Mckeown 1993) (Adby & Dempster 1982) (Chong & Zak 1996) (Kelley 1999) utiliza o gradiente g para determinar uma direção adequada de movimento em direção à suposta solução final, usando a aproximação de primeira ordem da Equação (A.7). O termo de primeira ordem pode ser reescrito como

$$\Delta F = g^T \Delta x = \sum_{i=1}^{n} \frac{\partial f(x)}{\partial x_i} \Delta x_i, \tag{A.8}$$

que ainda pode ser analisada como o produto escalar de dois vetores. Assim, pode ser reescrita como

$$\Delta F = g^T \Delta x = |g||\Delta x| \cos\theta. \tag{A.9}$$

Se partirmos de um ponto qualquer do $\mathbb{R}^n$ e avançarmos na direção do gradiente naquele ponto o aumento será máximo. E o que queremos é exatamente o efeito contrário, logo a escolha é clara, devemos avançar na direção oposta ao gradiente. Pela equação anterior, para valores fixos de $|g|$ e $|\Delta x|$, ΔF varia em função de θ. A redução máxima em F ocorrerá quando $\cos\theta$ atingir seu valor mínimo, isto é, para $\theta = \pi$, logo tem-se que a variação ótima correspondente em Δx ocorrerá na direção do gradiente negativo $-g$. Portanto podemos tirar algumas conclusões (Belegundu & Chandrupatla 1999).

- Se o produto escalar se anula quando o ângulo entre os vetores é $\frac{\pi}{2}$, concluí-se que

uma direção de avanço perpendicular ao gradiente não altera o valor da função, ou seja, esta é a direção da curva de nível que passa pelo ponto;

- Se o produto escalar é positivo quando o ângulo entre os vetores é agudo, um avanço que aponta na mesma direção do gradiente causa aumento no valor de f. No caso limite de ângulo nulo entre os vetores o produto escalar assume seu valor máximo;

- Mudando o que deve ser mudado no parágrafo acima concluímos que a direção de máxima descida é $-g$.

E resumindo (Belegundu & Chandrupatla 1999), sendo θ o ângulo entre o gradiente g e a direção de teste escolhida Δx temos

$$\begin{aligned} \theta &= 0^o, \text{ direção de máxima subida} \\ 0 < \theta &< 90^o, \text{ direção de subida} \\ \theta &= 90^o, \text{ direção de subida nula} \\ \theta &= 180^o, \text{ direção de descida máxima .} \end{aligned}$$

O vetor unitário na direção de $-g$ é dado por

$$u = -\frac{g}{|g|} \tag{A.10}$$

de forma que a variação Δx seja proporcional a x:

$$\Delta x = \alpha u, \tag{A.11}$$

$\alpha \in I\!R^+$, ou seja α é positivo.

Então a direção de avanço, ou seja o vetor u, em uma iteração qualquer já está determinada, faltando apenas obter α, a quantidade de avanço nessa direção. E para isso usa-se o que se chama de *busca linear*. Portanto uma busca linear da função dada por $f(x + \alpha u)$ determina o valor ótimo de α. Em outras palavras esta busca produz o mínimo da função na direção de u. Assim, sendo x_k e u_k conhecidos, o próximo ponto x_{k+1} depende apenas do avanço α_k ou, para simplificar, α:

$$x_{k+1} = x_k + \alpha_k u_k \tag{A.12}$$

onde α_k é o avanço que causa o maior decréscimo em f, então o processo é repetido iterativamente até atingir-se o mínimo F_{min}.

Portanto o avanço α_k é obtido minimizando $f(x_k + \alpha_k u_k)$ no k-ésimo passo. Isto quer dizer que

$$\frac{df(x_k + \alpha u_k)}{d\alpha} = 0 = \nabla f(x_k + \alpha_k u_k)^T u_k, \tag{A.13}$$

em $\alpha = \alpha_k$ onde emprega-se o conceito de derivada direcional. Mas $x_{k+1} = x_k + \alpha_k u_k$ e isto acarreta $[\nabla f(x_{k+1})]^T u_k = 0$. Desde que $u_{k+1} = -\nabla f(x_{k+1})$, chega-se ao resultado $[u_{k+1}]^T u_k = 0$, ou seja, duas direções de descida consecutivas quaisquer são perpendiculares. A conclusão é que o método avança em *zig-zag* com ângulos retos, e se os α_k forem decrescentes a sequência deve convergir (Belegundu & Chandrupatla 1999) (Dennis & Snabil 1983). O algoritmo deste método está resumido abaixo:

Algoritmo do método de descida máxima

1. **Passo 1:** Selecionar uma solução inicial x_0 e um critério de término ε, com $0 \leq \varepsilon \leq 1$. Calcular $F_0 = f(x_0)$ e α_0. Fazer $k = 1$.

