Fabrício Nicolato

Estudo e Implementação de um Método de Cinemática Inversa Baseado em Busca Heurística para Robôs Manipuladores: Aplicação em Robôs Redundantes e Controle Servo Visual

Tese de Doutorado apresentada à Faculdade de Engenharia Elétrica e de Computação como parte dos requisitos para obtenção do título de Doutor em Engenharia Elétrica. Área de concentração: Automação.

Orientador: Marconi Kolm Madrid

Campinas, SP
2007

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE - UNICAMP

Nicolato, Fabrício
N543e Estudo e implementação de um método de cinemática inversa baseado em busca heurística para robôs manipuladores: aplicação em robôs redundantes e controle servo visual /Fabrício Nicolato. – Campinas, SP: [s.n.], 2007.

Orientador: Marconi Kolm Madrid.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Robótica. 2. Processamento de imagens. 3. Cenemática. 4. Heurística. 5. Inteligência artificial. 6. Servomecanismo. 7. Visão por computador. I. Madrid, Marconi Kolm. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título

Título em Inglês:	Heuristic search based inverse kinematics for robotic manipulators: application to redundant robots and visual servoing
Palavras-chave em Inglês:	Robotics, Inverse kinematics, State space search Heuristics, Visual servoing
Área de concentração:	Automação
Titulação:	Doutor em Engenharia Elétrica
Banca Examinadora:	Celso Pascoli Bottura, Edson Roberto De Pieri, João Maurício Rosário, José Raimundo de Oliveira e Leo Pini Magalhães
Data da defesa:	01/06/2007
Programa de Pós-Graduação:	Engenharia Elétrica

COMISSÃO JULGADORA – TESE DE DOUTORADO

Candidato: Fabrício Nicolato

Data da Defesa: 01 de junho de 2007

Título da Tese: "Estudo e Implementação de um Método de Cinemática Inversa Baseado em Busca Heurística para Robôs Manipuladores: Aplicação em Robôs Redundantes e Controle Servo Visual"

Prof. Dr. Marconi Kolm Madrid (Matr. 262706): _____

Prof. Dr. João Mauricio Rosario: _____

Prof. Dr. Edson Roberto de Pieri: _____

Prof. Dr. José Raimundo de Oliveira: _____

Prof. Dr. Léo Pini Magalhães: _____

Prof. Dr. Celso Pascoli Bottura: _____

Resumo

Esta tese trata o problema da resolução do modelo cinemático inverso para manipuladores industriais redundantes ou não. O problema foi abordado por um método de busca heurística no qual a solução da cinemática inversa é construída passo a passo calculando-se a contribuição do movimento de apenas uma junta a cada iteração. Dessa forma, o problema n-dimensional é transformado em problemas unidimensionais mais simples, cuja solução analítica tanto para juntas rotacionais quanto para juntas prismáticas é apresentada em termos da representação de Denavit-Hartenberg. O método proposto não possui singularidades internas. Além disso, o método foi expandido para incorporar informações de sensores externos visando fazer com que o processo seja mais robusto a incertezas nas modelagens envolvidas. Foram realizadas diversas simulações e comparações com técnicas tradicionais que evidenciaram as vantagens da abordagem proposta. O trabalho também englobou o projeto e a construção de um ambiente experimental e a implementação das técnicas desenvolvidas na parte teórica. Desenvolveu-se um sistema com um robô planar redundante de 3 DOF, assim como seus sistemas de controle, acionamento e interfaceamento usando técnicas de sistemas *hardware-in-the-loop* e lógica programável. As técnicas desenvolvidas foram aplicadas no ambiente experimental demonstrando características como: facilidade de lidar com redundâncias, capacidade de resolução em tempo real, robustez a incertezas de parâmetros etc.

Palavras-chave: Robótica, cinemática inversa, busca em espaços de estados, heurística, controle servo visual.

Abstract

This thesis deals with the problem of solving the inverse kinematics model of redundant and non-redundant industrial manipulators. The work was developed in a theoretical and a practical part. The problem was approached by an heuristic search method in which the solution of the inverse kinematics is built step by step calculating the movement contribution of just a single joint for each iteration. In that way, the n-dimensional problem is transformed in simpler one-dimensional problems, whose analytic solution for both rotational joints and prismatic joints is presented in terms of the Denavit and Hartenberg representation. The proposed method does not possess internal singularities. Furthermore, the method was expanded to incorporate information of external sensor in order to make the process more robust to uncertainties in the involved modelings. Several results of simulations and comparisons with traditional techniques, which evidence the advantages of the proposed approach, are presented. The work also included the construction of an experimental environment and the implementation of the techniques developed in the theoretical part. The details of a system with a 3-DOF redundant robot as well as its control system, drivers and interfaces using hardware-in-the-loop techniques and programmable logic are presented. The developed techniques were applied in the experimental environment are demonstrating their efficiency and evidencing characteristics like: easiness of dealing with redundancies, real time capacity, robustness for parameters uncertainties etc.

Keywords: Robotics, inverse kinematics, state space searches, heuristics, visual servoing.

Agradecimentos

Ao professor Madrid pela oportunidade de aprofundar meus estudos, pela orientação e paciência durante estes anos;

Aos colegas do LSMR, LabSim, LADIME e LCEE, destacando os senhores Edson A. Vendrúsculo e Filipe I. Fazanaro, pelo companheirismo e contribuições no trabalho;

Aos antigos companheiros de república Jim, Kenji e Massakiti pela amizade;

Em especial, gostaria de agradecer ao amigo Eduardo Bonani cuja ajuda no desenvolvimento da estrutura mecânica do robô foi essencial para a realização desse trabalho;

Esta tese foi financiada pela FAPESP, projeto número 02/05046-4.

A minha esposa Mônica,
ao meu filhinho Pedro,
a minha irmã Deise e seu marido Ubaldino,
aos meus pais Tercílio e Lourdes.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Notação, Terminologia e Siglas

Notação e Terminologia

$\mathbf{p}$: letra minúscula em negrito representa um vetor;
$\mathbf{A}$: letra maiúscula em negrito representa uma matriz;
γ, Z: letra minúscula ou minúscula sem estar em negrito representa um escalar;
${}^{c}(:)_{a}$: indica que uma entidade representada pelo sistema de coordenadas a está expressa relativamente ao sistema de coordenadas c;
$f(t)$: Letras maiúsculas ou minúsculas, em negrito ou não, seguidas por outras letras entre parênteses indicam funções. Ex: $h(t)$ é uma função escalar da variável escalar t, $\mathbf{A}(\mathbf{q})$ é uma função matricial do vetor $\mathbf{q}$.
$SE(3)$: Designa um grupo especial que representa posições e orientações no espaço euclidiano.
Pose: significa a posição e orientação da garra do robô, objetos etc; sendo definida no grupo $SE(3)$.
A presença de um subíndice 'd' em um escalar, vetor, matriz ou função indica uma quantidade desejada (referência).
O termo **sensor externo** refere-se a sensores cujos parâmetros intrínsecos, de seu modelo interno, não dependem dos parâmetros do robô.
$\mathcal{F}_i$ designa o i-ésimo sistema de coordenadas cartesianas associado. O sub-índice i indica o significado do respectivo sistema, como: $\mathcal{F}_0$ indica o sistema de coordenadas inercial ou fixo ou do mundo, $\mathcal{F}_c$ indica o sistema associado a uma câmera e $\mathcal{F}_i,\ i = 1,\ 2, \cdots$ indica o sistema associado a cada elo do robô.
Robô: Durante o texto do trabalho, as palavras robô, manipulador e máquina são usadas indistintamente com o intuito de representar todo o aparato, tanto mecânico, quanto eletrônico de um braço robótico de cadeia serial aberta [1].
Trajetória significa uma função do tempo, definida no espaço de trabalho ou de juntas, que o robô deverá rastrear.
Caminho é um conjunto de pontos, definido no espaço de trabalho ou de juntas, pelos quais o robô deverá passar, mas sem uma lei de tempo associada.

O termo **u.m** significa unidade de medida de uma grandeza qualquer.

Siglas

CAD: *Computer aided design*
CLF: *Control Lyapunov function*
CLIK: *Closed loop inverse kinematics*
DLS: *Damped least squares*
DOF: *Degrees of freedom*
DSP: *Digital signal processor*
DH: *Denavit-Hartenberg*
HIL: *Hardware-in-the-loop*
HVS: *Hybrid visual servoing*
IBVS: *Image based visual servoing*
PBVS: *Position based visual servoing*
PLD: *Programmable logic device*
PWM: *Pulse Width Modulation*
QPS: *quadros de vídeo por segundo*
SSP: *State space search*

Capítulo 1

Introdução

São apresentados aspectos que motivaram o desenvolvimento do trabalho, suas contribuições mais importantes e uma revisão do estado da arte das pesquisas relacionadas ao estudo apresentado na tese.

1.1 Motivação

De maneira semelhante ao que ocorre com os seres humanos, os robôs industriais ou manipuladores robóticos executam tarefas no espaço de trabalho, geralmente descrito por posições e orientações do seu elemento terminal ou garra; sendo, os movimentos dessas máquinas gerados no espaço de juntas ou de atuação. Dessa forma, é necessário que haja um sistema que possa traduzir tarefas descritas no espaço de trabalho para tarefas no espaço de juntas. Esse processo é conhecido como determinação da cinemática inversa do robô [1, 2, 3] e é uma tarefa básica, porém fundamental, em robótica.

A cinemática inversa, de modo geral, envolve a resolução de sistemas de equações altamente não lineares. Um fator complicador é a existência de redundâncias cinemáticas, que significam que o robô tem mais possibilidades de movimento ou graus de liberdade (DOF) do que aqueles estritamente necessários para a realização da tarefa em questão. Nesse caso, como o sistema é sobre-determinado, é necessário o emprego de algum tipo de técnica de otimização para se determinar uma solução “adequada" dentre as infinitas possibilidades existentes.

Embora diversos conceitos e técnicas, tais como algoritmos genéticos [4], lógica nebulosa [5], redes neurais [6], busca heurística [7] etc, sejam usadas para abordar esse assunto, a grande maioria das estratégias para resolver o problema da cinemática inversa para robôs industriais é baseada na cinemática diferencial [1, 8], que relaciona as velocidades das juntas às velocidades da garra através do jacobiano do robô. As posições das juntas são então determinadas via integração numérica. Algu-

mas das razões que fazem com que as estratégias de cinemática diferencial sejam as preferidas entre os pesquisadores de robótica são:

- São abordagens generalizantes;
- São tratáveis matematicamente via teoria de matrizes e geometria diferencial [9, 10];
- Os algoritmos de cinemática inversa podem ser formulados como sistemas de controle em malha fechada e suas convergências analisadas pela teoria da estabilidade de Lyapunov [1, 11].

Embora sejam largamente utilizadas e tenham um forte apelo analítico, as estratégias baseadas em cinemática diferencial possuem limitações severas [12, 13, 14], tais como:

- Singularidades matemáticas;
- Dificuldade para lidar com restrições de juntas;
- Desempenho deteriorado na presença de redundâncias;

Em particular, o problema de singularidades matemáticas ocorre quando, em certas configurações no espaço de juntas, a matriz jacobiana perde posto (*rank*). À medida que o robô se aproxima dessas configurações no espaço de juntas, o jacobiano torna-se mal condicionado numericamente, e sua inversão numérica leva à geração de velocidades extremamente elevadas nas juntas, o que é impraticável do ponto de vista de engenharia aplicada a sistemas físicos. No caso de robôs redundantes, um problema importante é a falta de repetibilidade, onde trajetórias periódicas (repetitivas) no espaço de trabalho não são mapeadas em trajetórias repetitivas no espaço de juntas. Em casos extremos, essas trajetórias chegam a apresentar comportamento caótico [13].

Esses problemas impõem limitações técnicas e econômicas para a utilização mais ampla dos robôs manipuladores. Exemplos dessas limitações são:

- Redução do espaço de trabalho útil dos robôs devido as singularidades matemáticas;
- Redução na precisão no processo de rastreamento de trajetórias cartesianas;
- Necessidade de controladores mais sofisticados;
- Diminuição da vida útil da máquina.

A verificação da execução da tarefa no espaço de trabalho é feita, em geral, de forma indireta através do cálculo da cinemática direta, que é a determinação da posição e orientação da garra do robô através de uma função matemática que tem como parâmetros elementos construtivos da máquina,

como: comprimentos de elos, ângulos entre elos etc; e como variáveis, os dados vindos dos sensores (*encoders/resolvers*) de junta. Este equacionamento é relativamente simples de ser obtido e pode ser realizado de modo sistemático [15, 1]. Porém, a obtenção precisa da pose do robô de forma indireta, depende de um ajuste preciso dos seus parâmetros cinemáticos e do modelamento preciso da tarefa a ser realizada no espaço de trabalho [2]. Isso se deve ao fato de que nenhum parâmetro físico pode ser determinado de forma exata e a obtenção do modelo cinemático fica extremamente dificultada quando o robô realiza tarefas de manipulação de objetos complexos, com tamanhos e orientações variáveis [16].

Para evitar tais problemas, é desejável do ponto de vista prático a inclusão de esquemas de sensoriamento que sejam menos dependentes de variações dos parâmetros da máquina, e que permitam uma maior adaptabilidade do sistema de controle. Para lidar com o problema de incertezas cinemáticas e dinâmicas várias técnicas foram propostas recentemente [17, 18, 19, 16, 20, 21]. Tais técnicas são também baseadas em cinemática diferencial, cujos parâmetros são ajustados durante a evolução do processo para que o sistema de controle possa obter resultados satisfatórios. Todavia, como essas idéias fazem uso da matriz jacobiana, estão sujeitas às mesmas dificuldades mencionadas anteriormente.

Embora exista uma grande variedade de sensores que podem ser usados neste intento, tais como: ultra-som, laser etc, as câmeras de vídeo têm sido largamente usadas e preferidas nesse sentido devido a alguns fatores:

- Uma imagem grande quantidade de informações pode ser obtida de uma imagem. Algumas dessas informações são: cor, posição, orientação, textura, tamanho etc;
- As câmeras de vídeo oferecem a possibilidade de se controlar máquinas de forma similar àquela que os seres humanos usam no controle de seus movimentos, uma vez que muitas tarefas cotidianas são controladas e/ou verificadas por meio da visão. Um exemplo disso são as tarefas de limpeza.

A utilização de informações provenientes de câmeras de vídeo para realizar o controle de robôs é conhecida como controle servo visual e vem sendo motivo de grande interesse por parte da comunidade científica internacional [22, 23, 24, 25]. O controle servo visual pode ser dividido em duas abordagens básicas [23]. Na primeira, procura-se determinar as posições e orientações (pose) dos objetos envolvidos na tarefa em questão. Já a segunda visa realizar o controle por meio de comparações feitas diretamente nas imagens. Cada uma possui vantagens e desvantagens, sendo que em muitos casos, os melhores resultados são obtidos combinando-se ambas [26]. Todavia, o controle servo visual possui um conjunto de desafios, dos quais destacam-se os seguintes [27]:

- Análise de estabilidade;

- Geração de trajetórias a serem realizadas pelo robô que sejam factíveis e suaves dentro de seu espaço de trabalho;
- Manter as características de interesse nas imagens dentro do campo de visão da câmera;
- Manipulação de singularidades.

De acordo com as exposições anteriores, a motivação deste trabalho está na necessidade de se obter um método simples, tanto teórica quanto computacionalmente, para resolver o problema do cálculo da cinemática inversa dos robôs de cadeia serial aberta que simplifique as dificuldades impostas a outros processos que a utilizam, tais como a resolução de redundâncias, falhas em juntas, controle servo visual etc. Nesse contexto, a cinemática inversa, mesmo sendo um problema clássico, básico e amplamente estudado em robótica, ainda não possui solução adequada que facilite sua integração com processos de níveis hierárquicos distintos no controle de sistemas robotizados.

Nesse mesmo contexto, métodos alternativos de solução do problema da cinemática inversa baseados em conceitos de busca heurística [28, 29, 30, 7, 31], que procuram resolver o problema movimentan–se, de forma simulada, apenas uma junta de cada vez e determinando sua contribuição para a solução geral, oferecem a possibilidade de solucionar esse problema de forma simples e em tempo real, necessitando apenas do modelo cinemático direto. Embora atrativas, tais técnicas encontram-se apenas no nível conceitual, com resultados de simulações, e carecem de implementações práticas em robótica[1]. A visão computacional é usada como um módulo que possibilita maior robustez e flexibilidade ao sistema, uma vez que a informação visual pode oferecer informações externas, que não necessariamente dependam do modelagem cinemática do robô.

1.2 Proposta e Contribuições

Neste trabalho, apenas os parâmetros do modelo cinemático são levados em conta para o desenvolvimento dos algoritmos, implicando que o robô é considerado como um “posicionador perfeito". De fato, a maioria dos fabricantes disponibiliza robôs com acesso apenas a controladores cinemáticos [24, 14]. A dinâmica dos robôs é geralmente levada em consideração em controladores de mais baixo nível. Mesmo para aquelas tarefas mais complexas onde poder-se-ia supor a utilização do modelo dinâmico como condição imprescindível para sua realização, como por exemplo as que requeiram altas velocidades, a obtenção da solução cinemática continua sendo uma etapa fundamental. Adicionalmente, as técnicas estudadas e desenvolvidas neste trabalho de tese podem ser vistas pelos níveis mais baixos na arquitetura de sistemas robóticos como “geradoras de referências e/ou trajetórias", e dessa

[1] Durante o desenvolvimento desse trabalho, apesar de intensa procura nas bases de dados disponíveis, não foi possível encontrar registro de implementações e testes práticos da referida técnica para robôs industriais

forma, podem ser adicionadas, de forma modular, aos sistemas de controle dinâmico sofisticados sem que haja perda da generalidade da proposta.

Nesse sentido, são propostas e desenvolvidas as seguintes contribuições para o estado da arte do estudo de cinemática inversa aplicada a sistemas robóticos:

1. Formulação de expressões gerais para o método de busca baseadas na formulação de Denavit-Hartenberg para a cinemática direta, englobando o rastreamento de pontos e trajetórias definidas em posição e orientação;

2. Formulação e análise do método proposto por meio de buscas em espaços de estados guiadas por funções de controle de Lyapunov;

3. Inclusão de informações adicionais no processo de resolução de cinemática inversa via busca heurística, por exemplo uma câmera de vídeo, para aumentar sua robustez e flexibilidade;

4. Análise do comportamento do método de busca na presença de incertezas de modelagem;

5. Comparação com abordagem afim em diversas situações, como: singularidades, redundâncias, limites de junta e controle servo visual;

6. Desenvolvimento de um sistema experimental com robô redundante planar de 3 graus de liberdade (DOF), assim como todo seu sistema de controle, processamento e interfaceamento visando a absorção e desenvolvimento de tecnologia nacional aplicada a sistemas robóticos e áreas afins;

7. Implementação prática do método.

1.3 Trabalhos Relacionados

Nessa seção, é apresentado um apanhado de diversas contribuições referentes ao estudo da cinemática inversa e controle servo visual presentes na literatura. Estes trabalhos, embora representem apenas uma pequena parte do universo das pesquisas relacionadas ao tema da tese, fornecem exemplos da origem e evolução das pesquisas nos temas em questão, fundamentação para as propostas apresentadas e subsídios para a contextualização deste trabalho.

Técnicas para a determinação da cinemática inversa têm sido alvo de consideráveis esforços de pesquisas há vários anos. Soluções analíticas podem ser encontradas apenas para manipuladores com estruturas geométricas simples [32], por exemplo, quando o grau do polinômio característico é menor ou igual a 4 [33] . Para todas aquelas estruturas onde uma solução analítica fechada não pode ser encontrada, ou é de difícil obtenção, um grande número de técnicas algoritmicas têm sido

desenvolvidas. A literatura de robótica apresenta diversas técnicas baseadas em métodos iterativos para o problema da resolução do modelo cinemático inverso.