2. **Passo 2:** Calcular $F(k) = f(x_k)$ e α_k. Se $\alpha_{k-1} \leq \varepsilon\alpha_0$ então terminar o algoritmo; senão avançar para o próximo passo.

3. **Passo 3:** Calcular g_k e $u_k = -\frac{g_k}{|g_k|}$.

4. **Passo 4:** Realizar uma busca linear na direção de busca para encontrar α_k.

5. **Passo 5:** Calcular $x_{k+1} = x_k + \alpha_k u_k$ e fazer $k = k + 1$; em seguida, retornar para o passo 2.

A.1.2 Método de Newton

Primeiramente o método de Newton usa derivadas segunda ou seja a matriz Hessiana, por isso é considerado de segunda ordem, justamente pelo fato do cálculo das

direções u_k dependerem da Hessiana, além do gradiente. Isto em contraste ao método de descida máxima que é um método de primeira ordem, que solicita somente derivadas primeira ou seja apenas informação do gradiente. Então não é surpreendente que o método de Newton, quando converge, converge numa razão mais rápida que os métodos de primeira ordem, por simplesmente ser um método de segunda ordem (Belegundu & Chandrupatla 1999).

Pode-se dizer então que métodos mais avançados de minimização não se limitam a utilizar apenas aproximações de primeira ordem, utilizam também aproximações de segunda ordem da expansão de Taylor (A.6) para chegar ao mínimo (Adby & Dempster 1982) (Jennings & Mckeown 1993) (Chong & Zak 1996) (Kelley 1999). Assim a equação a minimizar toma a forma

$$f(x + \Delta x) \approx f(x) + g^T \Delta x + \frac{1}{2} \Delta x^T H \Delta x, \tag{A.14}$$

expandindo-a, temos,

$$f(x + \Delta x) \approx f(x) + \sum_{i=1}^{N} \frac{\partial f(x)}{\partial x_i} \Delta x_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \Delta x_i \frac{\partial^2 f(x)}{\partial x_i \partial x_j} \Delta x_j, \tag{A.15}$$

diferenciando com respeito a x_j, $j = 1, \ldots, N$, e igualando a zero,

$$\frac{\partial f(x)}{\partial x_i} + \sum_{j=1}^{N} \Delta x_i \frac{\partial^2 f(x)}{\partial x_i \partial x_j} = 0. \tag{A.16}$$

Sendo representado da forma matricial por:

$$g + H \Delta x = 0. \tag{A.17}$$

A Equação (A.17), nos leva à equação fundamental para todas as soluções de segunda ordem para o problema de minimização, ficando

$$\Delta x = -H^{-1} g, \tag{A.18}$$

onde esta equação representa o método de Newton. Os métodos de minimização usando técnicas de mínimos quadrados utilizam aproximações de segunda ordem. Seja por exemplo um conjunto de M equações não-lineares $f(x) = 0$,

$$f = [f_1(x), f_2(x), \ldots, f_M(x)]^T. \tag{A.19}$$

A função objetivo ($F = f(x)$), é dada agora pelo quadrado da norma de f (vetor dos resíduos), ficando:

$$F = \sum_{k=1}^{M} [f_k(x)]^2 = \|f\|^2 = f^T f. \tag{A.20}$$

E ao derivar F em relação a cada parâmetro x_i, tem-se

$$\frac{\partial F}{\partial x} = \sum_{k=1}^{M} 2 f_k(x) \frac{\partial f_k(x)}{\partial x_i}. \tag{A.21}$$

Finalmente,

$$\begin{bmatrix} \frac{\partial F}{\partial x_1} \\ \\ \frac{\partial F}{\partial x_2} \\ \\ \vdots \\ \\ \frac{\partial F}{\partial x_N} \end{bmatrix} = 2 \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_2}{\partial x_1} & \cdots & \frac{\partial f_M}{\partial x_1} \\ \\ \frac{\partial f_1}{\partial x_2} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_M}{\partial x_2} \\ \\ \vdots & \vdots & \ddots & \vdots \\ \\ \frac{\partial f_1}{\partial x_N} & \frac{\partial f_2}{\partial x_N} & \cdots & \frac{\partial f_M}{\partial x_N} \end{bmatrix} = \begin{bmatrix} f_1(x) \\ \\ f_2(x) \\ \\ \vdots \\ \\ f_M(x) \end{bmatrix}. \tag{A.22}$$

Na forma matricial fica,

$$g = 2J^T f. \tag{A.23}$$

Ao se diferenciar a Equação (A.21) em relação ao parâmetro x_j, tem-se

$$\frac{\partial^2 F}{\partial x_i \partial x_j} = 2 \sum_{k=1}^{M} \frac{\partial f_k(x)}{\partial x_i} \frac{\partial f_k(x)}{\partial x_j} + 2 \sum_{k=1}^{M} f_k(x) \frac{\partial^2 f_k(x)}{\partial x_i \partial x_j}, \tag{A.24}$$

e desprezando os termos de ordem superior, fica:

$$\frac{\partial^2 F}{\partial x_i \partial x_j} \approx 2 \sum_{k=1}^{M} \frac{\partial f_k(x)}{\partial x_i} \frac{\partial f_k(x)}{\partial x_j}. \tag{A.25}$$

Logo a equação anterior nos fornece a matriz Hessiana:

$$H = 2J^T J. \tag{A.26}$$

Se substituindo as equações (A.23) e (A.26) em (A.18) chega-se

$$\begin{aligned} \Delta x = -H^{-1}g &= -H^{-1}(2J^T f) \\ \Delta x &= -\frac{1}{H}(2J^T f) \\ H\Delta x = -(2J^T f) &= 2J^T J\Delta x = -2J^T f, \end{aligned}$$

$$J\Delta x = -f. \tag{A.27}$$

Sendo esta equação válida quando o número de equações M é iqual ao número de parâmetros N. E para os casos onde $M > N$, J (a matriz do Jacobiano) não seria mais quadrada. Então, a equação (A.27) deve ser reescrita como

$$u = \Delta x = -[J^T J]^{-1} J^T f. \tag{A.28}$$

A correção nos valores de x a cada iteração k é então realizada fazendo

$$x_{k+1} = x_k + u_k. \tag{A.29}$$

Portanto o uso de derivadas de segunda ordem faz com que o método de Newton tenha melhores resultados em comparação com método de primeira ordem. O método de Newton é baseado na facilidade de se minimizar uma forma quadrática, ou seja, em cada passo encontra-se uma aproximação quadrática para função objetivo, cujo o mínimo será o próximo passo. Além disso o método de Newton quando a função objetivo é quadrática, garante a convergência em apenas uma iteração, pois neste caso temos uma fórmula para obtenção direta do mínimo e não um método numérico (Belegundu & Chandrupatla 1999).

Então no caso geral, partindo de um inicializador $x(0)$ a sequência $x_{k+1} = x_k + u_k$ poderá levar ao mínimo procurado x_{min}, e por ser este método de segunda ordem, pode convergir rapidamente. Entretanto este é um problema do método de Newton, a convergência. Pois poderia-se afimar então que o método de Newton normalmente deveria convergir para a solução final x_{min}, havendo uma escolha adequada da estimativa ini-

cial x_0. Contudo, há situações nas quais o algoritmo falha (Jennings & Mckeown 1993), que podem ser:

- a convergência para uma solução não é garantida;
- a matriz $J^T J$ pode apresentar problemas de mal-condicionamento, podendo conduzir a resultados divergentes ou mesmo à sua singularidade;
- pode haver dificuldade na obtenção das derivadas de f_i, caso as funções não sejam definidas analiticamente;
- como somente uma solução é encontrada, mantidas as condições iniciais, nenhuma informação pode ser obtida correspondente ao mínimo global.

O algoritmo do Método de Newton é mostrado a seguir:

Algoritmo do método de Newton

1. **Passo 1:**Selecionar uma solução inicial x_0 e um critério de término ε. Calcular $J(0)$, $f(0)$. Se $\|f(0)\| \leq 0$, então terminar o algoritmo; senão fazer $k = 0$ e avançar para o passo 2
2. **Passo 2:** Calcular $J(k)u(k) = -f(k)$;
3. **Passo 3:** Calcular $x_{k+1} = x_k + u_k$;
4. **Passo 4:** Calcular $f(k+1)$. Se $\|f(k+1)\| \leq \varepsilon$ então terminar o algoritmo; senão, calcular $J(k+1)$, fazer $k = k+1$ e voltar para o passo 2.