Em geral, estas técnicas são baseadas na inversão do mapeamento estabelecido entre o espaço de juntas e o espaço de tarefas pela matriz jacobiana do manipulador. Um dos trabalhos pioneiros neste campo foi proposto por Whitney [34], que usa a pseudo-inversa da matriz jacobiana para obter as velocidades das juntas correspondentes à uma dada velocidade da garra do robô. Para resolver as singularidades do modelo cinemático, a inversão da matriz jacobiana por meio de uma técnica de mínimos quadrados amortecidos foi proposta posteriormente por Nakamura e Hanafusa [35] e Wampler [36]. Resultados experimentais desta técnica podem ser encontrados em [37]. Porém, na vizinhança de singularidades, a precisão do rastreamento de trajetórias é extremamente degradada [12]. Outras técnicas , tais como juntas virtuais [38, 39] e transformações do espaço de trabalho [40] têm sido propostas também para a manipulação de singularidades matemáticas.

Uma vez que os ângulos das juntas são obtidos por integração numérica, podem haver erros de integração. Algoritmos capazes de lidar com esses problemas são baseados numa correção via realimentação. Estes algoritmos são chamados de cinemática inversa de malha fechada (CLIK). Um dos primeiros algoritmos de CLIK, baseado na transposta do jacobiano, foi proposto independentemente por Balestrino et al. [41], e Wolovich e Elliott [42]. Algoritmos similares usam tanto a pseudo-inversa do jacobiano [43], quanto a inversa dos mínimos quadrados amortecidos [44, 37, 45]. Um algoritmo para lidar com as restrições de velocidade e aceleração foi proposto por Antonelli et al. [14], embora, como a maioria dos algoritmos existentes para esse fim, seja suscetível a singularidades matemáticas e possua uma formulação matemática bastante complexa.

No caso de robôs redundantes, o movimento das juntas não é determinado somente pelas exigências da tarefa, uma vez que o manipulador possui mais graus de liberdade do que o requerido pela tarefa. De maneira a resolver tal indeterminação, tarefas adicionais, ou sub-tarefas, são adicionadas ao processo de modo a otimizar um determinado critério como a norma euclidiana [1] ou a norma infinita [46] instantânea das velocidades das juntas. Estes tipos de abordagem impõem esquemas de prioridade de tarefas para o processo e, em geral, usam ponderações na pseudo-inversa do jacobiano para resolver as sub-tarefas [47, 48]. De modo geral, as sub-tarefas são colocadas como problemas de otimização, nos quais deseja-se otimizar funções como: distância de obstáculos e limites de juntas, índices de manipulabilidade etc.

Uma nova e crescente aplicação de dispositivos robóticos é denominada Robôs de Serviço (*service robots*). Esta classe de robôs deve possuir características e comportamentos parecidos com os de seres vivos, como por exemplo: movimentação, comunicação, inteligência etc. Neste sentido, Potkonjak et al. [49] e Potkonjak et al. [50] propuseram uma técnica denomina “fadiga virtual" para obter movimentos similares aos dos seres humanos em braços robóticos. Nesta mesma linha, Ari-

moto [10] introduz dois conceitos de geometria diferencial e uma lei de controle baseada na matriz jacobiana transposta do robô com realimentação de velocidade para o caso da movimentação de um braço robótico.

Técnicas estatísticas e de inteligência artificial também têm sido utilizadas para o aprendizado e/ou resolução da cinemática inversa de robôs manipuladores. D'Souza et al. [51] apresenta a aplicação de um algoritmo de aprendizagem estatística para um robô humanóide de 30 graus de liberdade. Na área de inteligência artificial, Chapelle e Bidaud [4] apresentam uma aproximação para a cinemática inversa de manipuladores gerais com 6 juntas rotacionais via programação genética. Já Wei [52] propõe a determinação da cinemática inversa de orientação para um manipulador através do uso de redes neurais. Zhang et al. [6] usam uma rede neural dual para o controle de manipuladores redundantes, enquanto Wang [53] propõe uma rede neural recorrente para o mesmo propósito. Xu e Nechyba [54] usam uma técnica de lógica nebulosa (*fuzzy*) para a resolução da cinemática inversa de manipuladores genéricos, tanto redundantes quanto não redundantes. Her et al. [55] propõe um algoritmo recursivo baseado em lógica nebulosa no qual o número de iterações é reduzido por meio de algoritmos genéticos. Já Antonelli e Chiaverini [5] apresentam a aplicação de um sistema baseado em lógica nebulosa para manipuladores subaquáticos.

Abordagens alternativas para a resolução da cinemática inversa aplicada a manipuladores robóticos foram propostas por Wang e Chen [28], Madrid e Maciá [29], Madrid e Palhares [7] e Ahuactzin e Gupta [31]. Estas técnicas são baseadas em algoritmos de busca heurística nos quais a solução da cinemática inversa é construída passo-a-passo, permitindo-se que apenas uma das juntas se mova a cada instante no processo simulado e, posteriormente, com a solução encontrada, todas as juntas são colocadas em movimento no processo. Welman [30] aplicou tais técnicas para computação gráfica visando a animação de figuras, enquanto Canutescu e Dunbrack [56] a aplicaram em biologia visando o projeto de medicamentos.

No que se refere à inclusão de informações visuais no sistema de controle de robôs, o trabalho pioneiro é creditado a Shirai e Inoue [57]. Sanderson e Weiss [58] apresentaram uma classificação para as diferente arquiteturas usadas nos esquemas de controle por visão. Espiau et al. [22] propuseram a incorporação de características de imagem a serem controladas. Um tutorial bastante detalhado sobre controle com realimentação visual é apresentado em [23]. Chaumette [27] traz uma análise cuidadosa sobre a estabilidade de sistemas de servo controle visual, tanto aqueles baseados em posição quanto aqueles baseados em imagem. Melhorias no desempenho desses sistema foram propostas inicialmente por Malis et al. [26] e Corke e Hutchinson [59]. Essas são estratégias híbridas de controle servo visual que procuram minimizar ou eliminar alguns dos problemas de singularidades matemáticas dos métodos até então propostos.

Para lidar com inconvenientes das falhas de convergência e geração de trajetórias inadequadas

pelo sistemas IBVS, Chesi et al. [60] propuseram um esquema de controle servo visual desenvolvido especialmente para tarefas de controle nas quais a câmera está muito deslocada de sua pose desejada. Um sistema de controle servo visual baseado em imagem é apresentado em [61], no qual as características da imagem são obtidas por meio de iluminação estruturada.

Um grande problema associado ao controle servo visual é a dependências dos parâmetros de câmera e da distância da câmera até o objeto filmado. Sendo assim, Malis [62] desenvolveu um método de controle servo visual robusto a variações nos parâmetros intrínsecos da câmera. Nessa mesma linha, Liu et al. [16] desenvolveram um esquema de controle servo visual dinâmico baseado em imagem que se fundamenta numa matriz de iteração que não depende da profundidade. Este esquema não depende de qualquer calibração de câmera e apresenta bons resultados práticos.

De modo a lidar com incertezas na modelagem do robô e da câmera, Zachi et al. [19] propuseram um esquema de controle servo visual adaptativo que leva em consideração a dinâmica do robô na lei de controle. São apresentadas simulações para o modelo dinâmico, mas a implementação prática leva em consideração apenas os parâmetros cinemáticos da máquina. Akella [63] apresenta um sistema de controle adaptativo de duas camadas, no qual a interna procura estabilizar a dinâmica rápida do robô, enquanto que a camada externa foi projetada para lidar com a dinâmica mais lenta envolvida com o processo de visão. O sistema é estável, mas não assintoticamente. São apresentadas simulações para um robô de 2 GDL. Uma técnica adaptativa que lida com incertezas nos parâmetros cinemáticos e dinâmicos é apresentada em [17, 18], com extensão para robôs redundantes e inclusão dos modelos dinâmicos dos atuadores em [16].

Como é possível perceber através dessa revisão bibliográfica sucinta, há na atualidade uma grande motivação da comunidade científica para pesquisa de temas relacionados ao presente trabalho, indicando que esta área deverá ter grandes avanços, tanto no sentido científico quanto tecnológico.

1.4 Organização do Texto

Este restante do texto está organizado da seguinte maneira:

O **capítulo 2** traz uma breve apresentação dos principais conceitos envolvidos na resolução da cinemática inversa via cinemática diferencial, assim como os conceitos fundamentais de busca heurística e controle servo visual necessários para o entendimento das técnicas desenvolvidas nos capítulos subseqüentes;

O **capítulo 3** traz o desenvolvimento teórico, as expressões gerais e os algoritmos propostos desde o início do trabalho de tese;

O **capítulo 4** mostra resultados alcançados com a simulação do método proposto para o rastreamento de trajetórias definidas em posição e orientação. São apresentados também resultados de com-

parações com abordagens tradicionais para a manipulação de singularidades, redundâncias, limites de juntas e controle servo visual;

O **capítulo 5** apresenta uma visão geral do sistema experimental desenvolvido, a metodologia usada de implementação prática do método e diversos resultados experimentais do método em diversas situações de modo a demonstrar a sua aplicabilidade;

O **capítulo 6** apresenta algumas considerações e conclusões sobre o desenvolvimento do trabalho, assim como um conjunto de sugestões para trabalhos futuros;

O **apêndice A** apresenta detalhes do projeto do ambiente experimental desenvolvido.

Capítulo 2

Fundamentação Teórica

São apresentados os conceitos básicos envolvidos no rastreamento de trajetórias cartesianas e busca heurística em espaços de estados. Estes conceitos constituem o embasamento para o entendimento das técnicas apresentadas nos capítulos posteriores.

2.1 Cinemática de Robôs de Cadeia Serial

2.1.1 Cinemática Direta

De maneira a facilitar a descrição da localização de cada elo (*link*) do robô, um sistema de coordenadas é associado a cada elo - o sistema de coordenadas i é associado ao elo i. Denavit e Hartenberg [64] propuseram um método matricial sistemático para associar sistemas de coordenadas a cada elo rígido de uma cadeia articulada. Nessa técnica, a cinemática direta de um manipulador de n elos rígidos, relacionando a pose (posição e orientação) do último elo, $\mathcal{F}_n$, em relação ao sistema de coordenadas da base, $\mathcal{F}_0$, é obtido pela equação 2.1 [1],

$$ {}^0\mathbf{T}_n = {}^0\mathbf{A}_1 \ldots {}^{i-1}\mathbf{A}_i \ldots {}^{n-1}\mathbf{A}_n = \mathbf{k}(\mathbf{q}), \tag{2.1} $$

sendo que ${}^0\mathbf{T}_n$ representa a pose do elemento terminal do robô e $\mathbf{q}$ o vetor de coordenadas generalizadas das juntas dado por

$$ q_i = \begin{cases} \theta_i & \text{para juntas rotacionais} \\ d_i & \text{para juntas prismáticas} \end{cases}, \tag{2.2} $$

para $i = 1, \ldots, n$. Esta técnica define um modo sistemático de associação de sistemas coordenados

aos elos do robô, resultando numa transformação homogênea, que relaciona a junta i com a junta $i-1$, dada pela equação 2.3

$$ {}^{i-1}\mathbf{A}_i = \begin{bmatrix} \cos\theta_i & -\operatorname{sen}\theta_i \cos\alpha_i & \operatorname{sen}\theta_i \operatorname{sen}\alpha_i & a_i\cos\theta_i \\ \operatorname{sen}\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i \operatorname{sen}\alpha_i & a_i \operatorname{sen}\theta_i \\ 0 & \operatorname{sen}\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.3} $$

sendo que ${}^{i-1}\mathbf{A}_i$ é chamada de matriz de Denavit-Hartenberg (DH) e θ_i, α_i, a_i e d_i são os parâmetros da i-ésima junta. Em notação matricial, tem-se

$$ {}^{0}\mathbf{T}_n = \begin{bmatrix} {}^{0}\mathbf{R}_n & {}^{0}\mathbf{p}_n \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.4} $$

e

$$ {}^{i-1}\mathbf{A}_i = \begin{bmatrix} {}^{i-1}\mathbf{R}_i & {}^{i-1}\mathbf{p}_i \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{2.5} $$

sendo $\mathbf{R}$ uma matriz 3×3 que representa a orientação do i-ésimo sistema de coordenadas e $\mathbf{p}$ um vetor 3×1 representando sua respectiva origem.

2.1.2 Cinemática Inversa

A configuração das juntas dada em termos da pose atual do robô é então dada pela equação 2.6

$$ \mathbf{q} = \mathbf{k}^{-1}(\mathbf{T}), \tag{2.6} $$

que é a solução da cinemática inversa. Em geral, esta solução não é única e, para algumas classes de manipuladores não existe uma solução analítica na forma fechada. Se o manipulador é redundante, a solução da cinemática inversa é sub-determinada. Se nenhuma solução matemática puder ser determinada para uma pose particular, esta configuração é dita singular.

Para resolver a cinemática inversa, uma técnica comumente utilizada na literatura baseia-se no uso da cinemática diferencial [1], onde as velocidades $\dot{\mathbf{q}}$ com as velocidades linear $\dot{\mathbf{p}}$ e angular $\mathbf{w}$ da garra do robô, agrupadas como $\mathbf{v} = [\dot{\mathbf{p}} \;\; \mathbf{w}]^T$, são dadas pelo mapeamento definido pela equação 2.7.

$$\mathbf{v} = \mathbf{J}\dot{\mathbf{q}} = \begin{bmatrix} \mathbf{J}_p \\ \mathbf{J}_o \end{bmatrix} \dot{\mathbf{q}}, \tag{2.7}$$

sendo que $\mathbf{J}_p$ é uma matriz $3 \times n$ relativa à contribuição das velocidades $\dot{\mathbf{q}}$ das juntas para a velocidade tangencial $\dot{\mathbf{p}}$ da garra, enquanto $\mathbf{J}_o$ é uma matriz $3 \times n$ relativa à contribuição das velocidades $\dot{\mathbf{q}}$ para a velocidade angular $\mathbf{w}$ da garra.

Considerando a equação 2.7, o vetor de velocidades das juntas pode ser obtido pela inversão da matriz jacobiana

$$\dot{\mathbf{q}} = \mathbf{J}^{-1}(\mathbf{q})\dot{\mathbf{p}}. \tag{2.8}$$

Se a configuração inicial $\mathbf{q}(0)$ for conhecida, as posições das juntas podem ser calculadas integrando-se as velocidades no tempo, isto é,

$$\mathbf{q}(t) = \int_0^t \dot{\mathbf{q}}(\zeta)d\zeta + \mathbf{q}(0). \tag{2.9}$$

A integração pode ser realizada de forma discreta por técnicas numéricas. A técnica mais simples é baseada no método de integração de Euler. Dado um intervalo de integração Δt, se as posições e velocidades das juntas num instante t_k são conhecidas, as posições das juntas no instante $t_{k+1} = t_k + \Delta t$ podem ser calculadas como

$$\mathbf{q}(t_{k+1}) = \mathbf{q}(t_k) + \dot{\mathbf{q}}(t_k)\Delta t. \tag{2.10}$$

Esta técnica de inversão matemática da cinemática é independente da capacidade de resolução da estrutura cinemática. Entretanto, é necessário que o jacobiano seja uma matriz quadrada e de posto (*rank*) completo.

Assim, considerações adicionais devem ser feitas no caso de manipuladores redundantes e na ocorrência de singularidades cinemáticas.

2.1.3 Algoritmos Tradicionais de Cinemática Inversa

A solução, cinemática inversa, da equação 2.1 é não linear e, em geral, não possui uma solução analítica na forma fechada. Dessa forma, a cinemática inversa é calculada por técnicas algoritmicas através do uso da cinemática diferencial dada pelas equações 2.8 e 2.10. Para um robô não redundante

com o número mínimo de graus de liberdade requeridos pela tarefa em questão, $m = n$. Entretanto, esta abordagem leva à determinação de velocidades elevadas nas juntas quando o manipulador se aproxima de configurações singulares, isto é, quando $\mathbf{J}(\mathbf{q})$ tende a ficar com posto (*rank*) incompleto.

Uma estratégia clássica para a resolução da cinemática mesmo nas proximidades de singularidades cinemáticas é a técnica dos mínimos quadrados amortecidos (DLS) [36, 37]. O método corresponde à resolução da equação 2.11

$$\dot{\mathbf{q}} = \left(\mathbf{J}^T(\mathbf{q})\mathbf{J}(\mathbf{q}) + \lambda^2\mathbf{I}\right)^{-1}\mathbf{J}^T(\mathbf{q})\mathbf{v}, \tag{2.11}$$

sendo que $\lambda \geq 0$ é o fator de amortecimento e $\mathbf{I}$ é a matrix identidade ($n \times n$). Note que, quando $\lambda = 0$ a equação 2.11 corresponde à equação 2.15. O fator de amortecimento pode ser escolhido de acordo com o seguinte critério

$$\lambda^2 = \begin{cases} 0 & \text{quando } \sigma_n \geq \varepsilon \\ \left(1 - \left(\frac{\sigma_n}{\varepsilon}\right)^2\right)\lambda_{max}^2 & \text{caso contrário} \end{cases}, \tag{2.12}$$

sendo que σ_n é o menor (ou uma estimativa) valor singular de $\mathbf{J}(\mathbf{q})$ e ε define o raio de uma região singular esférica centrada no ponto de singularidade. A variável λ_{max} é usada para modelar adequadamente a solução na vizinhança de uma singularidade.

De modo a reduzir erros devidos à integração numérica, um termo corretivo de realimentação é introduzido pela substituição da velocidade da garra $\mathbf{v}$ por

$$\mathbf{v}_d + \mathbf{K}\mathbf{e}, \tag{2.13}$$

ou simplesmente por $\mathbf{K}\mathbf{e}$, no qual $\mathbf{v}_d$ representa a velocidade desejada para a garra, $\mathbf{K}$ é uma matriz definida positiva - geralmente diagonal - $(n \times n)$, e $\mathbf{e}$ é uma medida do erro entre as poses atual e desejada para a garra do robô. O termo de realimentação é modelado em torno das singularidades usando $\mathbf{K} = \varrho\mathbf{K}_0$, onde $\mathbf{K}_0$ é uma matriz constante e ϱ é um fator variante ajustado como

$$\varrho = \begin{cases} 0 & \text{quando } \sigma_n \leq \varepsilon \\ \frac{(\sigma_n - \varepsilon)^2}{(3\varepsilon)^2} & \text{quando } \varepsilon < \sigma_n < 4\varepsilon \\ 1 & \text{caso contrário} \end{cases}. \tag{2.14}$$

No caso de robôs redundantes, uma vez que $n > m$, a equação 2.6 é sobre-determinada e, portanto, admite um número infinito de soluções. A cinemática inversa passa a ser calculada de acordo

com a equação 2.15.

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}(\mathbf{q})\left(\mathbf{v}_d + \mathbf{K}\mathbf{e}\right), \tag{2.15}$$

sendo que $\mathbf{J}^{\dagger} = \mathbf{J}^T\left(\mathbf{J}\mathbf{J}^T\right)^{-1}$ é a matriz pseudo-inversa. Esta solução minimiza localmente a norma euclidiana das velocidades das juntas.

Uma solução mais geral para a cinemática inversa pode ser determinada a partir da inclusão de uma função-objetivo secundária de acordo com a equação 2.16

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}(\mathbf{q})\left(\mathbf{K}\mathbf{e}\right) + \left(\mathbf{I} - \mathbf{J}^{\dagger}(\mathbf{q})\mathbf{J}(\mathbf{q})\right)\dot{\mathbf{q}}_0, \tag{2.16}$$

sendo que a matriz $\left(\mathbf{I} - \mathbf{J}^{\dagger}(\mathbf{q})\mathbf{J}(\mathbf{q})\right)$ é um projetor do vetor de velocidades de juntas $\dot{\mathbf{q}}_0$ no espaço nulo, $\mathcal{N}\left(\mathbf{J}\right)$, da matriz jacobiana. Uma escolha típica do vetor q_0 é dada pela equação 2.17

$$\dot{\mathbf{q}}_0 = \alpha\left(\frac{\partial w(\mathbf{q})}{\partial \mathbf{q}}\right)^T, \tag{2.17}$$

sendo que $\alpha > 0$ é um escalar, $w(\mathbf{q})$ é uma função-objetivo escalar das variáveis de junta e $\left(\partial w(\mathbf{q})/\partial \mathbf{q}\right)^T$ é uma função vetorial que representa o gradiente de w. Dessa forma w é localmente minimizada. Funções-objetivo secundárias usuais são: índice de manipulabilidade, distância dos limites de junta, distância de obstáculos etc.