A.2 Métodos globalmente convergentes

Nesta seção são mostradas algumas soluções para superar as limitações apresentadas anteriormente. Em seguida são abordados os métodos globalmente convergentes, onde o termo "globalmente convergente"significa "convergente para uma solução a partir de qualquer condição inicial", caso esta solução exista (Jennings & Mckeown 1993).

Muitos métodos globalmente convergentes adotam como critério de avaliação do progresso do algoritmo, a seguinte função objetivo:

$$F(x) = \sum_{i=1}^{N} f_i^2 = \|f\|^2 = f^T f. \tag{A.30}$$

O método clássico para minimizar qualquer função não-linear $F(x)$ é o método de Newton. Então sendo $x_{(k)}$ a k-ésima estimativa de F, o valor de F é dado por

$$F(x + \Delta x) = F(x_k) + \Delta x_T \nabla F(x_k) + \frac{1}{2}\Delta x^T \nabla^2 F(x_k)\Delta x + ..., \tag{A.31}$$

onde $\nabla F = g_k$ e $\nabla^2 F = H_k$ são o vetor gradiente e a matriz hessiana, contendo respectivamente as derivadas de primeira e segunda ordem de $F(x)$, e sendo definidos de forma similar às equações (A.4) e (A.5). Desprezando os termos a partir da terceira ordem e derivando a Equação (A.31), temos

$$\nabla F(x) \approx g_k + H_k(x_{k+1} - x_k). \tag{A.32}$$

Ao se igualar esta equação a zero, obtem-se:

$$H_k(x_{k+1} - x_k) = -g_k, \tag{A.33}$$

substituindo as seguintes expressões na Equação (A.33), tem-se

$$H = 2(J^T J + \sum_{i=1}^{N} f_i \nabla^2 f_i), \tag{A.34}$$

e

$$g \;\; = \;\; 2J^T f,$$

ficando finalmente

$$(J^T J + B)_{(k)} u_k = -(J^T)^{(k)} f_k, \tag{A.35}$$

onde

$$B = \sum_{i=1}^{N} f_i \nabla^2 f_i. \tag{A.36}$$

A matriz B pode ser interpretada como uma medida da não-linearidade das equações, sendo uma soma ponderada de suas derivadas de segunda ordem. Nota-se que B em comaparação com $J^T J$ normalmente é desprezível. Contudo, a sua ausência pode causar dificuldades no processo de minimização (Jennings & Mckeown 1993). E um método de fornecer B será tratado nas próximas seções, que é o método de Levenberg-Marquardt.

A.2.1 Método de Newton globalmente convergente

O objetivo geral desse método é corrigir os problemas de convergência indicados, para que se possa preservar a grande vantagem do método, a sua rapidez.

A função objetivo dada pela equação (A.20) é suposta ser bem comportada com respeito aos seus parâmetros x. E uma condição que precisa ser satisfeita é que $F(x)$ seja contínua e diferenciável em qualquer ponto. Se F for considerada bem comportada, e caso $J^T J$ seja não-singular, o método de Newton globalmente convergente (Jennings & Mckeown 1993) (Adby & Dempster 1982) (Kelley 1995) é obtido incluindo um fator α_k que controla a atualização dos parâmetros. Assim sendo este parâmetro obtido de uma busca linear: minimize $f(x_k + \alpha_k u_k)$, onde u_k é obtido como já foi visto. E com isso modificando a Equação (A.29) de forma a ter

$$x_{k+1} = x_k + \alpha_k u_k \tag{A.37}$$

onde $0 < \alpha_k \leq 1$, podendo este fator ser fixo ou variável (ou seja alterado a cada iteração). Quando o valor é fixo, escolhe-se um valor de forma que uma redução significativa em F seja obtida a cada iteração. O uso de um fator fixo, embora não seja a melhor solução, apresenta um menor custo computacional.

Por outro lado se o fator é variável, usa-se um fator modificado a cada iteração, pode-se determinar um α_k de forma que a função F seja minimizada ao longo da direção de busca $u_{(k)}$, requerendo uma busca iterativa a cada iteração, aumentando

o custo computacional. Ao invés disso, uma condição mais simplificada é adotada, escolhendo-se α_k de forma que uma redução significativa de F seja alcançada a cada iteração, junto com uma variação significativa no valor de x. Uma estratégia utilizada nesse caso é a aplicação da Regra de Armijo. A regra de Armijo é um dos vários métodos de busca linear inexatos que garante um grau suficiente de precisão para assegurar a convergência do algoritmo que é mostrada a seguir:

$$\begin{aligned} Regra1 : F_{(k+1)} &\leq F_{(k)} + \rho_1 \alpha_k g_k u_k \\ Regra2 : (g^T)_{(k+1)} u_k &\leq \rho_2 (g^T)_{(k)} u_{(k)}. \end{aligned}$$

A regra de Armijo requer dois parâmentros, que são denominados de ρ_1 e ρ_2 onde a Regra 1 está compreendida em um intervalo de $0 < \rho_1 \leq 1$ e enquanto a Regra 2, por $\rho_2 < \rho_1 \leq 1$.

A Regra 1 garante que a redução no valor da função é no mínimo proporcional ao seu valor se F for linear, contínua e descrecer na mesma taxa de sua inclinação inicial. Já a regra 2, forçando uma variação finita no vetor gradiente, impede que o passo se torne muito pequeno. O valor de α_k é então escolhido de forma que o passo se torne muito pequeno. O valor de α_k é então escolhido de forma que $\rho_1 \alpha_k \leq \alpha_{k+1} \leq \rho_2 \alpha_k$ seja satisfeito. A escolha de $\alpha_k = \rho_1 = \rho_2$ é a mais simples.

O algoritmo para esse método é dado a seguir:

Algoritmo de Newton globalmente convergente

1. **Passo 1:** Selecionar uma solução inicial $x_(0)$ e dois critérios de término ε_1 e ε_2. Calcular $J(0)$, $f(0)$ e g_k. Se $\|f(0)\| \leq \varepsilon_1$ ou $\|g_{(0)}\| \leq \varepsilon_2$ então terminar o algoritmo; senão fazer $k = 0$ e avançar para o passo 2;

2. **Passo 2:** Calcular $J(k)u_k = -f_(k)$.

3. **Passo 3:** Realizar uma busca ao longo da direção u_k de forma a determinar um passo α_k satisfazendo à Regra de Armijo;

4. **Passo 4:** Calcular $x_{k+1} = x_k + \alpha_k u_k$;

5. **Passo 5:** Calcular $f(k+1)$. Se $\|f(k+1)\| \leq \varepsilon_1$ então terminar o algoritmo; senão calcular $J_{(k+1)}$ e $g_{(k)}$. Se $\|g_{(k+1)}\| \leq \varepsilon_2$ então terminar o algoritmo; senão fazer $k = k+1$ e voltar para o passo 2.

A.2.2 Método de Levenberg-Marquardt

A matriz do Jacobiano, J, é a principal limitação do método de Newton, pois a matriz Hessiana, $J^T J$, precisa ser invertida a cada iteração, e pode apresentar problemas de mal-condicionamento, o que pode conduzir a resultados divergentes, e até mesmo à singularidade de $J^T J$. Portanto é essencial adotar uma estratégia alternativa quando ocorre um mal-condicionamento extremo. Então o método de Levenberg-Marquardt (Jennings & Mckeown 1993) (Kelley 1999) (Hartley & Zisserman 2000) (Motta, Carvalho & McMaster 2001) (Levemberg 1944) (Marquardt 1963) é como se fosse o método de Newton só que neste método modifica-se a matriz que deve ser invertida de forma a evitar o mal-condicionamento, fazendo $B = \mu I$ na Equação (A.35), onde μ é um escalar não-negativo e I é a matriz identidade, ficando (A.35) da forma

$$(J^T J + \mu I)_{(k)} u_{(k)} = -(J^T)_{(k)} f_{(k)}. \tag{A.38}$$

A modificação realizada altera apenas as componentes a_{ii} da diagonal principal de $J^T J$, resultando em componentes modificadas a'_{ii} de forma que $a'_{ii} = a_{ii} + \mu$. Algumas implementações realizam a modificação $a'_{ii} = a_{ii}(1+\mu)$, porém com efeitos similares.