Uma forma computacionalmente mais simples é obtida pela relação entre $\mathbf{e}$ e $\dot{\mathbf{q}}$ descrita pela equação 2.18,

$$\dot{\mathbf{q}} = \mathbf{J}^T(\mathbf{q})\mathbf{K}\mathbf{e}. \tag{2.18}$$

Está formulação, embora não-linear e mais simples de resolver computacionalmente que o método da pseudo-inversa, não garante que $\mathbf{e} \to \mathbf{0}$ à medida que $tempo \to \infty$ para trajetórias cartesianas que variem com o tempo [1].

2.2 Informação Visual Para Controle de Robôs

Um fator importante para sistemas de controle servo visual é a modelagem da câmera, pois dele e de seus respectivos parâmetros é que são desenvolvidas a maioria das técnicas existentes. Sendo assim, o modelo de câmera obtido a partir de projeção perspectiva é apresentado a seguir.

Considere uma câmera e seu sistema de coordenadas, $\mathcal{F}_c$, cujo eixo z está alinhado com o eixo

óptico. Considere também um ponto $3D$, $\mathbf{p} = [X, Y, Z, 1]^T$, descrito em função do sistema de coordenadas inercial $\mathcal{F}_0$. O ponto $\mathbf{p}$ pode ser representado em relação ao sistema $\mathcal{F}_c$ de acordo com a equação 2.19

$$\mathbf{p}_c = {}^c\mathbf{T}_0\mathbf{p} \tag{2.19}$$

sendo que

$${}^c\mathbf{T}_0 = \begin{bmatrix} {}^c\mathbf{R}_0 & {}^c\mathbf{t}_0 \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.20}$$

é a transformação homogênea que leva do sistema de coordenadas inercial, $\mathcal{F}_0$, para o sistema de coordenadas $\mathcal{F}_c$ associado à câmera e $\mathbf{p}_c = [X_c, Y_c, Z_c, 1]^T$ é o ponto $\mathbf{p}$ representado no sistema de coordenadas da câmera. A transformação descrita na equação 2.20 representa os parâmetros extrínsicos da câmera.

Usando o modelo de projeção perspectiva ilustrado na figura 2.1 e considerando inicialmente, sem perda de generalidade, uma distância focal de $1\ m$, $\mathbf{p}_c$ é projetado no ponto $\mathbf{m} = [x, y, 1]^T$ do plano Π de imagem de acordo com a equação 2.21,

$$\mathbf{m} = \frac{1}{Z_c}\begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \end{bmatrix}\mathbf{p}_c \tag{2.21}$$

na qual $\mathbf{I}_3$ é uma matriz identidade 3×3. É importante observar que a equação 2.21 implica na perda da informação de profundidade, Z_c.

Na equação 2.21, o ponto $\mathbf{m}$ está em metros e pode ser transformado para coordenadas do plano de tela $\mathbf{m}' = [u, v, 1]^T$, medidas em *pixels*, segundo a equação 2.22,

$$\mathbf{m}' = \mathbf{C}\mathbf{m} \tag{2.22}$$

sendo que

$$\mathbf{C} = \begin{bmatrix} \lambda k_u & -\lambda k_u \cot\phi & u_c \\ 0 & \lambda k_v / \operatorname{sen}\phi & v_c \\ 0 & 0 & 1 \end{bmatrix} \tag{2.23}$$

é a matriz que representa os parâmetros intrínsecos da câmera, na qual λ é a distância focal, k_u e k_v são fatores de escala medidos em $pixels/m$, u_c e v_c são as coordenadas do ponto principal da imagem,

e ϕ é o ângulo entre os eixos u e v. O ponto principal da câmera é aquele no qual o eixo óptico cruza o plano da imagem. Este ponto é, em geral, diferente da origem do sistema de coordenadas de tela.

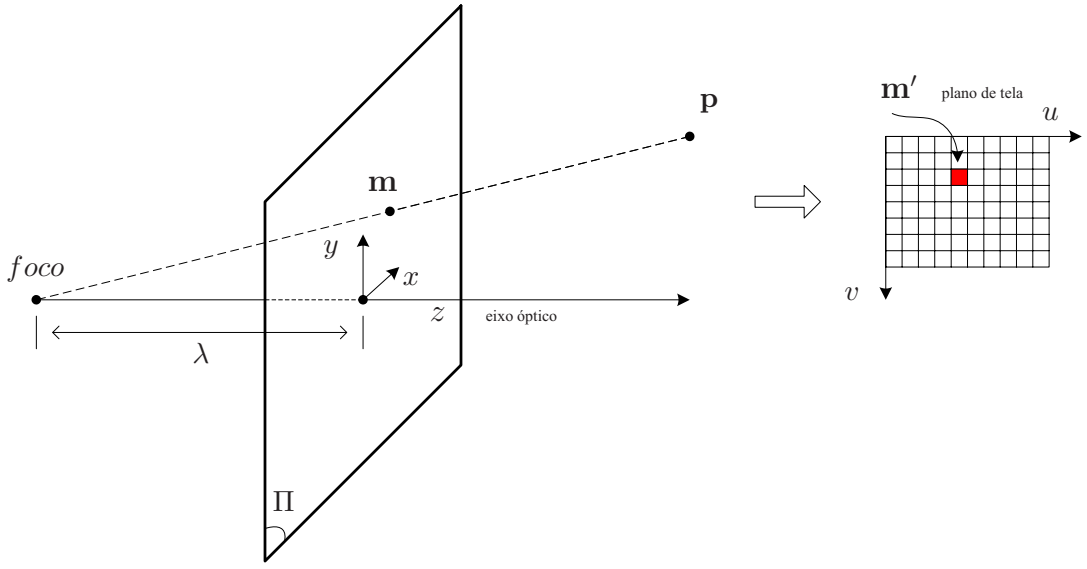


Fig. 2.1: Modelo de câmera

Os parâmetros intrínsecos e extrínsecos da câmera são determidados precisamente por técnicas de calibração [65, 66]. Estes processos são, em geral, realizados *off-line* por serem complexos e necessitarem de um tempo relativamente longo para serem realizados [67, 16].

2.3 Estratégias de Controle Servo Visual

A utilização de visão por computador para controlar um robô é denominada na literatura como controle servo visual (*visual servoing*) [68]. A inclusão da visão em sistemas robóticos tem como principal objetivo o aumento da flexibilidade e precisão destes sistemas, sendo que um dos primeiros trabalhos a utilizar a visão para controle de robôs foi apresentado por Shirai e Inoue [57]. O papel da visão por computador é o de fornecer ao robô o estado do seu ambiente de trabalho, para que esta informação possa ser utilizada no seu controle. No caso de robôs manipuladores, a classe de robôs considerada no escopo do trabalho, no seu ambiente de trabalho encontra-se um, ou vários, objeto(s) a manipular. A informação sobre o ambiente de trabalho do robô obtida através da imagem, informação visual, pode ser utilizada para controle de duas formas distintas [23]. A primeira abordagem é denominada na literatura como controle em malha aberta (*Open-Loop-Robot-Control*) e tem por base a separação entre a parte relacionada com a obtenção e processamento da informação visual e o controle do robô. A designação de controle em malha aberta surge devido à não utilização de informação visual durante o controle, pois o laço de realimentação é fechado com coordenadas de junta

ou cartesianas.

Pode-se afirmar que segundo esta abordagem a posição inicial e desejada do robô é determinada *off-line* através da informação visual, sendo que o movimento do robô é feito “às cegas", pois o controle é unicamente realizado com base nos sensores de junta. As posições iniciais e finais são determinadas através de algoritmos de estimação da pose, sendo para tal necessário calibrar a câmera e ter o modelo do objeto.

A segunda abordagem, a qual se convencionou chamar controle servo visual (*visual servoing*), utiliza direta ou indiretamente na lei de controle a informação visual. Neste caso, é usualmente utilizado um controle interno do robô, com realimentação das variáveis de junta para estabilizar o robô em torno da ação de comando vinda da malha externa de visão. O emprego direto de informação visual implica ter como variáveis controladas as coordenadas 2D do plano da imagem, enquanto a utilização indireta de informação visual implica na extração de características relevantes do objeto a partir das coordenadas 2D do plano da imagem. Como exemplo desta última abordagem tem-se a pose do objeto relativamente à câmera. As aplicações típicas de controle visual de robôs podem ser classificadas como:

1. Posicionamento do robô ou do seu elemento terminal relativamente a um determinado objeto;

2. Rastreamento de um objeto em movimento, mantendo uma relação constante entre este e o robô.

O objeto em atenção é parte fundamental do processo, pois é relativamente a este e a forma como é caracterizado, através da imagem, que a formulação do problema é obtida. É portanto fundamental obter informação visual que o caracterize, sendo esta utilizada para medir o erro entre a posição atual do robô e a sua posição desejada. Esta informação visual, pode ser obtida por uma ou mais câmeras em que esta(s) pode(m) estar colocada(s) no espaço (visualizando o objeto e o elemento terminal do robô) ou no elemento terminal (visualizando o objeto).

As estratégias de controle servo visual são classificadas classicamente em função do tipo de informação usada no cálculo do erro usado na malha de controle [58, 69]. Dessa forma, dependendo de qual parâmetro o sinal de erro está relacionado, pode-se dividi-las em:

1. Controle servo visual baseado na posição (PBVS), no qual o erro é definido em coordenadas de pose no espaço de trabalho;

2. Controle servo visual baseado na imagem (IBVS), no qual o sinal de erro é bidimensional e definido diretamente de características da imagem;

3. Controle servo visual híbrido (HVS), no qual o sinal de erro é definido parte no espaço de trabalho e parte diretamente nas características da imagem.

Uma ou mais câmeras podem ser usadas em configurações diferentes e podem ser empregadas para fornecer a informação visual para o sistema de controle. De acordo com o local de fixação da câmera, a configuração será classificada como:

1. *Eye-in-hand*: quando a(s) câmera(s) está(ão) fixada(s) no elemento terminal do robô;

2. *Eye-to-hand*: quando a(s) câmera(s) está(ão) fixada(s) externamente ao robô;

2.3.1 Controle Servo Visual baseado na Posição

Nesta estratégia de controle servo visual, o sinal de erro usado no sistema de controle é calculado nas coordenadas do espaço de trabalho do robô, $SE(3)$, [70, 71, 72, 73, 25]. Uma vez que se tenha armazenado a pose desejada $\mathbf{s}_d = [x_d \;\; y_d \;\; z_d \;\; \upsilon_{x_d} \;\; \upsilon_{y_d} \;\; \upsilon_{z_d}]^T$, em que a orientação é representada por ângulos *roll-pitch-yaw* ($\upsilon_{x_d} \;\; \upsilon_{y_d} \;\; \upsilon_{z_d}$) [1] [15, 1], e se tenha estimado a pose atual do robô [75, 76, 77], uma lei de controle proporcional pode ser definida como

$$\dot{\mathbf{s}} = \mathbf{\Gamma}(\mathbf{s}_d - \mathbf{s}), \tag{2.24}$$

sendo que $\mathbf{\Gamma}$ é uma matriz de ganhos.

Dessa forma, é necessário que se obtenha a pose do objeto e/ou robô indiretamente através da informação visual, o que, em geral, demanda muita capacidade computacional e requer conhecimento preciso dos parâmetros da(s) câmera(s) usada(s) no sistema. Além disso, é necessário o conhecimento do modelo $3D$ do(s) objeto(s) em questão. Alguns desafios ainda enfrentados pelas estratégias baseadas em PBVS estão listados a seguir.

- necessidade de reconstrução $3D$ do objeto;
- conhecimento do modelo $3D$ do objeto;
- muito suscetível a ruídos;
- dependência de calibração precisa da(s) câmera(s);
- modelagem precisa do robô;
- alta capacidade computacional para atender critérios de tempo real;
- dificuldade de manter o(s) objeto(s) de interesse dentro do campo de visão da câmera, uma vez que o controle é feito no espaço cartesiano e não diretamente na imagem.

[1] É important salientar que, embora a representação *roll-pitch-yaw* esteja sendo usada, qualquer outro tipo de representação para a orientação poderia ser considerada. [74].

Entretanto, uma característica positiva da abordagem PBVS é que ela tende a gerar trajetórias “adequadas" para o robô [74]. Isto quer dizer que, sob PBVS, as trajetórias cartesianas impostas ao robô requerem uma menor quantidade de movimentos desnecessários e longos, o que do ponto de vista prático é uma característica atrativa.

2.3.2 Controle Servo Visual Baseado na Imagem

A idéia dessa abordagem é a de se determinar as velocidades lineares e angulares da câmera a partir das velocidades de determinadas características extraídas diretamente nas imagens [22], tais como: pontos, retas, elipses etc. Sendo assim, se $\mathbf{s} = [X, Y, Z, \upsilon_x, \upsilon_y, \upsilon_z]^T$ representa as coordenadas da pose da câmera em relação ao sistemas de coordenadas inercial e $\dot{\mathbf{s}} = \left[\dot{X}, \dot{Y}, \dot{Z}, \omega_x, \omega_y, \omega_z\right]^T$ suas velocidades correspondentes, se $\mathbf{f} = [u, v]^T$ for o vetor de coordenadas no plano da imagem de um ponto filmado e $\dot{\mathbf{f}} = [\dot{u}, \dot{v}]^T$ suas velocidades, o jacobiano da imagem, também conhecido como matriz de iteração, é dado pela equação 2.25

$$\dot{\mathbf{f}} = \mathbf{J}_i(\mathbf{f}, \mathbf{r})\dot{\mathbf{s}}, \tag{2.25}$$

sendo que

$$\mathbf{J}_i = \begin{bmatrix} \frac{\lambda}{z} & 0 & \frac{-u}{z} & \frac{-uv}{\lambda} & \frac{\lambda^2+u^2}{\lambda} & -v \\ 0 & \frac{\lambda}{z} & \frac{-v}{z} & \frac{-\lambda^2-u^2}{\lambda} & \frac{uv}{\lambda} & u \end{bmatrix} \tag{2.26}$$

é o jacobiano para um ponto da imagem, λ é a distância focal da câmera e z é a distância ao longo do eixo óptico entre o plano de imagem e o ponto no espaço. O jacobiano de imagem mais comum é baseado em pontos da imagem, embora várias outras características também possam ser usadas [22]. É importante notar que $\mathbf{J}_i$ na equação 2.26 forma uma matriz 2×6 e mais pontos podem ser usados para aumentar o posto de $\mathbf{J}_i$. Para que o jacobiano seja quadrado são necessários 3 pontos. Entretanto, sabe-se que a mesma imagem de 3 pontos pode ser vista de 4 poses de câmera diferentes [78]. Sendo assim, teoricamente, pode-se obter uma pose única para uma determinada imagem a partir de, pelo menos, 4 pontos não colineares, o que faz com que $\mathbf{J}_i$ tenha dimensões 8×6.

A forma mais simples de empregar o IBVS é usando a equação 2.25 para construir a lei de controle

$$\dot{\mathbf{r}} = \mathbf{\Gamma}\mathbf{J}_i^{\dagger}(\mathbf{f}, \mathbf{r})\dot{\mathbf{f}} \tag{2.27}$$

sendo que $\mathbf{\Gamma}$ é uma matriz de ganhos e $\mathbf{J}_i^{\dagger}(\mathbf{f}, \mathbf{r})$ é a pseudo-inversa de $\mathbf{J}_i$. Nesse caso, $\dot{\mathbf{f}}$ passa a

representar o erro entre o vetor de características desejadas e o vetor atual de características.

2.3.3 Controle Servo Visual Híbrido

O controle visual baseado na imagem, como visto anteriormente, é robusto a erros na modelagem da cinemática do robô, da câmara e ainda da forma do objeto. Contudo o seu domínio de estabilidade é local [27]. Visando resolver este problema, tem-se proposto novas metodologias que englobam características do controle visual baseado na imagem e do controle baseado em posição [26, 59, 60, 79]. Estas estratégias são denominadas como controle servo visual híbrido (HVS). Uma das primeiras metodologias de controle servo visual híbrido foi proposta por Malis et al. [26], controle visual $2\ 1/2D$, e é baseada na utilização de seis características do objeto na imagem. As duas primeiras dizem respeito a um ponto do objeto na imagem, a terceira refere-se à profundidade z, e as três coordenadas seguintes referem-se à rotação entre a posição atual e desejada da câmara. Na estratégia proposta por Corke e Hutchinson [59], há uma tentativa de se fazer a separação entre a rotação e translação do eixo z em relação aos demais.

2.4 Busca em Espaço de Estados

Definição 2.4.1 (Busca em Espaços de Estados) *Um problema de busca em espaço de estados (SSP) é um conjunto* $(\mathbb{S},\ \mathbb{G},\ s_0,\ \{\mathbb{O}_1\ \ldots\ \mathbb{O}_k\})$*, sendo:*

- $\mathbb{S}$ *é o conjunto de estados;*
- $\mathbb{G} \in \mathbb{S}$ *é o conjunto de estados-objetivo;*
- $s_0 \notin \mathbb{G}$ *é o estado ou condição inicial.*
- $\{\mathbb{O}_1\ \ldots\ \mathbb{O}_k\}$ *é um conjunto de operadores de busca. Alguns operadores de busca podem não ser prontamente aplicáveis em alguns estados. Quando um operador de busca* $\mathbb{O}_j$ *é aplicado a um estado* $s \notin \mathbb{G}$*, tem-se como resultado um novo estado* $Suc_j(s)$ *incorrendo num custo* $c_j(s) \geq 0$*.*

Uma solução para uma SSP é uma seqüência de operadores de busca que, iniciando a partir de s_0, resulta em algum estado pertencendo a $\mathbb{G}$.

A maneira tradicional de resolver problemas de SSP é aplicar técnicas de busca heurística [80, 81]. Nestes procedimentos algorítmicos, o próximo operador é escolhido como aquele com o menor valor de f na equação 2.28,

$$f(s) = g(s) + h(s) \tag{2.28}$$

onde $g(s)$ é a distância do estado inicial e $h(s)$ é uma função heurística que estima a distância atual para o estado-objetivo.

A maioria das técnicas de busca heurística foram desenvolvidas para espaços de estados discretos. Porém, algumas de suas propriedades foram estendidas para espaços de estados contínuos por meio da análise de Lyapunov [82]. Uma função de controle de Lyapunov por ser definida da seguinte forma:

Definição 2.4.2 (Função de Controle de Lyapunov) *Dado um problema de busca em espaço de estados, uma função de controle de Lyapunov (LCF) é uma função* $L : \mathbb{S} \mapsto \mathbb{R}$ *com as seguintes propriedades:*

1. $L(s) \geq 0, \forall\, s \in \mathbb{S}$ *;*
2. *Existe* $\delta > 0$ *tal que para todo* $s \notin \mathbb{G}$ *existe algum operador de busca* $\mathbb{O}_j$ *tal que* $L(s) - L\left(Suc_j(s)\right) \geq \delta$.

A propriedade 1 da definição 2.4.2 impõe que uma LCF seja semi-definida positiva, enquanto a propriedade 2 impõe que uma LCF, sujeita a um operador de busca, seja sempre decrescente. Uma função de controle de Lyapunov é uma boa candidata para uma função heurística devido a sua característica de diminuição monotônica à medida que se aproxima do objetivo.

No caso de buscas em tempo real, geralmente, a otimalidade dos algoritmos é relaxada de forma a melhorar o desempenho temporal. Uma generalização desta situação seria construir um caminho para o objetivo através de buscas com profundidade limitada (*depth-limited search*), selecionando-se o melhor operador de busca a ser aplicado no próximo passo da busca. Korf [83] apresenta um algoritmo de busca em tempo real para espaços de estados discretos, o *real-time* A^* (RTA*). A versão continua para o algoritmo RTA* é a busca repetitiva com profundidade limitada [84]. Nestes casos, a função-custo $g(s)$ possui um significado ligeiramente diferente. Ela agora representa o custo atual do movimento a partir do estado atual em vez do estado onde a busca teve início.