A escolha do parâmetro μ deve ser feita com muito cuidado, pois se o valor de μ for muito grande, a matriz μI torna-se dominante em relação a $J^T J$, a direção de busca tende à direção do método de descida máxima e o módulo do vetor de busca torna-se pequeno. E da mesma foma, se o valor de μ for muito pequeno, as direções de busca são similares às do método de Newton. Consequentemente deve-se escolher então um valor de μ de forma a evitar as situações extremas . Esse valor precisa estar relacionado com a magnitude, ou seja, o módulo de $J^T J$, sendo uma medida fornecida pelo traço dessa matriz, da forma: $tr(J^T J) = \sum \lambda_i = \lambda(J^T J)_{ii}$ (que é a soma dos auto-valores ou a soma dos elementos da diagonal principal da matriz). Neste caso μ, tipicamente apresenta valores da ordem de 10^{-4} (Jennings & Mckeown 1993).

Com as devidas condições feitas, o valor do parâmetro μ pode ser alterado a cada iteração, de forma a obter uma redução do valor da função objetivo F, ou seja, em cada iteração k, $F_{(k)}$ é avaliado. Se $F_{(k)} < F_{(k-1)}$ o valor de μ_k é reduzido, e passa-se para a próxima iteração. Senão se $F_{(k)} > F_{(k-1)}$ o valor de μ é aumentado, e $F_{(k)}$ é novamente avaliado. O processo é repetido até que a condição $F(k+1) < F(k)$ seja enfim satisfeita.

O algoritmo de Levenberg-Marquardt é apresentado a seguir:

Algoritmo de Levenberg-Marquardt

1. **Passo 1:** Selecionar uma solução inicial $x(0)$ e dois critérios de término ε_1 e ε_2. Calcular $J(0)$, $f(0)$ e g_k. Se $\|f(0)\| \leq \varepsilon_1$ ou $\|g_{(0)}\| \leq \varepsilon_2$ então terminar o algoritmo; senão fazer $k = 0$ e avançar para o segundo passo;

2. **Passo 2:** Calcular $J(k)u_(k) = -f(k)$. Se $J(k)$ for singular então calcular $(J^T J)_{(k)}$, $\tau = tr(J^T J)_{(k)}$ e $(J^T J + \mu I)_{(k)}$. Calcular $(J^T J + \mu\tau I)_{(k)})u_{(k)} = -f_{(k)}$.

3. **Passo 3:** Realizar uma busca ao longo da direção u_k de forma a determinar um passo α_k satisfazendo à Regra de Armijo;

4. **Passo 4:** Calcular $x_{k+1} = x_k + \alpha_k u_k$;

5. **Passo 5:** Calcular $f(k+1)$. Se $\|f(k+1)\| \leq \varepsilon_1$ então terminar o algoritmo; senão calcular $J(k+1)$ e g_k. Se $\|g_{k+1}\| \leq \varepsilon_2$ então terminar o algoritmo; senão fazer $k = k+1$ e voltar para o passo 2.

Apêndice B

Sensores visuais

A tarefa de uma câmera como um sensor visual é medir a intensidade da luz refletida pelo objeto. Para este propósito, um elemento sensível a luz, chamado *pixel* (ou photosite), é empregado, que é capaz de transformar energia de luz em energia elétrica. Diferentes tipos de sensores estão disponíveis dependendo do efeito fisico explorado para realizar a transformação de energia. Os dipositivos mais amplamente usados são os sensores CCD e o CMOS baseados no efeito fotoelétrico de semicondutores.

Um sensor CCD (charge coupled device) é constituído por uma estrutura retangular de fotosensores. Devido ao efeito fotoelétrico, quando um fóton atinge a superfície do semicondutor, um número de elétrons livres são criados, portanto cada elemento acumula uma carga dependendo do tempo integral da iluminação incidente sobre o elemento fotosensível. Esta carga é então passada por um mecânismo de transporte (similar a um registrador analógico) para a saída amplificada, enquanto que ao mesmo tempo o pixel é descarregado. Então o sinal elétrico é processado para produzir o sinal de vídeo real.

Um sensor CMOS (Complementary Metal Oxide Semiconducter) é constituído por uma estrutura retangular de fotodiodos. A junção de cada fotodiodo é pré-carregada e é descarregada quando é atingida pelos prótons. Um amplificador integrado em cada pixel pode transformar esta carga em um nível de voltagem ou corrente. A principal diferença com o sensor CCD é que os pixels de um sensor CMOS são dispositivos não-integrados; depois de ativado eles medem durante todo o tempo, não volume. Desta

maneira, um pixel saturado nunca encherar-se e influencerá um pixel na vizinhança. Isto previne o efeito de *blooming*, que de fato afeta os sensores CCD (Sciavicco & Siciliano 1996).

Uma câmera é um sistema complexo que engloba vários dispositivos como sensor fotosensível (sensível a luz), um shutter, lente e processamento eletrônico analógico. A lente foca a luz refletida pelo objeto no plano imagem, a Figura (B.1) mostra esses dispositivos.

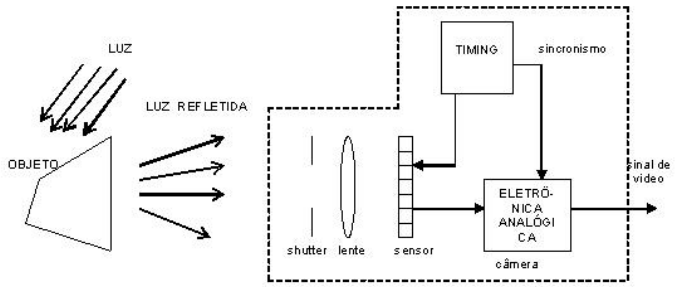

FIGURA B.1: Representação esquemática de um sistema de visão.

Apêndice C

Configuração de câmeras

Os sistemas servovisuais geralmente usam uma das duas configurações de câmera que são: montada no efetuador do manipulador ou fixa no espaço de trabalho.

A primeira configuração, frequentemente chamada de configuração câmera na mão (eye-in-hand), tem a câmera montada na extremidade do manipulador, ou seja, no efetuador. Nesta configuração há uma relação conhecida entre a pose da câmera e a pose do efetuador, que é representada por x_{ec}. A pose do objeto com relação a câmera é representado por $x_{c\lambda}$. A relação entre essas poses é ilustrada na Figura C.1.

A segunda configuração tem uma câmera fixa no espaço de trabaho. Neste caso, a câmera (ou as câmeras) esta relacionada com o sistema de coordenadas da base por x_{bc} e ao objeto por $x_{c\lambda}$. Neste caso, a imagem do objeto na câmera é, independente do movimento do robô (a não ser que o objeto esteja na extremidade do efetuador), vide Figura C.2. Os índices acima se referem:

- e ao efetuador;
- c à câmera;
- b à base do robô;
- λ ao alvo (objeto).

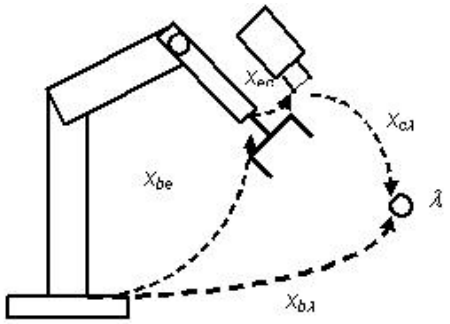

FIGURA C.1: Configuração câmera na mão (eye-in-hand) e eixos de coordenadas.

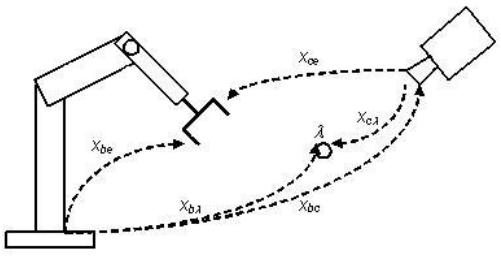

FIGURA C.2: Configuração câmera fixa no espaço de trabalho eixos de coordenadas (câmera, efetuador, e objeto).

Referências Bibliográficas

Adby, P. R. & Dempster, M. A. H. (1982), *Introduction to Optimization Methods*, Chapman an Hall, London.

Belegundu, A. D. & Chandrupatla, T. R. (1999), *Optimization Concepts and Applications in Engineering*, Prentice Hall.

Chen, T. H., Cheng, F. T. & Sun, Y. Y. (1992), 'Efficient algorithm for resolving manipulator redundancy - the compact method', *Proc. IEEE Robotics and Automat. Conf.*

Chong, E. K. P. & Zak, S. H. (1996), *An Introduction to Optimization*, Jonh-Wiley & Sons, New York, USA.