Capítulo 3

Cinemática Inversa via Busca Heurística

Apresenta o desenvolvimento teórico da técnica proposta, trazendo obtenção de expressão e análise de convergência.

3.1 Formulação do Problema

Faça $\mathbf{p}$ e $\mathbf{R}$ representarem respectivamente a posição e a orientação atuais da garra do robô, dadas pela resolução do modelo cinemático direto, equação 2.1, e faça a função escalar L, relacionando o erro de posição $\mathbf{e}_p$ e o erro de orientação $\mathbf{e}_o$, ser

$$L = \mathbf{e}_p{}^{\mathbf{T}}\mathbf{e}_p + \gamma tr\left(\mathbf{e}_o^{\mathbf{T}}\mathbf{e}_o\right) \tag{3.1}$$

sendo que $\gamma > 0$ é um escalar usado como fator de ponderação, $tr(\cdot)$ é a função traço e

$$\mathbf{e}_p = \mathbf{p}_d - \mathbf{p} \tag{3.2}$$

$$\mathbf{e}_o = \mathbf{R}_d - \mathbf{R} \tag{3.3}$$

A função L é usada como uma função heurística de avaliação, h, na equação 2.28 e dá uma medida da distância $d(|\mathbf{T}_d - \mathbf{A}|)$, isto é, a distância entre as poses desejada e atual da garra do robô.

O operador de busca $\mathbb{O}_j$ é definido como uma seqüencia de movimentações individuais das juntas que faz com que a garra do robô se aproxime de uma pose-objetivo no espaço operacional. A ordem destas movimentações pode ser definida por um critério de "melhor primeiro" (*best-first*), no qual a próxima junta a ser movimentada é escolhida entre aquelas que ainda não contribuíram, com a

máxima diminuição de h ou por seqüencias de movimentações predefinidas. Será mostrado na seção 3.2.2 que, uma vez que a trajetória desejada esteja dentro do espaço de trabalho do robô, haverá sempre um operador de busca, isto é, uma seqüencia de movimentações de juntas capaz de diminuir L. Portanto, L, sendo sujeita a aplicação do operador $\mathbb{O}_j$, estabelece uma CLF.

A trajetória, definida em posição e orientação, é discretizada em N pontos com intervalos de tempo Δt. Assim, o conjunto de estados-objetivo $\mathbb{G}$ é

$$\mathbb{G} = \left\{ \mathbf{k}^{-1}(\mathbf{T}_{dk}) \right\}, \;\; k = 1, \ldots, N. \tag{3.4}$$

sendo que $k^{-1}(\mathbf{T}_{dk})$[1] é a solução de cinemática inversa da k-ésima pose da trajetória. O espaço de estados é formado pelas posições das juntas, $\mathbb{S} = \mathbf{q}$, onde $\mathbf{q} = [q_1 \;\; \ldots \;\; q_n]^T$. É importante notar que $\mathbb{G}$ é discreto para robôs não redundantes e contínuo para robôs redundantes, enquanto que $\mathbb{S}$ é contínuo para ambos os casos, uma vez que para uma determinada pose do robô, há infinitas configurações de juntas que podem resolver o equacionamento matemático (cinemática inversa).

3.2 Expressões Gerais e Interpretação Geométrica

3.2.1 Expressões Gerais para Posição e Orientação

A contribuição da i-ésima junta para a solução do problema pode ser formulada usando a representação de Denavit-Hartenberg, equação 2.1, incluindo-se uma junta virtual i' como

$${}^0\mathbf{T}_n = \underbrace{{}^0\mathbf{A}_1 \ldots {}^{i-2}\mathbf{A}_{i-1}}_{{}^0\mathbf{A}_{i-1}} A'_i \underbrace{{}^{i-1}\mathbf{A}_i \ldots {}^{n-1}\mathbf{A}_n}_{{}^{i-1}\mathbf{A}_n} \tag{3.5}$$

na qual

$$\mathbf{A}'_i = \begin{bmatrix} \mathbf{R}_{z_i}(\Delta\theta_i) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{3.6}$$

para juntas rotacionais, sendo que $\mathbf{R}_{z_i}(\Delta\theta_i)$ representa uma rotação de $\Delta\theta$ em torno do i-ésimo eixo z. No caso de juntas prismáticas

$$\mathbf{A}'_i = \begin{bmatrix} \mathbf{I} & \mathbf{p}_{z_i}(\Delta d_i) \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{3.7}$$

[1] De modo a simplificar a notação, o sub-índice k será omitido de agora em diante, mas sempre que houver uma referência a $\mathbf{T}_d$, isto significa uma referência à k-ésima pose exigida pela trajetória.

sendo que $\mathbf{p}_{z_i}(\Delta d_i) = [0\ 0\ \Delta d_i]^T$ uma translação de Δd_i na direção do eixo z_i.

Sendo assim, para juntas rotacionais, a posição e a orientação da garra do robô são dadas pelas equações 3.8 e 3.9 respectivamente

$$^0\mathbf{p}_n = \mathbf{p} = {}^0\mathbf{R}_{i-1}\mathbf{R}_{z_i}(\Delta\theta_i)^{i-1}\mathbf{p}_n + {}^0\mathbf{p}_{i-1}. \tag{3.8}$$

$$^0\mathbf{R}_n = \mathbf{R} = {}^0\mathbf{R}_{i-1}\mathbf{R}_z(\Delta\theta_i)^{i-1}\mathbf{R}_n \tag{3.9}$$

e no caso prismático, tem-se

$$^0\mathbf{p}_n = {}^0\mathbf{R}_{i-1}{}^{i-1}\mathbf{p}_n + {}^0\mathbf{R}_{i-1}\mathbf{p}_{z_i}(\Delta d_i) + {}^0\mathbf{p}_{i-1}. \tag{3.10}$$

lembrando que a orientação da garra do robô não é alterada no caso de juntas prismáticas.

Para $\Delta\theta_i = \Delta d_i = 0$, $^{i-1}\mathbf{p}_n$ e $^{i-1}\mathbf{R}_n$ são facilmente calculados a partir das equações 3.8 e 3.9 como

$$^{i-1}\mathbf{p}_n = {}^0\mathbf{R}_{i-1}^T\left({}^0\mathbf{p}_n - {}^0\mathbf{p}_{i-1}\right). \tag{3.11}$$

$$^{i-1}\mathbf{R}_n = {}^0\mathbf{R}_{i-1}^T{}^0\mathbf{R}_n \tag{3.12}$$

Substituindo as equações 3.8 e 3.9 nas equações 3.2, 3.3 e 3.1, depois de algumas manipulações matemáticas e levando em consideração a ortogonalidade das matrizes de rotação, $\mathbf{R}^T\mathbf{R} = \mathbf{I}$, para o caso de juntas rotacionais, chega-se a

$$h_i = kp_{0,i} + \gamma ko_{0,i} + (kp_{1,i} + \gamma ko_{1,i})\ \mathrm{sen}\,(\Delta\theta_i) + (kp_{2,i} + \gamma ko_{2,i})\ \cos(\Delta\theta_i) \tag{3.13}$$

sendo que $kp_{0,i}$, $kp_{1,i}$ e $kp_{2,i}$ são constantes relacionadas com o problema de posicionamento e dadas por

$$\begin{aligned} kp_{0,i} &= b_z c_z - 2\mathbf{p}_d^{T\,0}\mathbf{p}_{i-1} + {}^0\mathbf{p}_{i-1}^T\,{}^0\mathbf{p}_{i-1} + {}^{i-1}\mathbf{p}_n^{T\,i-1}\mathbf{p}_n + \mathbf{p}_d^T\mathbf{p}_d \\ kp_{1,i} &= b_y c_x - b_x c_y \\ kp_{2,i} &= b_x c_x + b_y c_y \end{aligned} \tag{3.14}$$

enquanto que $ko_{0,i}$, $ko_{1,i}$ e $ko_{2,i}$ são constantes relacionadas com o problema de orientação e dadas pela equação 3.15,

$$\begin{aligned} ko_{0,i} &= -2(ra_{13}rb_{31} + ra_{33}rb_{33} + ra_{23}rb_{32}) + 6 \\ ko_{1,i} &= 2\left(rb_{21}ra_{11} + rb_{22}ra_{21} - rb_{12}ra_{22} - rb_{13}ra_{32} + rb_{23}ra_{31} - rb_{11}ra_{12}\right) \\ ko_{2,i} &= -2\left(rb_{13}ra_{31} + rb_{21}ra_{12} + rb_{11}ra_{11} + rb_{22}ra_{22} + rb_{12}ra_{21} + rb_{23}ra_{32}\right) \end{aligned} \tag{3.15}$$

com

$$[b_x\ b_y\ b_z]^T = 2^0\mathbf{R}_{i-1}^T\left({}^0\mathbf{p}_{i-1} - \mathbf{p}_d\right) \tag{3.16}$$

$$[c_x\ c_y\ c_z]^T = {}^i\mathbf{p}_n \tag{3.17}$$

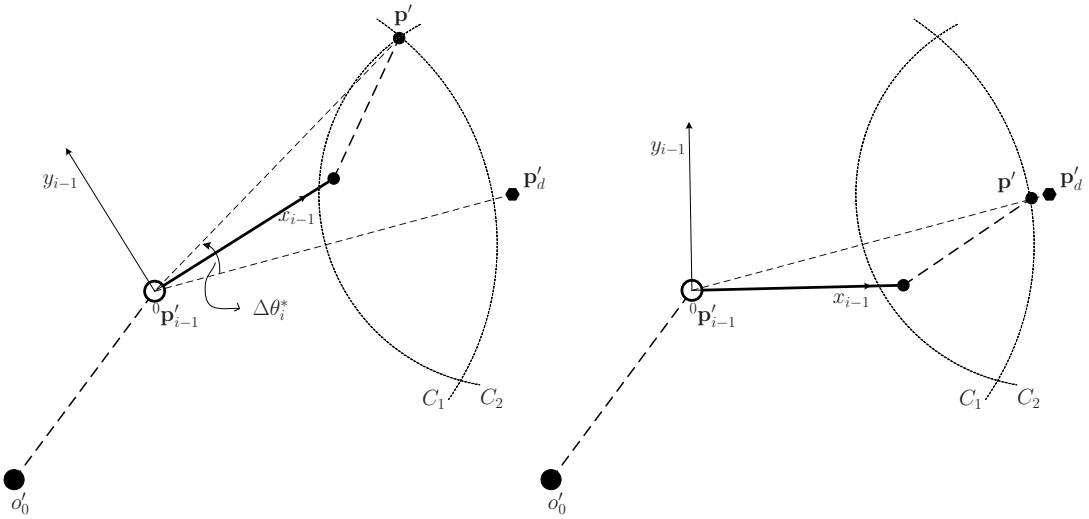
$$\begin{bmatrix} ra_{11} & ra_{12} & ra_{13} \\ ra_{21} & ra_{22} & ra_{23} \\ ra_{31} & ra_{32} & ra_{33} \end{bmatrix} = \mathbf{R}_d^{T\,0}\mathbf{R}_{i-1} \tag{3.18}$$

$$\begin{bmatrix} rb_{11} & rb_{12} & rb_{13} \\ rb_{21} & rb_{22} & rb_{23} \\ rb_{31} & rb_{32} & rb_{33} \end{bmatrix} = {}^{i-1}\mathbf{R}_n \tag{3.19}$$

No intervalo $[-\pi, \pi]$ existem apenas dois pontos extremos para h_i, e eles estão deslocados $\pi\ rad$ entre si. Portanto, o mínimo de h_i é facilmente encontrado a partir das equações 3.25 e 3.27.

$$\Delta\theta_i^* = \arctan\left(\frac{kp_{1,i} + \gamma ko_{1,i}}{kp_{2,i} + \gamma ko_{2,i}}\right) \tag{3.20}$$

Da mesma forma, resolvendo para juntas prismáticas e substituindo a equação 3.10 nas equações 3.2,



(a) Initial (b) Final

Fig. 3.1: Geometria da contribuição da i-ésima junta.

e 3.1, chega-se a

$$h = {\Delta_{d_i}}^2 + kp_{1,i}\Delta_{d_i} + k_{2,i} \tag{3.21}$$

A equação 3.21 possui apenas um mínimo dado por

$$\Delta d_i^* = -\frac{kp_{1,i}}{2} \tag{3.22}$$

3.2.2 Interpretação Geométrica Para o Caso do Posicionamento

De modo a mostrar a funcionalidade do método de busca heurística e suprir maiores detalhes para um melhor entendimento do seu processo de convergência, pode-se usar uma interpretação geométrica para o problema do rastreamento de trajetórias em posição apenas.

Considere $\mathbf{p}$ como um ponto que representa a posição da garra do robô e $\mathbf{p}_d$ como o próximo ponto da a ser alcançado na trajetória. Considere também, sem perda de generalidade, que todas as juntas são rotacionais. A figura 3.1a mostra as projeções $\mathbf{p}'_d$ e $\mathbf{p}'$ dos pontos $\mathbf{p}_d$ e $\mathbf{p}$ no plano ortogonal ao eixo de rotação da i-ésima junta (z_i), enquanto que a figura 3.1b mostra a configuração assumida pelo robô depois de uma iteração do algoritmo de busca na qual a i-ésima junta sofreu uma rotação de $\Delta\theta_i^*\ rad$. O processo de determinação de valor de $\Delta\theta_i^*$ pode ser interpretado geometricamente conforme descrição a seguir.

C_2 é uma circunferência centrada em $\mathbf{p}'_d$ e raio $|\mathbf{p}'_d - \mathbf{p}|$. C_1 é uma circunferência centrada em $\mathbf{p}_i$, definida no plano $x_i y_i$ e com raio $|\mathbf{p}' - \mathbf{p}_{i-1}|$. Assim, estas circunferências sempre se interceptam

em $\mathbf{p}'$. No caso dessas circunferências se interceptarem somente em $\mathbf{p}'$, qualquer movimentação na i-ésima junta irá aumentar $|\mathbf{p}' - \mathbf{p}'_d|$ e, conseqüentemente, o valor de h. Isto ocorre quando $\mathbf{p}'_d$ se situa no eixo x_i, exceto em ${}^0\mathbf{p}_{i-1}$. Quando $\mathbf{p}'_d$ coincide com ${}^0\mathbf{p}_{i-1}$, qualquer movimentação na i-ésima junta não alterará $|\mathbf{p}' - \mathbf{p}'_d|$. Caso C_1 e C_2 sejam secantes, um deslocamento $\Delta\theta_i^*$ pode ser determinado de forma que os vetores $|\mathbf{p}'_d - {}^0\mathbf{p}_{i-1}|$ e $|\mathbf{p}' - {}^0\mathbf{p}_{i-1}|$ fiquem alinhados, $\mathbf{p}' = \mathbf{p}^*$, o que significa a distância $d(|\mathbf{p}_d - \mathbf{p}|)$ mínima que pode ser alcançada movendo-se somente a junta i (veja a figura 3.1b). Note que $k_{1,i}$ and $k_{2,i}$ representam, respectivamente, os termos $seno$ e $cosseno$ de $[c_x\ c_y\ c_z]\,\mathbf{R}_{z_i}(\Delta\theta_i)\,[b_x\ b_y\ b_z]^T$.

Com as considerações anteriores, é possível notar que, dada uma seqüência de juntas a serem movimentadas, a busca sempre irá convergir desde que o ponto desejado na trajetória esteja sob alcance do robô, ou seja, dentro de seu espaço de trabalho. A única situação singular irá ocorrer quando o robô estiver completamente esticado, correspondendo a uma posição na fronteira do espaço de trabalho, e o próximo ponto da trajetória estiver situado no eixo definido pela estrutura do robô. Entretanto, esta situação pode ser facilmente evitada pela simulação de um pequeno distúrbio na configuração do robô para uma posição mais adequada antes do rastreamento do referido ponto [85].

3.3 Algoritmo Básico

A idéia básica para a resolução da cinemática inversa via busca em espaços de estados é a de elaborar e aplicar um algoritmo de busca heurística que seja capaz de determinar a melhor seqüência de juntas a serem movimentadas. Neste sentido, o processo de resolução da cinemática inversa pode ser elaborado como a seguir.

Defina a seguinte função para $i = 1,\ \ldots,\ n$,

$$f_i(k) = h_i(k) \tag{3.23}$$

sendo que h é definida como na equação 3.25. Defina também os conjuntos: $\textit{FECHADO}$ como o conjunto de juntas que já foram selecionadas para contribuir para a solução da cinemática inversa; e $\textit{ABERTO}$ como o conjunto de juntas que ainda não foram selecionadas. Inicialmente, $\textit{ABERTO}$ contém todas as juntas e $\textit{FECHADO}$ é vazio. Depois de n iterações, $\textit{ABERTO}$ é vazio e $\textit{FECHADO}$ contém todas as juntas. De acordo com estas definições, o método de busca proposto pode ser descrito pelo algoritmo 1.

Inicialmente, os conjuntos $\textit{ABERTO}$ e $\textit{FECHADO}$ são iniciados. Então, a k-ésima pose exigida pela trajetória é amostrada e o critério de parada (cp)[2] é calculado. Se cp for maior que uma tolerância, tol,

[2]É importante salientar que o critério de parada pode ser definido independentemente do processo de busca, ou seja,

predefinida, o algoritmo calcula $\Delta\theta_i^*$ para todas as juntas, seguido pelo cálculo de f_i, passos 7 e 8. A seguir, a próxima junta a contribuir é selecionada, passo 10. Esta junta, j, é escolhida como sendo aquela que possui o menor valor de f_i entre as juntas que ainda não contribuíram na iteração atual , isto é, $\forall i \in ABERTO$. Uma vez que a junta j tenha sido selecionada, $\tilde{\theta}_j$ correspondente é atualizado de acordo com o passo 11. Então, o algoritmo atualiza os conjuntos $ABERTO$ e $FECHADO$. A junta j é movida do conjunto $ABERTO$ para o $FECHADO$, significando que ela não será mais usada até que todas as juntas que ainda estiverem em $ABERTO$ tiverem dado suas respectivas contribuições. Quando isto ocorre, todas as juntas são colocadas novamente no conjunto $ABERTO$ como no passo 1. Este processo é repetido até que $cp \leq tol$, quando a melhor configuração obtida para o robô é atualizada , passo 15.

Algoritmo 1 Algoritmo básico do método de busca heurística.

1: $ABERTO \Leftarrow \{1, \ldots, n\}$; $FECHADO \Leftarrow \emptyset$;
2: Amostre a k-ésima T_d;
3: $\tilde{\theta} \Leftarrow \theta$;
4: Calcule cp;
5: **while** $sc > tol$ **do**
6: **for** $i \Leftarrow 1$ to n **do**
7: Calcule $\Delta\theta_i$ que minimiza h_i;
8: Calcule f_i como na equação 3.23;
9: **end for**
10: Selecione a junta $j \mid f_j = min\{f_i\}, \forall i \in ABERTO$;
11: $\tilde{\theta}_j \Leftarrow \tilde{\theta}_j + \Delta\theta_j$;
12: Atualize $ABERTO$ e $FECHADO$;
13: Calcule cp;
14: **end while**
15: $\theta \Leftarrow \tilde{\theta}$;
16: $k \Leftarrow k + 1$;
17: Volte ao passo 1;

3.4 Cinemática Inversa Calculada com um Processo Recursivo

Uma vez que o objetivo principal deste trabalho é o de desenvolver uma técnica simples para a resolução da cinemática inversa de robôs manipuladores de cadeia serial que possa ser facilmente implementada, inclusive considerando o uso de dispositivos lógicos programáveis, é interessante examinar a possibilidade de se modificar o algoritmo 1 de modo a torná-lo mais simples e regular.

Neste sentido, para $\Delta\theta_i$ pequeno o bastante, $R_{z,i}(\Delta\theta_i)$ pode ser representada de acordo com a equação 3.24,

pode ser calculado inclusive usando sensores externos ao robô como, por exemplo, câmeras de vídeo.

$$\mathbf{R}_{z,i}(\Delta\theta_i) = \begin{bmatrix} 1 & -\Delta\theta_i & 0 \\ \Delta\theta_i & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.24}$$

Levando em consideração o caso do posicionamento e substituindo a equação 3.8, com $\mathbf{R}_{z,i}(\Delta\theta_i)$ dada pela equação 3.24, nas equações 3.2 e 3.1, chega-se a

$$h_i = k_{0,i}{\Delta\theta_i}^2 + k_{1,i}\Delta\theta_i + k_{2,i} \tag{3.25}$$

que é uma equação do segundo grau, cujas constantes $k_{0,i}$, $k_{1,i}$ e $k_{2,i}$ são dadas por

$$\begin{aligned} k_{0,i} &= c_x^2 + c_y^2 \\ k_{1,i} &= b_y c_x - b_x c_y \\ k_{2,i} &= -2\mathbf{p}_d^T {}^0\mathbf{p}_{i-1} + {}^0\mathbf{p}_{i-1}^T {}^0\mathbf{p}_{i-1} + \mathbf{p}_d^T\mathbf{p}_d + [b_x\ b_y\ b_z]\,[c_x\ c_y\ c_z]^T + \\ &\quad [c_x\ c_y\ c_z]\,[c_x\ c_y\ c_z]^T \end{aligned} \tag{3.26}$$

Pode-se observar da equação 3.26 que $k_{0,i} \geq 0$. Portanto, quando $k_{0,i} > 0$ o mínimo de h_i é facilmente calculado a partir da equação 3.25 como

$$\Delta\theta_i^* = \frac{-k_{1,i}}{2k_{0,i}} \tag{3.27}$$

Sendo que, quando $k_{0,i} = 0$, a condição de contorno é $\Delta\theta_i^* = 0$.

Diferentemente do algoritmo 1, a idéia agora é a de retirar o processo de escolha da seqüência de juntas a serem movimentadas. Neste caso, o operador de busca é mantido como uma seqüência predefinida. Com o uso desta seqüência, a estrutura do método é simplificada, assim como a complexidade computacional, uma vez que não é mais necessário calcular as constantes da busca para mais que uma junta a cada iteração do algoritmo. Este procedimento transforma o algoritmo de busca num algoritmo recursivo mais simples, embora implique numa busca menos agressiva pela solução da cinemática inversa.

O deslocamento de cada junta na seqüência é determinado pela equação 3.27. Este deslocamento é então adicionado a sua posição de junta correspondente, $\theta_{i,j} = \theta_{i,j-1} + \Delta\theta_i^*$, onde o sub-índice j representa a j-ésima iteração do algoritmo. Os deslocamentos das juntas são também acumulados, $\Delta\theta_{i,j} = \Delta\theta_{i,j-1} + \Delta\theta_i^*$, de maneira a determinar o deslocamento total durante o intervalo de tempo Δt. Note que $\Delta\theta_{i,0} = 0$ e $\theta_{i,0}$ representa a configuração atual do robô. Se a i-ésima contribuição for $\Delta\theta_i^* \neq 0$, a garra irá se aproximar da posição objetivo.

Se as variáveis de junta $\theta_{i,j}$ ou $\dot{\theta}_{i,j} = \Delta\theta_{i,j}/\Delta t$ violarem seus respectivos limites definidos pela

equação 3.28, $\Delta\theta_i$ é recalculado de modo a fazer com que os limites não sejam mais violados.

$$\begin{aligned} \theta_{i,min} \leq \theta_{i,j} \leq \theta_{i,max} \\ \dot{\theta}_{i,min} \leq \dot{\theta}_{i,j} \leq \dot{\theta}_{i,max} \end{aligned} \tag{3.28}$$

De modo a evitar paradas repentinas das juntas quando as mesmas atingirem seus limites de posição, uma função de penalização de velocidades, equação 3.29, é usada para desacelerar o movimento das juntas à medida que elas se aproximam dos limites de posição. Assim, um novo deslocamento é determinado de acordo com a equação 3.30. Uma exemplo da forma da função de penalização de velocidade é mostrada na figura 3.2, na qual o valor máximo absoluto para o limite da junta foi escolhido, a título de exemplicação, como sendo de $1\ rad$.

$$w_i = \frac{4\left(\theta_{i,max} - \theta_{i,j}\right)\left(\theta_{i,j} - \theta_{i,min}\right)}{\left(\theta_{i,max} - \theta_{i,min}\right)^2} \tag{3.29}$$

e

$$\Delta\theta'_i = w_i \Delta\theta_i \tag{3.30}$$

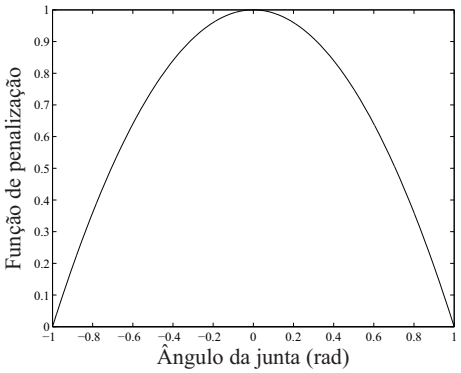


Fig. 3.2: Função de penalização de velocidades.

A seqüência de juntas é repetida até que nenhuma contribuição adicional possa ser dada, isto é, $\Delta\theta_i^* = 0$ para todo i. Isto significa que o ponto da trajetória foi alcançado ou, contrariamente, que o mesmo está fora do alcance do robô. Para verificar esta condição, a função $\sqrt{h_i}$ é calculada, se este valor for menor que uma tolerância predefinida, o ponto desejado é considerado como atingido. Então, o novo vetor de referência para as posições das juntas θ_j é enviado para os controladores das juntas e o próximo ponto da trajetória é amostrado. Este processo é repetido até que o último ponto da trajetória tenha sido rastreado.

3.5 Inclusão de Informações de Medidas Externas na Formulação do Processo de Busca

Faça $\mathbf{s}_n$ e $\mathbf{s}_d$ representarem respectivamente as poses atual e desejada para a garra do robô definidas no $SE(3)$. Assim, o objetivo primário da busca é fazer com que o robô alcance o ponto desejado, isto é que $\mathbf{s}_n \rightarrow \mathbf{s}_d$ ao longo do tempo. Porém, caso existam variações nos parâmetros cinemáticos do robô, a representação da garra do robô em relação ao sistema de coordenadas inercial, dada pela resolução do modelo cinemático inverso, será $\tilde{\mathbf{s}}_n$, veja figura 3.3. Tal situação implica em erros no processo da resolução da cinemática inversa, pois o processo de busca vai procurar eliminar a diferença $\mathbf{s}_d - \tilde{\mathbf{s}}_n$ e não a diferença real $\mathbf{s}_d - \mathbf{s}_n$.

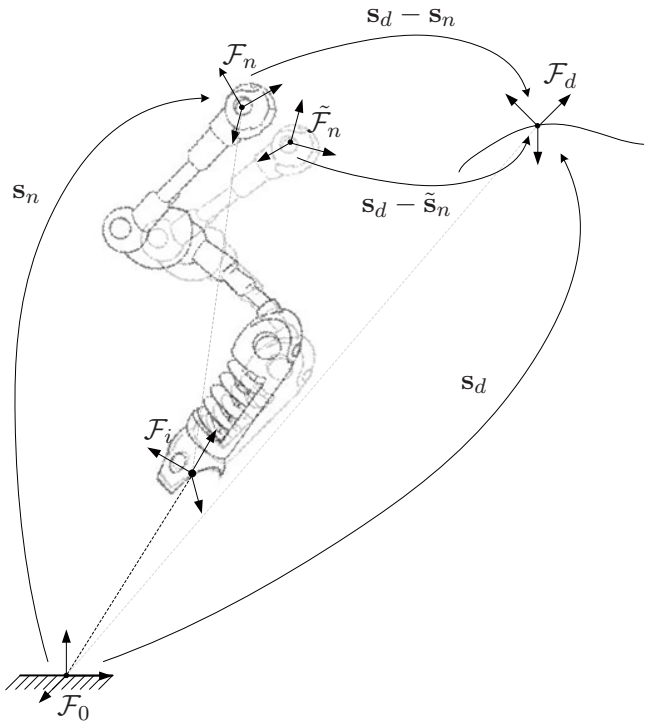


Fig. 3.3: Configurações de um robô usando parâmetros cinemáticos nominais e reais.

No sentido de reduzir esse efeito, suponha agora que se tenha algum tipo de sensor que determine/estime $\mathbf{s}_n$ e $\mathbf{s}_d$ ou a diferença entre ambos, independentemente dos parâmetros do robô, com as respectivas coordenadas dadas por $\mathbf{r}_n$ e $\mathbf{r}_d$, e redefina a referência para a busca segundo a equação 3.31

$$\mathbf{s}_d^* = \mathbf{g}(\mathbf{r}_d - \mathbf{r}_n) + \tilde{\mathbf{s}}_n, \tag{3.31}$$

sabendo-se que $\mathbf{g}(:)$ representa o mapeamento das coordenadas do sensor para as coordenadas $SE(3)$, ou seja, o modelo inverso do sensor. Tal redefinição é exemplificada na forma de diagrama de blocos

da figura 3.4, no qual o sensor externo é uma câmera de vídeo. É importante observar no diagrama de blocos da figura 3.4 que uma malha com realimemtação positiva (gera s_d^*) foi adicionada as malhas de controle de junta e servo visual.

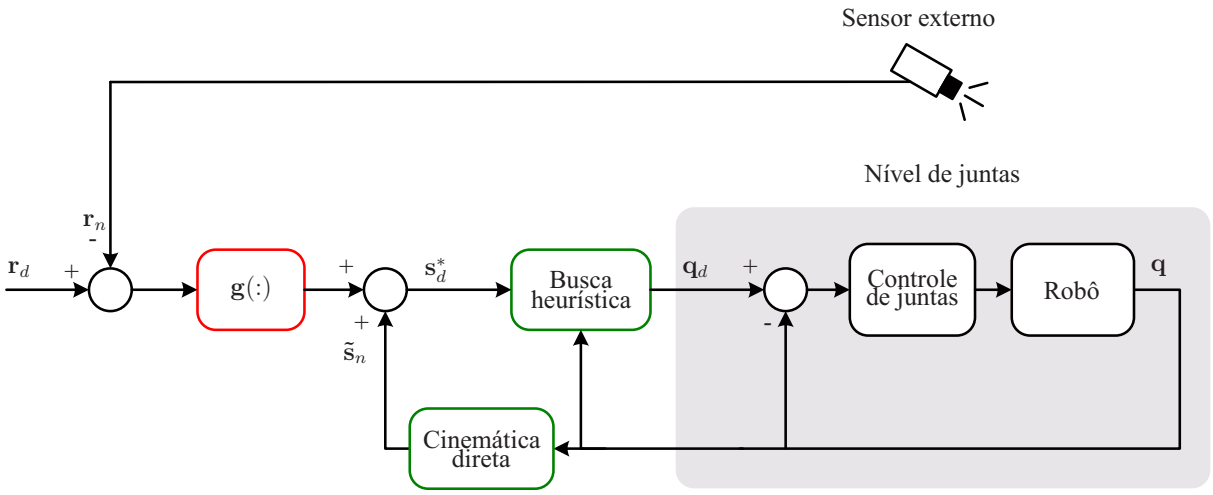


Fig. 3.4: Diagrama de blocos do sistema de controle incluindo sensoriamento externo.

A equação 3.31 pode ser interpretada como uma mudança das coordenadas da referência do espaço de trabalho real para o espaço de trabalho do robô modelado, usado na busca. Tal mudança de referência implica numa compensação dos erros de modelagem da cinemática do robô. O processo de busca irá minimizar a diferença $\mathbf{s}_d^* - \tilde{\mathbf{s}}_n$, que equivale a fazer $\mathbf{g}(\mathbf{r}_d - \mathbf{r}_n) \rightarrow 0$. Do ponto-de-vista cinemático, $\mathbf{g}(:)$ deve ser definida positiva para que o processo convirja ao longo do tempo.

Capítulo 4

Exemplos de Aplicação: Simulações

Apresenta uma série de implementações, no nível de simulação, e comparações dos métodos relacionados procurando demonstrar as vantagens potenciais da metodologia proposta na tese.

De maneira a demonstrar a aplicabilidade do processo de rastreamento de trajetórias proposto neste trabalho, foram realizadas diversas simulações no MATLAB/SIMULINK. As simulações foram divididas em 4 estudos de caso. O primeiro trata a comparação do método de busca heurística com uma abordagem tradicional quando ocorrem singularidades matemáticas. No segundo estudo de caso, é analisado o comportamento do algoritmo recursivo na resolução da cinemática inversa para robôs redundantes sob restrições de posição e velocidade de juntas. Novamente, o método proposto neste trabalho é comparado com uma abordagem tradicional para o mesmo problema. O terceiro estudo de caso apresenta resultados da aplicação do método de busca heurística para o rastreamento de trajetórias definidas tanto em posição quanto em orientação. Finalmente, o quarto exemplo mostra a aplicação do método proposto a um sistema de controle servo visual no qual tanto os parâmetros do robô quando da câmera possuem imprecisões de calibração.

4.1 Estudo de Caso 1: Comportamento em Singularidades

Neste estudo de caso, três simulações foram realizadas com base num braço antropomórfico [1], cujos parâmetros DH são dados na tabela 4.1.

Em todas as simulações, o critério de parada, cp, usado foi $\sqrt{h}$, que corresponde à distância euclidiana entre as posições desejada e atual da garra do robô. A tolerância requerida foi $tol = 10^{-7}\ m$.

Elo i	$a_i\ (m)$	$\alpha_i\ (rad)$	$d_i\ (m)$	$\theta_i\ (rad)$
1	0	$\pi/2$	0	θ_1
2	1	0	0	θ_2
3	1	0	0	θ_3

Tab. 4.1: Parâmetros DH do braço antropomórfico.

O objetivo da primeira simulação foi analisar o comportamento da busca heurística enquanto o o robô segue uma trajetória que leva a uma configuração singular. A configuração inicial do robô era $\mathbf{q} = [0\ \pi/6\ -\pi/3]^T\ rad$, correspondendo a uma posição inicial da garra $\mathbf{p}_d(0) = [0\ 0\ 1.7320]^T m$. O manipulador deveria seguir uma trajetória circular com diâmetro de $1.7320\ m$, cruzando o ponto de singularidade $\mathbf{p}_d = [0\ 0\ 0]^T\ m$. A lei de tempo associada com a trajetória foi dada por um polinômio de quinto grau, o que permite que a trajetória inicie e termine com velocidades e acelerações nulas. O período de amostragem da trajetória foi $\Delta t = 0.01s$ e o tempo total da mesma foi de $6\ s$. Os resultados obtidos para as posições, velocidades e acelerações das juntas, assim como a norma euclidiana do erro de posicionamento para a referida trajetória estão mostradas na figura 4.1. Como pode-se notar, o método de busca heurística gerou trajetórias suaves mesmo quando o manipulador passou pelo ponto de singularidade no instante $t = 3\ s$. Também pode-se observar que o erro de posicionamento assumiu valores desprezíveis, inferiores a $4,5 \times 10^{-8}\ m$.

De modo a comparar o método de busca heurística com uma abordagem tradicional que tenta lidar com singularidades, o método dos mínimos quadrados amortecidos, descrito brevemente na seção 2.1.3 e detalhado em [37], foi também aplicado à trajetória do primeiro exemplo com os respectivos parâmetros $\mathbf{K}_0 = 12\mathbf{I}$, $\lambda^2 = 0.016$ e $\varepsilon = 0.04$, escolhidos em processo exaustivo de tentativa e erro. O tempo de amostragem foi $\Delta t = 0.01s$. Os resultados deste experimento estão apresentados na figura 4.2. Diferentemente do método de busca heurística, a técnica dos mínimos quadrados amortecidos não gerou trajetórias suaves para as velocidades das juntas à medida que o manipulador se aproximou da configuração singular. Contudo, as velocidades ficaram limitadas. O fator de amortecimento foi acionado aproximadamente no instante $t = 2.85\ s$, correspondente a um aumento significativo no erro de posicionamento com o esperado.

A terceira simulação teve o objetivo de analisar o algoritmo proposto quando todos os pontos da trajetória são singulares. Neste caso, o braço antropomórfico teve que rastrear uma trajetória linear no eixo z_0 durante $5\ s$. Os pontos dessa trajetória correspondem à denominada singularidade de ombro. Novamente a lei de tempo associada à trajetória foi um polinômio de quinto grau. A trajetória teve início com o robô na configuração inicial $\mathbf{q} = [\pi/9,\ \pi/3,\ \pi/3]^T\ rad$, com $\mathbf{p}_d(0) = [0\ 0\ 1.7320]^T m$, e terminou em $\mathbf{p}_d(5) = [0\ 0\ -1.7320]^T m$. Os resultados correspondentes obtidos são mostrados na

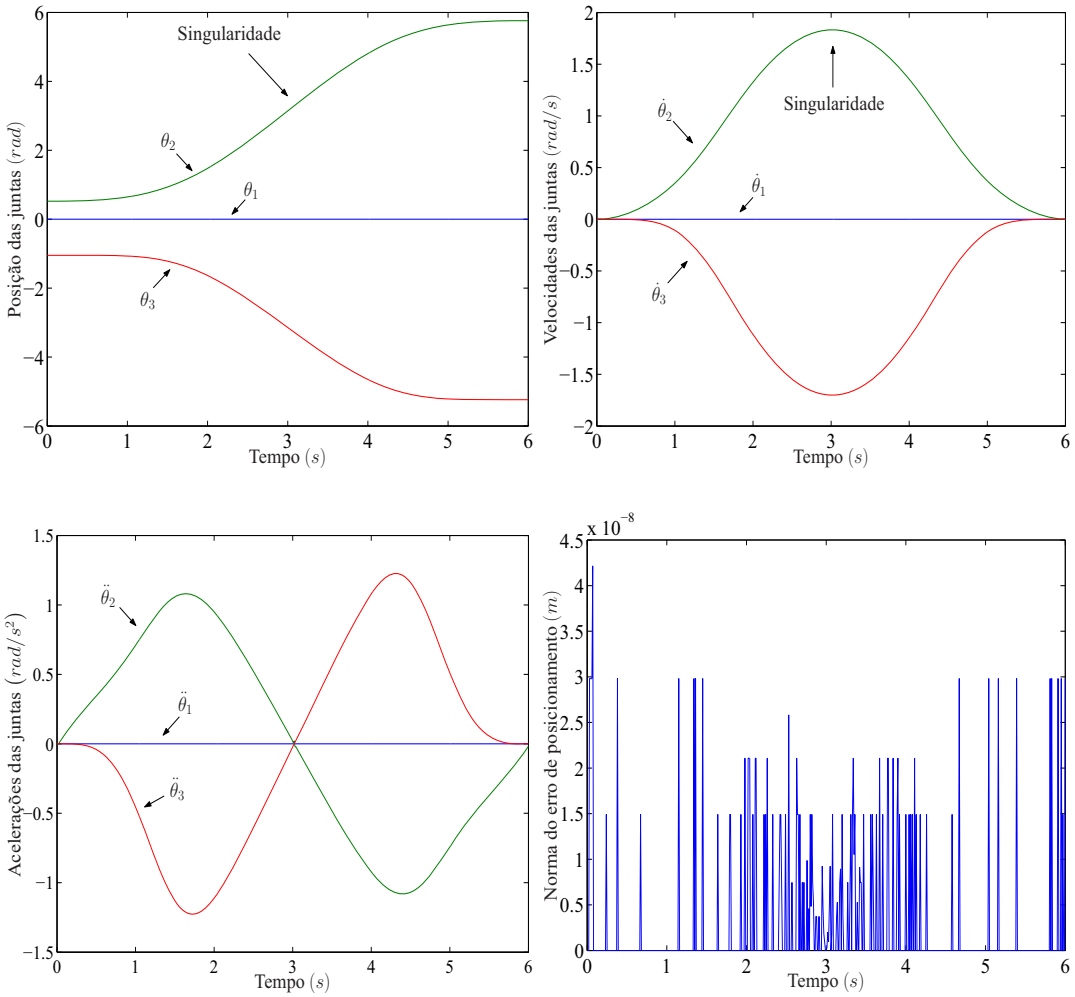


Fig. 4.1: Posições, velocidades, acelerações e norma do erro de posicionamento para o método de busca heurística.

figura 4.3. Mesmo a trajetória requerida estando em pontos singulares, o método de busca heurística foi capaz de gerar movimentos suaves nas juntas. Neste caso, o método dos mínimos quadrados falha.