Dennis, J. E. J. & Snabil, R. B. (1983), *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice Hall.

Divelbiss, A. & Wen, J. T. (1994), 'Nonholonomic motion planning with inequality constraints', *Proc. IEEE Robotics and Automat. Conf.*

Gill, P. E., Murray, W. & Wright, M. H. (1986), *Practical Optimization*, Academic Press, London.

Hartley, R. & Zisserman, A. (2000), *Multiple View Geometry in Computer Vision*, Cambridge University Press.

Hidalgo, F. & Brunn, P. (1998), 'Robot metrology and calibration systems - a market review', *Industrial Robot* **25**(1), 42–47.

Hollerbach, J. & Ikits, M. (1997), 'Kinematic calibration using a plane constraint', *Proc. of the IEEE Conf. on Robotics & Automation* pp. 3191–3196.

Hollerbach, J. M. & Suh, K. C. (1985), 'Redundancy resolution of manipulators through torque optimization', *Proc IEEE Conf. Robotics and Automat* pp. 308–315.

Hollerbach, J. M. & Wampler, C. W. (1996), 'The calibration index and taxonomy for robot kinematic calibration methods', *The International Journal of Robotics Research*.

Jennings, A. & Mckeown, J. (1993), *Matrix Computation*, 2nd edn, John-Wiley & Sons.

Kelley, C. T. (1995), *Iterative Methods for Linear and Nonlinear Equations*, SIAM, Philadelphia, USA.

Kelley, C. T. (1999), *Iterative Methods for Optimization*, SIAM, Philadelphia, USA.

Khatib, O. (1986), 'Real-time obstacle avoidance for manipulators and mobile robots', *Int. J. Robotics Res.* **5**(1), 90–98.

Lei, S., Jingtai, L., Weiwei, S., Shuihua, W. & Xingbo, H. (2004), 'Geometry - based robot calibration method', *Pro. of th IEEE Conf. on Robotics & Automation* pp. 1907–1912.

Leite, A. C., Hsu, L., Lizarralde, F. & Zachi, A. R. L. (2004), 'Rastreamento de trajetórias por servovisão adaptativa', *Revista Controle & Automação* **15**(3), 309–319.

Levemberg, K. (1944), 'A method for the solution of certain non-linear problems in least-square', *Quart. Appl. Math.* **II**(2), 164–168.

Marquardt, D. W. (1963), 'An algorithm for least-squares estimation of nonlinear parameters', *J. Soc. Indust. Appl. Math.* **11**(2), 431–441.

Mavroidis, C., Dubowsky, S., Drouet, P., Hintersteiner, J. & Flanz, J. (1997), 'A systematic error analysis of robotic manipulators: Application to a high performance medical robot', *IEEE-International Conference on Robotics and Automation* pp. 980–985.

Meggiolaro, M. A. & Dubowsky, S. (2000), 'An analytical method to eliminate the redundant parameters in robot calibration', *IEEE-International Conference on Robotics & Automation* pp. 3609–3615.

Motta, J. M., Carvalho, G. C. & McMaster, R. S. (2001), 'Robot calibration using a 3d vision-based measurement system with a single camera', *Robotics and Computer Integrated Manufacturing* **17**, 487–497.

Motta, J. M. S. T. & s. McMaster, R. (1999), 'Modeling optmizing and simulating robot calibration with accuracy improvement', *Journal of The Brazilian Society of Mechanical Sciences* **21**(3), 386–402.

Romano, V. F. (2002), *Robótica Industrial*, 1a. edição edn, Editora Edgard Blucher ltda, São Paulo, SP.

Rosário, J. M. (2005), *Princípios de Mecatrônica*, Prentice Hall.

Schöer, K. (1993), *Theory of Kinematic Modelling an Numerical Procedures for Robot Calibration*, Chapman & Hall, London.

Schoröer, K., Albrigth, S. L. & Grenthlein, M. (1997), 'Complete, minimal and model-continuos kinematic models for robot calibration', *Robotics & Computer-Integrated Manufacturing* **13**(1), 73–85.

Sciavicco, L. & Siciliano, B. (1996), *Modelling and Control of Robot Manipulators*, McGraw-Hill.

Yoshikawa, T. (1985), 'Manipulability of robot mechanisms', *Int. J. Robotics Res.* **4**(2), 3–9.