4.2 Estudo de Caso 2: Resolução de Redundâncias e Manipulação de Limites

Neste segundo estudo de caso, o problema da resolução de redundâncias matemáticas foi abordado, assim como o tratamento de limitação das variáveis de junta [85]. O robô usado neste experimento foi um manipulador planar redundante de 4 graus de liberdade, cujos comprimentos dos elos são de $0.2\ m$.

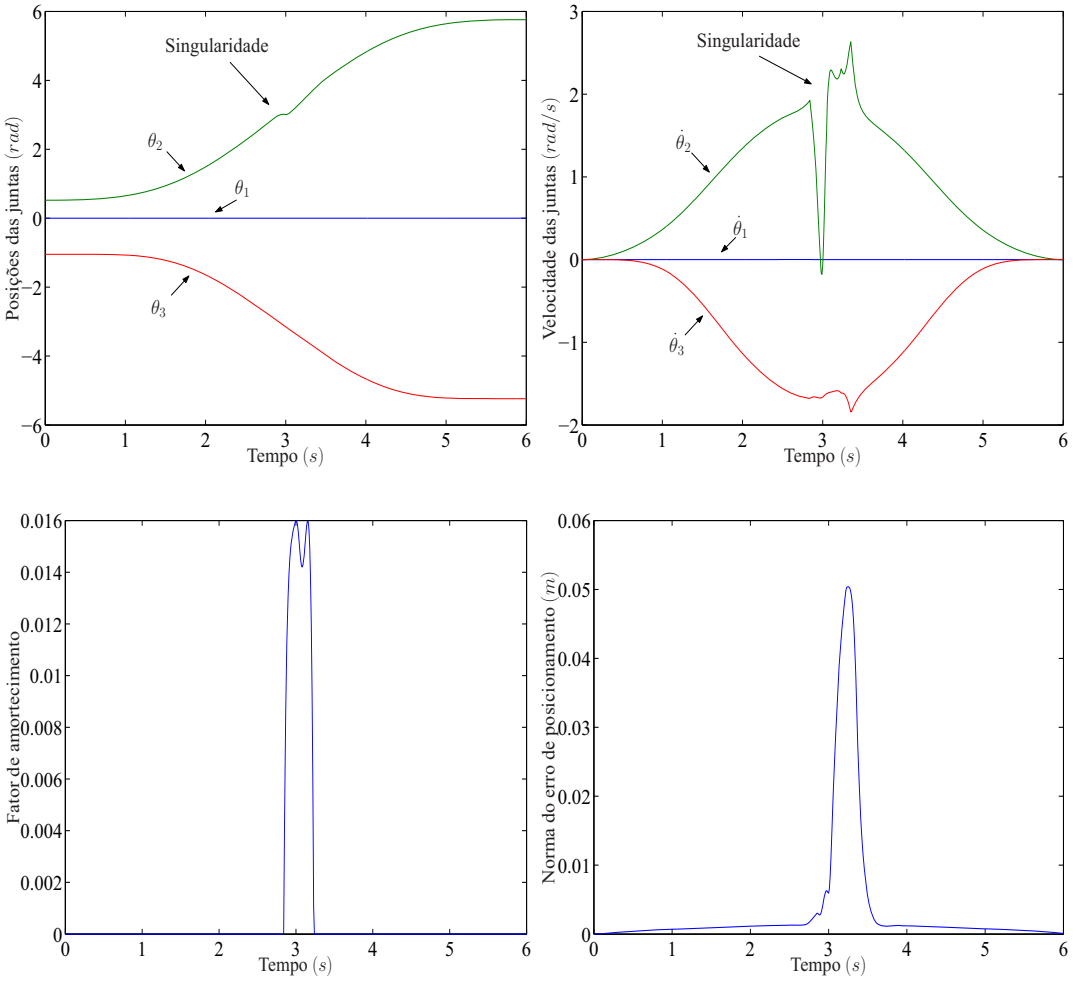


Fig. 4.2: Posições, velocidades, acelerações e norma do erro de posicionamento para o método DLS.

Todas as trajetórias usadas nesse estudo de caso são circulares, ou elípticas, e foram geradas usando a função `refcirc.m` do *Planar Manipulators Toolbox* [86]. Em todas as simulação, o período de amostragem usado foi $\Delta t = 0.001\ s$. A tolerância usada para verificar se um ponto foi rastreado foi de $10^{-5}\ m$, assim como a seqüência de juntas usada foi *1,2,3,4*.

A primeira simulação serve para verificar o comportamento do algoritmo recursivo quando o manipulador alcança limites de juntas. A trajetória requerida para o manipulador foi um círculo de raio $0.15\ m$ e período de $1\ s$. Neste primeiro caso, os limites de posição para a junta foram $\theta_{4,min} = 0.60\ rad$ e $\theta_{4,max} = 0.85\ rad$ e a configuração inicial do robô foi $\theta_0 = [\pi/4,\ \pi/6,\ \pi/2,\ \pi/4]^T\ rad$. A figura 4.4 mostra as posições das juntas para esta trajetória, assim como o respectivo erro de rastreamento. É importante observar que a estratégia de penalização de velocidades, equações 3.29 e 3.30, não foi usada, o que resultou em paradas bruscas da junta 4, embora o erro de rastreamento resultante não tenha sido afetado.

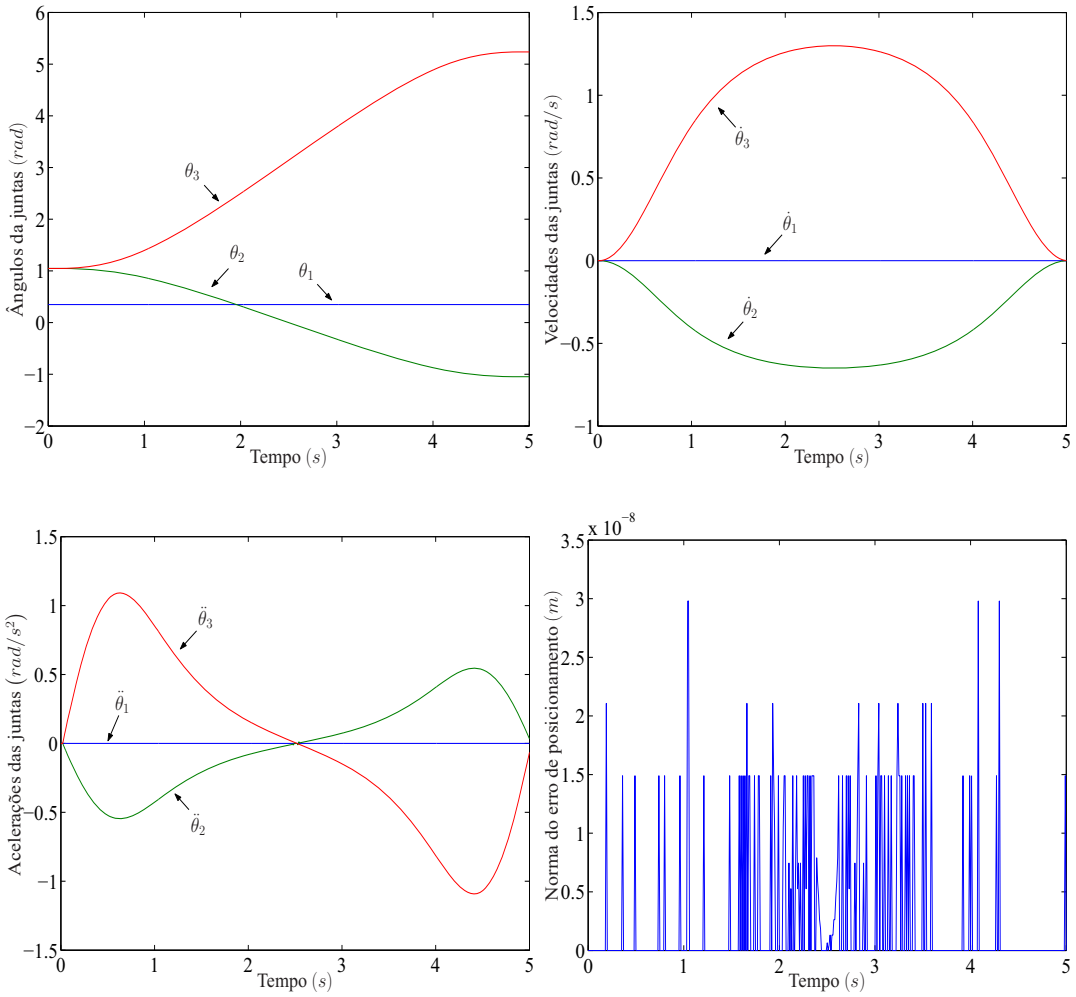


Fig. 4.3: Posições, velocidades, acelerações e norma do erro de posicionamento para o método de busca heurística numa trajetória singular.

Este mesmo experimento foi repetido, mas a velocidade da junta 4 é que foi limitada, $\dot{\theta}_{4,min} = -0.50\ rad/s$ e $\dot{\theta}_{4,max} = 0.5\ rad/s$ em vez da posição. Os resultados correspondentes são apresentados na figura 4.5. Novamente, embora a velocidade da junta 4 tenha ficado saturada (limitada) em alguns momentos, o erro de rastreamento continuou sendo desprezível.

Em seguida, o algoritmo recursivo proposto neste trabalho foi comparado com uma técnica tradicional para a resolução da cinemática inversa de robôs redundantes, o método da pseudo-inversa. Neste caso, escolheu-se uma trajetória elipsoidal repetitiva com raios x e y de $0.1\ m$ e $0.2\ m$ respectivamente e período de $2\ s$. O tempo total da trajetória foi de $50\ s$. O robô partiu de uma configuração inicial $\theta_0 = [\pi/3,\ \pi/3, -\pi/2, -\pi/2]^T\ rad$, correspondendo à posição inicial da garra $p_0 = [0.2732,\ 0.2732]^T\ m$.

A idéia desse experimento é a de mostrar que o algoritmo recursivo proposto pode ser usado para

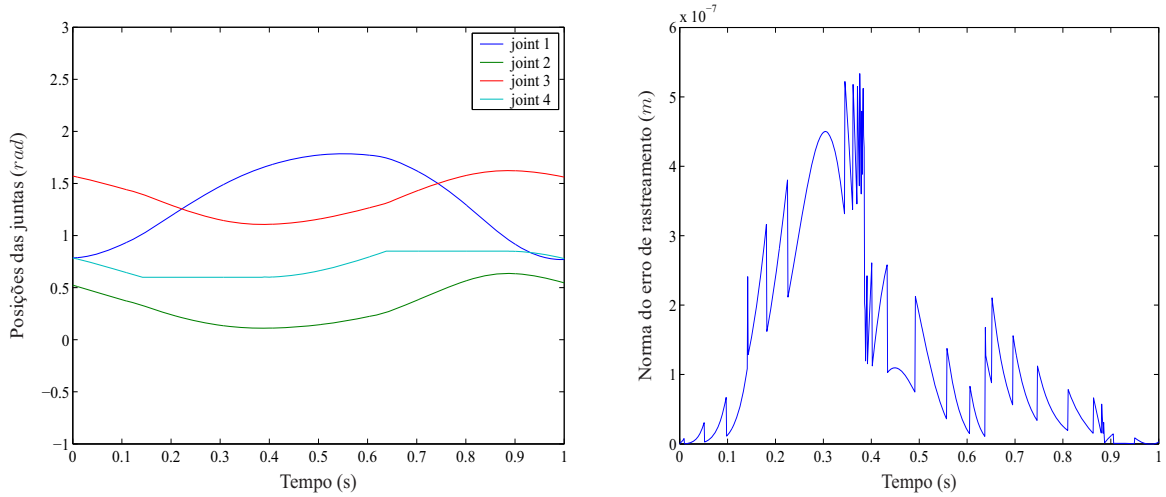


Fig. 4.4: Comportamento do algoritmo recursivo quando há limitação nas posições das juntas.

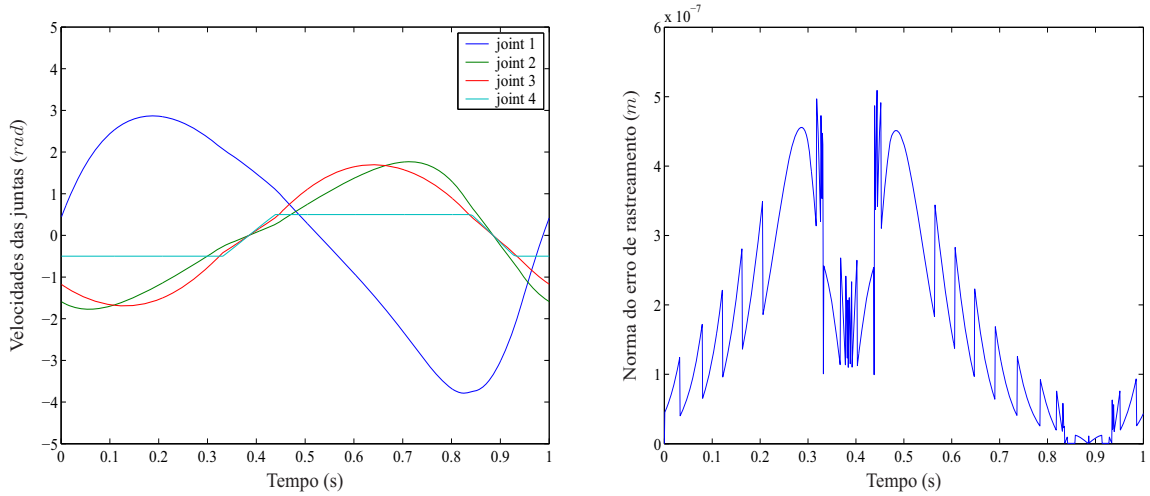


Fig. 4.5: Comportamento do algoritmo recursivo quando há limitação nas velocidades das juntas.

assegurar repetibilidade para robôs redundantes através da imposição de limites para as posições das juntas. Inicialmente, foram permitidas amplas faixas de variação para as posições das juntas, $\theta_{i,min} = -\pi\ rad$ e $\theta_{i,max} = \pi\ rad$ para todo i pertencente à seqüência de juntas predefinida. Posteriormente, a configuração inicial do robô foi modificada, num estágio simulado, para a posição intermediária dos limites de posição das juntas, que é, $\theta_0 = [0,\ 0,\ 0,\ 0]^T$ (uma configuração singular para a técnica da pseudo-inversa), e, em seguida, $\mathbf{p}_0$ foi rastreado. Esta operação é equivalente a se reconfigurar o robô antes do início do rastreamento da trajetória. A idéia aqui é a de partir o robô de uma configuração que esteja longe dos limites de junta.

Após este procedimento inicial, o rastreamento teve início e, após 6 s os limites de posição da juntas foram modificados para os correspondentes valores mínimos e máximos que as posições das juntas assumiram durante este período inicial. Os resultados correspondentes ao rastreamento da

trajetória repetitiva usando o método da pseudo-inversa, equação 2.15, com $\mathbf{K} = diag(100)$, e o algoritmo recursivo são mostrados nas figuras 4.6 e 4.7 respectivamente. Para este caso, a estratégia de penalização de velocidade, equações 3.29 e 3.30, foi usada, evitando assim paradas bruscas nas juntas. Diferentemente do algoritmo recursivo e, como esperado, o método da pseudo-inversa não gerou trajetórias repetitivas no espaço de juntas.

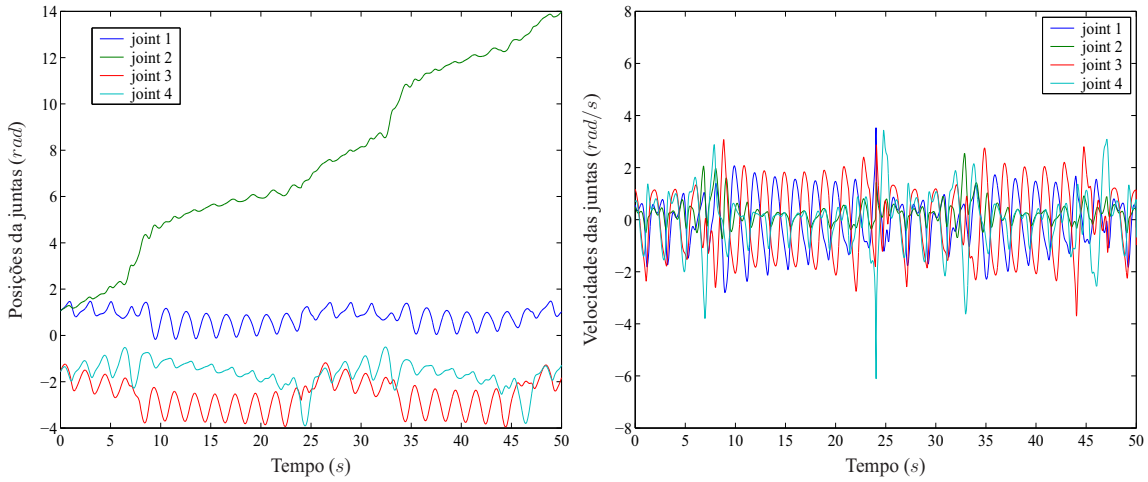


Fig. 4.6: Posições e velocidades das juntas para o método da pseudo-inversa.

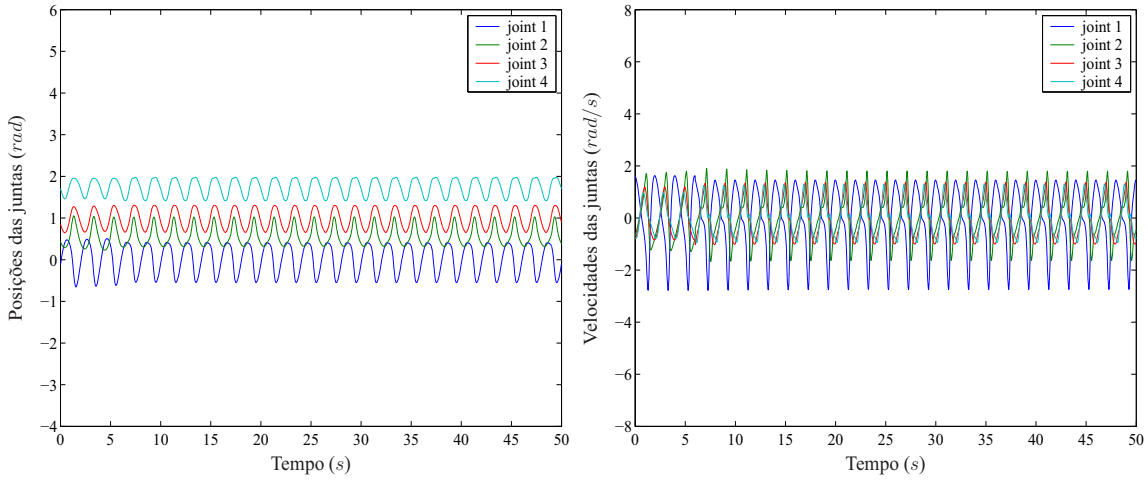


Fig. 4.7: Posições e velocidades das juntas para o algoritmo recursivo.

4.3 Estudo de Caso 3: Rastreamento de Trajetórias Definidas em Posição e Orientação

O último estudo de caso trata da aplicação do algoritmo recursivo no rastreamento de trajetórias definidas em posição e orientação (pose). Neste exemplo, foi usado o modelo cinemático de um robô do tipo PUMA, cujos parâmetros de Denavit-Hartenberg são dados na tabela 4.2.

Elo i	$a_i\ (m)$	$\alpha_i\ (rad)$	$d_i\ (m)$	$\theta_i\ (rad)$
1	0	$-\pi/2$	0	θ_1
2	$0,4318$	0	0	θ_2
3	$0,0203$	$-\pi/2$	0,1254	θ_3
4	0	$\pi/2$	0,4318	θ_4
5	0	$-\pi/2$	0	θ_5
6	0	0	0	θ_6

Tab. 4.2: Parâmetros DH do robô tipo PUMA.

A trajetória foi gerada usando a função `ctraj.m` do *Robotics toolbox for MATLAB* [69]. As poses inicial, $\mathbf{T}_{inicial}$, e final, $\mathbf{T}_{final}$, da trajetória em questão são dadas pelas equações 4.1 e 4.2 respectivamente. A lei de tempo usada na trajetória foi estabelecida por um polinômio de quinto grau e o intervalo de amostragem foi de $\Delta t = 0.01\ s$. O tempo total da trajetória foi de $2\ s$. O critério usado para verificar se uma pose foi rastreada foi $\sqrt{h_i}$, com h_i calculado de acordo com a equação 3.25 e $\gamma = 0.01$. A respectiva tolerância foi de 10^{-8}, assim como a seqüência de juntas usada foi *da junta* 1 *para a junta* 6. A configuração inicial do robô, correspondente a $\mathbf{T}_{inicial}$, era $\mathbf{q} = [\pi/4,\ \pi/4,\ \pi/3,\ \pi/3,\ \pi/3,\ \pi/5]^T$.

$$\mathbf{T}_{inicial} = \begin{bmatrix} 0,0331 & 0,6088 & -0,7925 & -0,1714 \\ -0,8779 & 0,3967 & 0,2680 & 0,0059 \\ 0,4776 & 0,6869 & 0,5476 & -0,2131 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

$$\mathbf{T}_{final} = \begin{bmatrix} -0,5291 & 0,1251 & -0,8392 & -0,2118 \\ 0,8081 & 0,3756 & -0,4535 & -0,1160 \\ 0,2585 & -0,9182 & -0,2999 & -0,2817 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

As posições e velocidades, assim como os respectivos erros de posicionamento e orientação, assumidas pelas juntas durante o rastreamento da trajetória referida anteriormente estão mostradas na

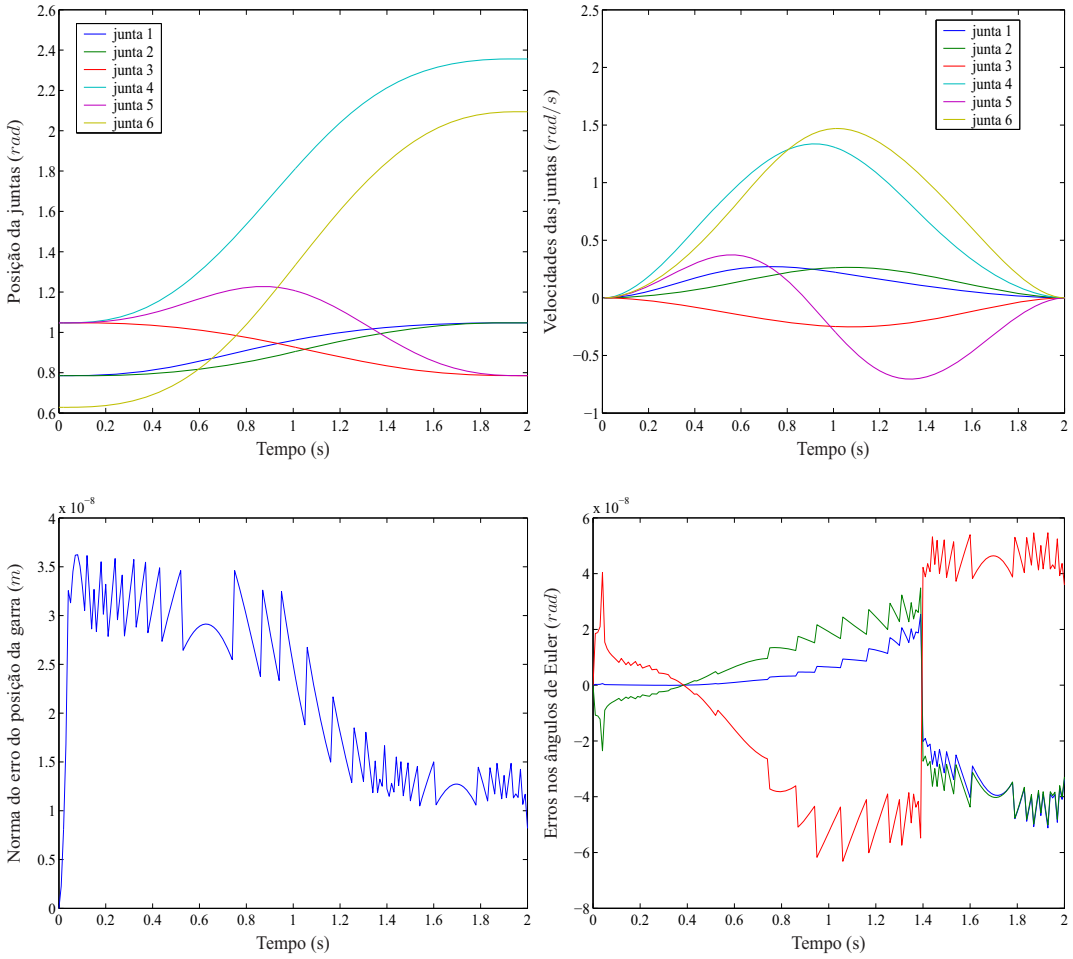


Fig. 4.8: Comportamento do algoritmo recursivo no rastreamento de trajetória definidas em posição e orientação.

figura 4.8. Nesta figura, pode-se observar que os erros assumiram valores desprezíveis durante toda a trajetória. No caso da orientação, os erros foram determinados em termos dos ângulos de Euler das orientações requeridas, embora a trajetória rastreada tenha sido definida em termos de matrizes de rotação. Isto se deve ao fato dos ângulos de Euler facilitarem a interpretação dos erros de orientação.

4.4 Estudo de Caso 4: Inclusão de Sensor Externo no Sistema de Controle

Objetivo deste exemplo é verificar a possibilidade de aplicação da metodologia proposta para sistemas de controle de robôs, usando câmera de vídeo como sensor externo, para eliminar os efeitos das imprecisões nas medidas dos parâmetros do robô e da câmera. Para tal, elaborou-se um sistema

Elo	1	2	3
Comp. (m)	0,2	0,2	0,2
Centro de massa (m)	0,1	0,1	0,1
Inércia ($Kg \times m^2$)	0,003018	0,00211	0,000748
Massa (Kg)	0,565	0,95	0,140
Coef. Atrit. Viscoso (N-m/rad)	0,2	0,2	0,2

Tab. 4.3: Parâmetros usados na simulação dinâmica do robô planar de 3 DOF

de simulação baseado no $Simulink$, cujo diagrama de blocos é apresentado na figura 4.9. O modelo

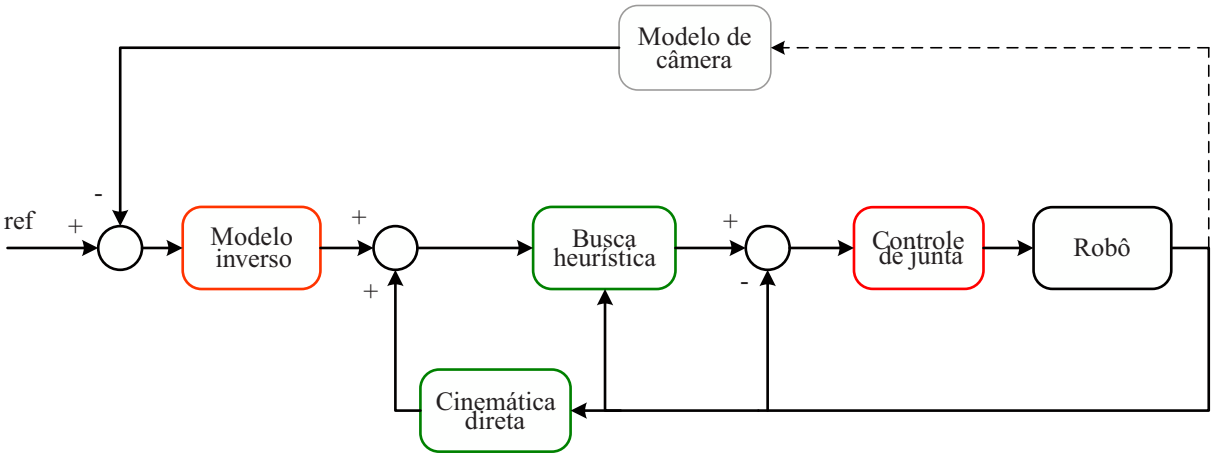


Fig. 4.9: Diagrama de blocos do sistema de simulação do controle servo visual.

dinâmico do robô foi simulado a partir do *Planar Manipulator Toolbox* [86], sendo que o robô considerado é o planar de 3 DOF descrito no apêndice A. Os parâmetros cinemáticos e dinâmicos usados na simulação estão mostrados na tabela 4.3. Também foram incluídos efeitos de zona morta (*dead zone*), saturação do sinal de controle e *backlash* nas juntas usando blocos básicos do *Simulink*. Mais especificamente, o sinal de controle é saturado fora dos limites de -3 e 3 *u.m.*, a vai de -1 a 1 *u.m* do sinal de controle. Para o *backlash* foi estipulada uma banda morta de 0,04. rad.

Nesse modelo de simulação, foi considerado um esquema de controle servo visual baseado na imagem (IBVS), com a câmera montada externamente ao robô. Essa câmera tem a função de monitorar a posição da garra do robô no plano de imagem. O controle de juntas é realizado por controladores do tipo PID, cujos ganhos são mostrados na tabela 4.4 e foram definidos de forma empírica. Os parâmetros (intrínsecos) reais da câmera são dados pela transformação

$$\mathbf{C} = \begin{bmatrix} 400 & 0 & 320 \\ 0 & 400 & 240 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.3}$$

Junta	K_P	K_I	K_D
1	150	10	3
2	150	10	3
3	150	10	3

Tab. 4.4: Ganhos dos controladores PID da simulação.

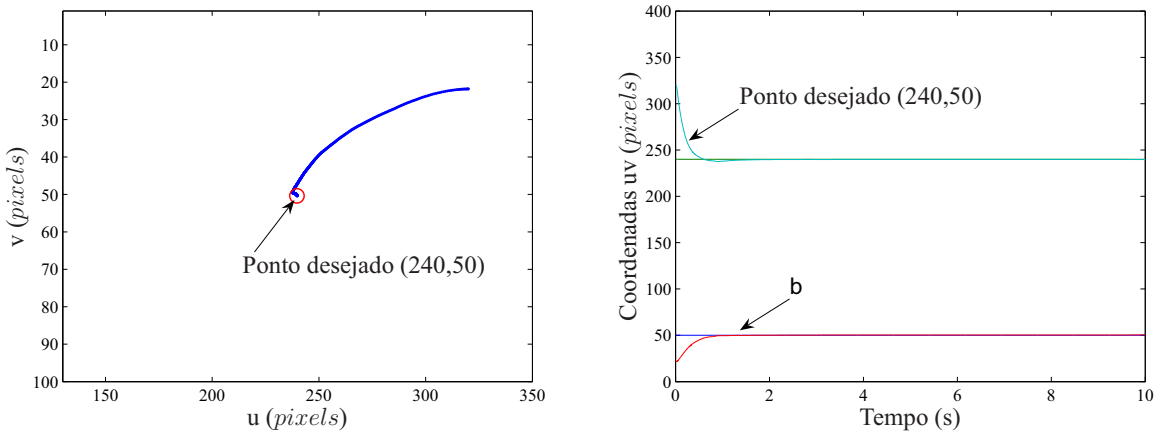


Fig. 4.10: Resposta do sistema para as coordenadas do espaço de câmera.

Além disso, a relação entre a pose da câmera e o sistema de coordenadas inercial associado ao robô é dada por

$$ {}^{0}\mathbf{A}_c = \begin{bmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.4} $$

Para a simulação realizada, os parâmetros aproximados para a câmera são dados como

$$ \tilde{\mathbf{C}} = \begin{bmatrix} 200 & 0 & 0 \\ 0 & 300 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.5} $$

$$ {}^{0}\tilde{\mathbf{A}}_c = \begin{bmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1.1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.6} $$

Os parâmetros cinemáticos do robô, no caso os comprimentos dos elos, usados pelo algoritmo de busca foram $\mathbf{l} = [0,22 \;\; 0,19 \;\; 0,17]^T$. Baseado no ambiente descrito anteriormente, foi realizada uma

simulação do robô sendo comandado a rastrear o ponto $\mathbf{m}' = [240\ 50]^T\ \ pixels$ no plano de imagem. A configuração inicial das juntas do robô era $\mathbf{q} = [0\ 0\ 0]^T rad$. O tempo de simulação foi de 50 s e o método de integração empregado foi o Runge-Kutta de quarta ordem com intervalo de integração fixo de 2 ms, que é também o período de amostragem do sistema de controle de juntas. O período de amostragem da câmera de vídeo foi fixado em 40 ms (25 QPS). A figura 4.10 mostra a trajetória descrita pela garra do robô até alcançar o ponto desejado e também a evolução de cada coordenada da imagem até o valor predeterminado. Pode-se notar que o movimento resultante foi suave e que o tempo de rastreamento foi de aproximadamente 2 s.

Já a figura 4.11 apresenta as evoluções dos erros da garra do robô em relação ao ponto desejado e também os movimentos das juntas gerados pela processo de busca e que foram efetivamente realizados pela máquina. É importante ressaltar que o erro de regime permanente no plano da imagem foi da ordem de 1 *pixel*.

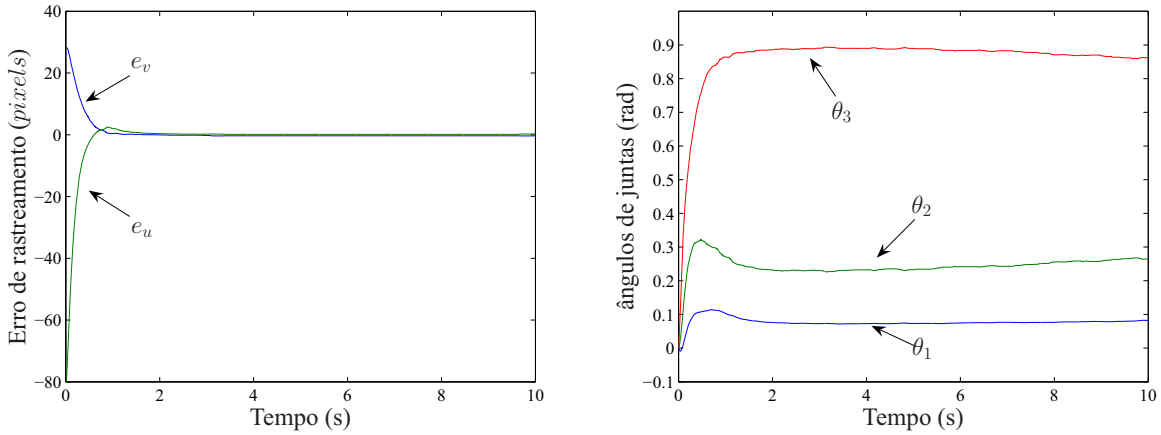


Fig. 4.11: Evolução do erro de rastreamento no espaço de câmera e movimentos de junta executados pelo robô.

Capítulo 5

Implementação Prática

Este capítulo apresenta uma visão geral do ambiente experimental desenvolvido e os principais resultados obtidos nos experimentos realizados com o emprego da metodologia proposta, aplicada a um sistema de controle robótico com um manipulador planar redundante de três graus de liberdade.

5.1 Visão Geral do Sistema Experimental

De modo a validar na prática os conceitos explorados no trabalho, foi desenvolvido um ambiente experimental baseado num robô planar redundante de 3 graus de liberdade, assim como todo o sistema de acionamento, sensoriamento, interfaceamento e controle associado. A figura 5.1 mostra uma visão geral do sistema e suas interconexões.

As etapas de supervisão e controle em tempo real são implementadas em dois microcomputadores, um chamado de hospedeiro (*host*) e outro chamado de alvo (*target*), interconectados por interfaces *Ethernet* e/ou *RS232*. Este sistema de processamento é baseado na metodologia *hardware-in-the-loop*, descrita na seção 5.2, desenvolvida no *Matlab/Simulink* sobre as plataformas *xPC target* [87] e *Real-Time Workshop* [88] da *MathWorks*.

O computador hospedeiro é responsável pelas pela supervisão de todo o sistema e também pelo processamento das imagens provenientes da câmera digital com interface *USB*. O computador alvo por sua vez roda as tarefas de tempo-real e comunicação com o hardware de interfaceamento/processamento dos sinais de controle e sensoriamento do robô. Esta placa possui também interfaces, não usadas no presente trabalho, para conexão de câmera e monitor de vídeo analógico, além de comunicações RS232 e USB, como mostrado pela linha vermelha tracejada na figura 5.1. A parte específica do trabalho que tratou do projeto e construção da placa de processamento baseada e dispositivos lógicos programáveis gerou um trabalho de iniciação científica [89] e uma dissertação de mestrado [90].

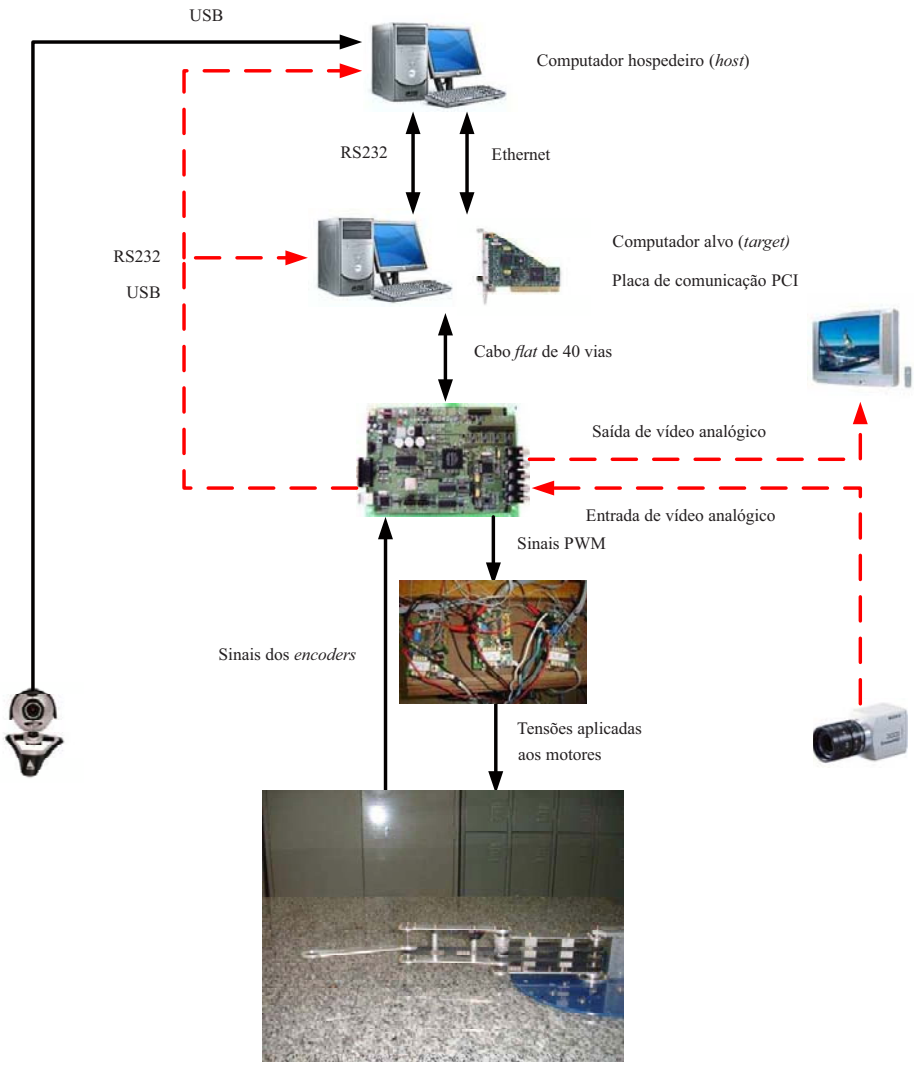


Fig. 5.1: Diagrama de blocos do sistema experimental.

Os circuitos de interfaceamento com o sistema de controle, condicionamento dos sinais de controle e sensores (*encoders*) da máquina são implementados no dispositivo lógico programável da placa de desenvolvimento. A comunicação com p computador alvo é feita por meio de um cabo *flat* de 40 vias que está conectado a uma placa de aquisição de sinais digitais com interface *PCI* da *National Instruments*.

Os sinais de controle (PWM) provenientes da placa desenvolvimento são enviados para as placas de acionamento que amplificam esses sinais gerando as tensões correspondentes a serem aplicadas aos terminais dos motores do robô.

5.2 Sistema *Hardware-in-the-loop*

A etapa de processamento do sistema, isto é, algoritmo de busca e controle, é implementado usando o *xPC target* da *MathWorks* um chamado de algo (*target*) e outro chamado de hospedeiro (*host*) ligados via Ethernet. A função do *xPC target* é permitir que sistemas de controle e processamento desenvolvidos no *Simulink* possam ser rapidamente testados na prática, este tipo de abordagem para a implementação prática é chamada de *hardware-in-the-loop* (HIL). O sistema funciona da seguinte forma:

1. Todo o processo de busca e controle é implementado no simulink, no computador hospedeiro, usando blocos básicos ou em linguagem C usando s-functions;

2. Depois que o sistema foi devidamente simulado no computador hospedeiro, os blocos do modelo *Simulink* são transformados em linguagem C e compilados, no caso desse trabalho, usando *Watcom* 1.3;

3. Este arquivo compilado é então carregado no computador, que roda um sistema operacional de tempo real, alvo via interface Ethernet. Dessa forma, o computador alvo passa a atuar como o controlador do sistema;

4. Uma vez que o modelo tenha sido compilado e carregado no computador alvo, o hospedeiro pode se conectar com ele, tenho a possibilidade de iniciar e parar o processamento, fazer aquisição (*logging*) de sinais e dados referentes ao processamento, modificar parâmetros do modelo, como ganhos de controladores, etc.

Dessa maneira, o processo de verificação prática fica integrado no mesmo ambiente de modelagem, simulação e projeto do sistema de controle, veja figura 5.2. Além disso, torna-se possível usar os recursos gráficos disponíveis no *MatLab* para a visualização a análise dos dados resultados obtidos.

Para a utilização do *xPC target*, o(s) intervalo(s) de amostragem do modelo são mantidos fixos de forma que o processo de criação do sistema de tempo real gere automaticamente as rotinas de interrupção para garantir esses intervalos. Nas implementações mostradas na tese os intervalos de amostragem foram de 2 ms para o sistema de controle de juntas e de 40 ms para o sistema de controle servo visual.

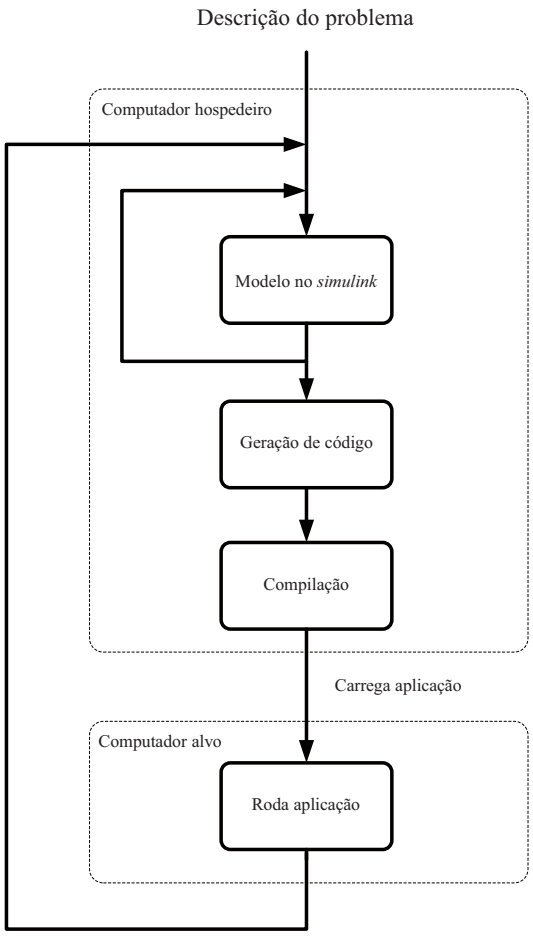


Fig. 5.2: Diagrama de fluxo de projeto no sistema *hardware-in-the-loop*.

5.3 Sistema Visual para a Determinação da Posição da Garra do Robô

Neste trabalho visou-se classificar os *pixels* de uma imagem de acordo com suas cores. Um *pixel* deve ser classificado como pertencente a uma das cores de interesse ou pertencente ao fundo (*background*). Para tal, foi utilizado um método de classificação conforme apresentado em [91]. Este método foi escolhido por ser bastante robusto a variações de iluminação.

Nesse esquema de processamento, Os *pixels* da imagem de entrada têm sua representação de cores transformada do espaço RGB para o plano de crominância (ES) do espaço YES conforme a equação 5.1.

$$\begin{bmatrix} E \\ S \end{bmatrix} = \begin{bmatrix} 0,50 & -0,50 & 0,00 \\ 0,25 & 0,25 & -0,50 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{5.1}$$

O espaço YES foi escolhido porque é linear e livre de singularidades, reduz a influência da intensidade luminosa e é computacionalmente eficiente (requer apenas deslocamentos de *bits* na obtenção das componentes E e S). Cada cor foi representada como uma classe no plano ES. A distribuição dessas classes (cores) no domínio ES foi realizada fazendo $\mathbf{x}_g$ representar um vetor composto das componentes E e S de um *pixel* na posição g. Então, a distribuição de $\mathbf{x}_g$ dentro de cada região foi modelada por uma gaussiana bidimensional dada pela equação 5.2.

$$p\left(\mathbf{x}_g \mid W_i\right) = \begin{array}{l} (2\pi)^{-n/2} \mid \mathbf{K}_i \mid^{-1/2} \\ exp\left\{-\frac{1}{2}[\mathbf{x}_g - m_i]^T \mathbf{K}_i^{-1}[\mathbf{x}_g - \mathbf{m}_i]\right\} \end{array}, \tag{5.2}$$

sendo que

$$\mathbf{x}_g = [E_g \;\; S_g]^T,\; \mathbf{m}_i = [m_E \;\; m_S]^T,\; e\; \mathbf{K}_i = \begin{bmatrix} \sigma_E^2 & \sigma_{ES} \\ \sigma_{ES} & \sigma_s^2 \end{bmatrix}, \tag{5.3}$$

e w_i, $i = 1, ..., N$ representa a classe de interesse. Este modelo é baseado na consideração de que o vetor de crominância na posição g, que pertence à região i, pode ser representado por um vetor de média da crominância da classe i e por uma gaussiana residual de média zero.

O contorno das funções de densidade de probabilidade (um para cada cor) na equação 5.2 define elipses, conforme equação 5.4, no domínio ES, cujos centros e eixos principais são determinados por $\mathbf{m}_i$ e $\mathbf{K}_i$, $i = 1, ..., N$.

$$\left[\mathbf{x}_g - \mathbf{m}_i\right]^T \mathbf{K}_i^{-1} \left[\mathbf{x}_g - \mathbf{m}_i\right] = \lambda_g^i. \tag{5.4}$$

A equação 5.4 define um mapeamento escalar no domínio ES sobre N estatísticas escalares para cada $pixel$ em g. O valor de $i = 1, ..., N$ é proporcional à probabilidade do $pixel$ pertencer à respectiva classe. Um pequeno valor numérico de implica em uma grande probabilidade dele pertencer a classe em questão.

A classificação dos $pixels$ é realizada comparando-se os valores com limiares globais, t_i, pré-definidos para cada classe, conforme a equação 5.5

$$pc_g = \begin{cases} 1 & Se\ \lambda_i \leq t_i \\ 0 & Senão \end{cases}. \tag{5.5}$$

Este classificador tem a capacidade de aceitar ou rejeitar uma determinada cor, o que é desejável, pois todas as outras cores são consideradas como pertencentes ao fundo. A seguir são apresentados resultados de classificação, obtidos via simulação, do classificador usado neste trabalho, como ilustração do que pode ser esperado usando-se a técnica para classificar (ou rejeitar) apenas uma cor.

Uma vez que se tenha a imagem de saída com a cor classificada, a determinação do centróide

$\mathbf{c} = [c_x \;\; c_y]^T$ do objeto da cor classificada se dá pela equação 5.6

$$\begin{aligned} c_x &= \frac{\sum x_{cor}}{A} \\ c_y &= \frac{\sum y_{cor}}{A} \end{aligned} , \tag{5.6}$$

sendo que x_{cor} e y_{cor} representam as coordenadas do $pixel$ pertencente à cor em questão, enquanto A é a área do objeto.

A figura 5.3 mostra um exemplo da classificação de cor e determinação de da posição da garra do robô implementado.

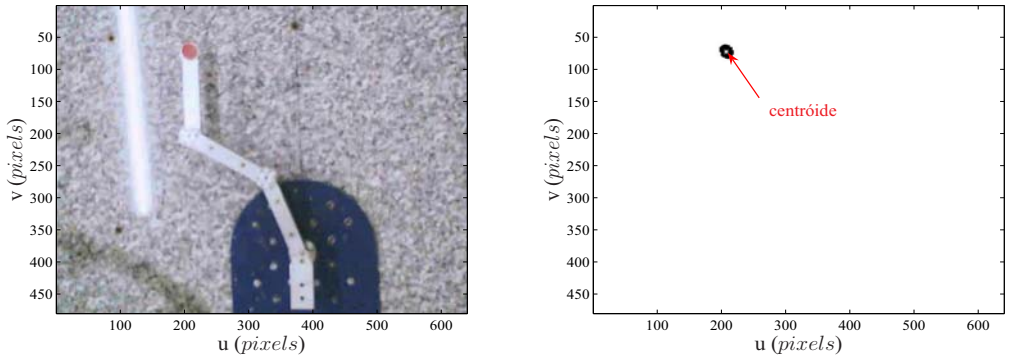


Fig. 5.3: Exemplo da classificação de cor e determinação de centróide.

A figura 5.4 mostra o sistema de visão usado no trabalho. A entrada de video é feita pelo bloco *Video Input* do *Image Acquisition BlockSet* do *Simulink*. Este bloco faz a aquisição de vídeo através do dispositivo de vídeo do *Windows*. O método de classificação de cor foi implementado em linguagem C via *s-Function*, enquanto que a determinação do centróide foi feita usando o bloco *blob analysis* do *Video and Image Processing BlockSet* do *Simulink*. Finalmente, o envio das coordenadas do centróide para o computador *target* é feita via interface *RS232* [92]. É importante salientar que este sistema não garante a determinação dos centróides em tempo real (intervalos de determinação fixos), uma vez que o sistema foi desenvolvido sobre o sistema operacional $Windows$, que não é de tempo real.

5.4 Resultados Experimentais

Dentre os experimentos realizados durante o desenvolvimento desta tese, foram escolhidos os resultados de quatro exemplos que se considera muito ilustrativos para demonstrar a aplicabilidade bem sucedida da metodologia criada, e para destacar as suas eficiências quando aplicada a um sistema robótico com arquitetura hierárquica de controle, e que necessite da solução da cinemática inversa em tempo real nos seus processos durante a realização de tarefas.

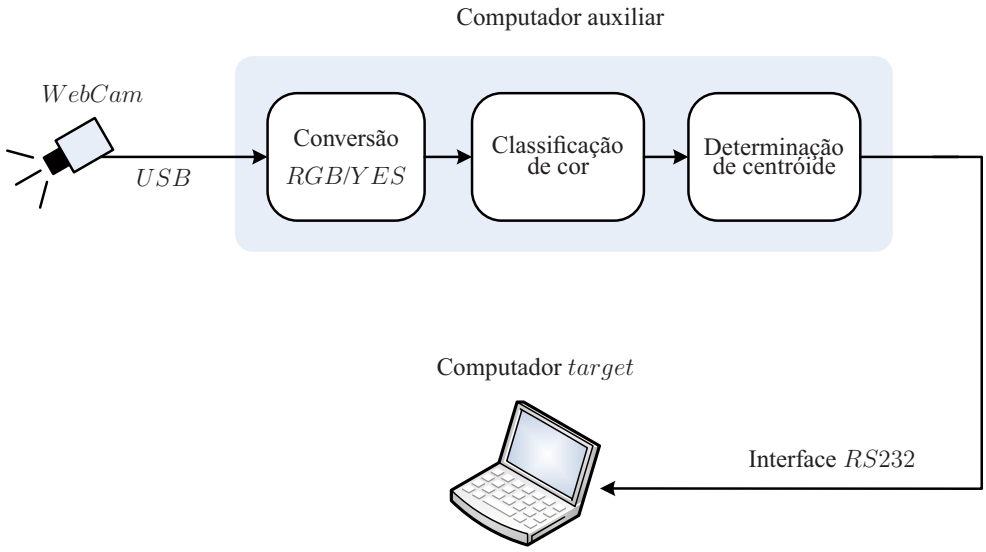


Fig. 5.4: Diagrama de blocos do sistema de visão.

Os objetivos principais destes experimentos apresentados aqui foram: mostrar como a metodologia é aplicável aos robôs, mesmo que tenham cadeia cinemática redundante; mostrar como ela pode auxiliar nos rastreamentos tipo ponto-a-ponto e de trajetórias contínuas; mostrar como ela pode auxiliar nos casos de aplicação em sistemas que possuem grandes incertezas paramétricas e controladores de juntas com projetos pouco sofisticados.

Para todos estes experimentos apresentados foram utilizados sistemas de controle de tempo real dos servomecanismos de juntas baseados no *xPC-Target/Matlab* com períodos de amostragem de 2 ms, e apenas controladores de classe PID com ganhos ajustados empiricamente.

5.4.1 Rastreamento de Ponto no Espaço de Trabalho

Neste experimento foi programado para o robô atingir o ponto $\mathbf{p}_d = [0,3 \;\; 0,3]^T \;\; m$ partindo da posição $\mathbf{p} = [0,6 \;\; 0,0]^T \;\; m$, que é uma posição de singularidade matemática do Jacobiano deste robô, e com velocidades inicial e final nulas. Também foi considerada a idealização de que todos os parâmetros necessários para a obtenção do modelo cinemático do robô através da técnica de *Denavit e Hartenberg* estão corretos. O resultado obtido pode ser visto na figura 5.5.

Pode-se perceber pela curva resultante que este movimento teve uma evolução suave na ponta do robô, um rumo coerente, e que o objetivo foi atingido. A realização desse movimento foi comandada para a máquina depois de 1 segundo que o sistema de controle foi posto em operação. A figura 5.6 mostra como foi a evolução do erro durante este mesmo movimento.

Pode-se notar que o movimento total durou aproximadamente 1 segundo, e que o objetivo foi alcançado com erro nulo. A figura 5.7 mostra a evolução do movimento da ponta do robô nas suas

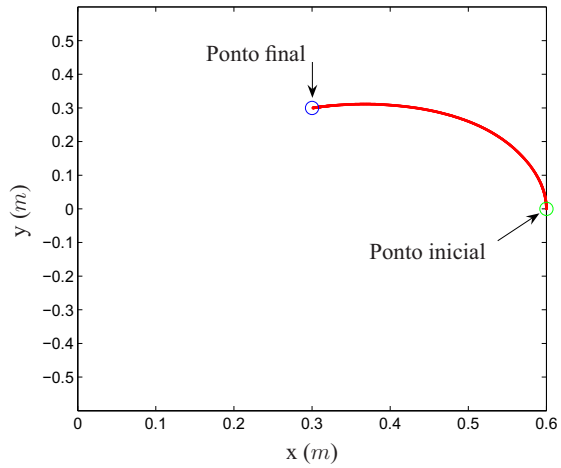


Fig. 5.5: Caminho descrito pela ponta do robô durante a tarefa.

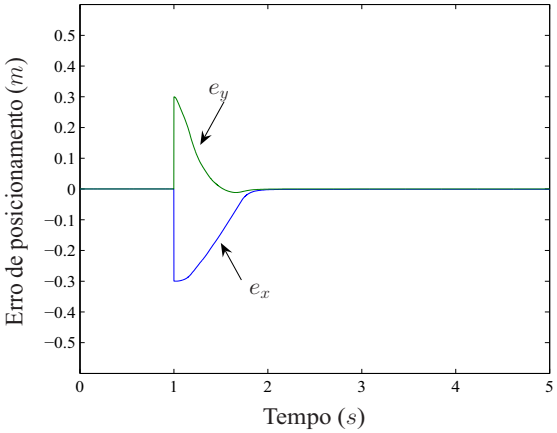


Fig. 5.6: Evolução do erro de rastreamento durante a tarefa.

coordenadas cartesianas x, y, e também as respectivas soluções encontradas pelo método de busca heurística x_d, y_d.

Verificou-se então que neste caso o método de busca heurística empregado consegue resolver o problema muito rapidamente. Com apenas um intervalo de amostragem de busca o problema foi resolvido e as respectivas referências de solução para os servomecanismos já estavam disponíveis. A figura 5.8 mostra como foram os movimentos realizados pelas juntas do robô.

Percebe-se portanto que também no espaço de juntas as trajetórias foram suaves, indicando que estas tenham propriedades de uma boa solução dentre as muitas soluções cinemáticas que o mesmo problema admitiria em se tratando de uma máquina estruturalmente redundante.

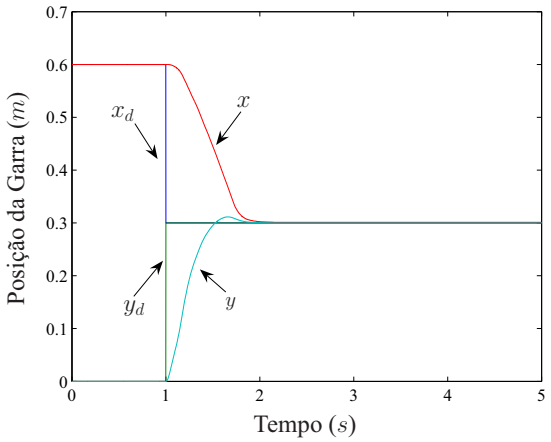


Fig. 5.7: Trajetórias da solução dada pela busca heurística.

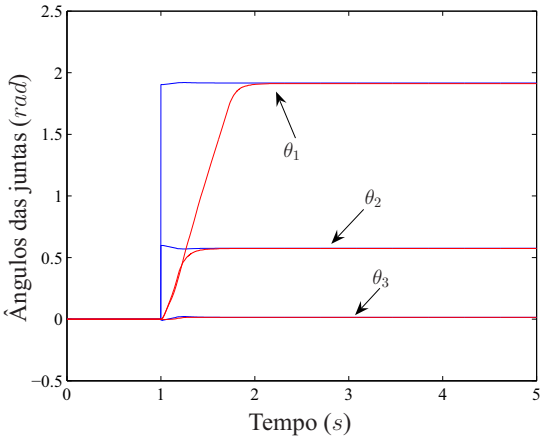


Fig. 5.8: Respostas das juntas do robô durante a tarefa.

5.4.2 Rastreamento com Ponderação nos Movimentos de Junta

Este segundo experimento teve objetivo de verificar detalhes do que pode ocorrer quando são impostas restrições ou limitações aos movimentos das juntas da máquina. O experimento consistiu numa ponderação, $w = 0,5$ imposta à junta 1 durante as buscas pelas soluções, de tal forma que ficasse permitido a ela realizar apenas metade do movimento que teria capacidade de realizar em cada momento do movimento. Então programou-se o sistema para realizar a mesma tarefa apresentada na sessão anterior, considerando também a idealização de que todos os parâmetros do modelo cinemático tivessem sido determinados sem erros ou variações.

O resultado obtido pode ser visto na figura 5.9.

Neste caso o movimento da ponta do robô também teve uma evolução suave, um rumo coerente, e o objetivo foi atingido. Pode-se notar que o movimento total durou também aproximadamente

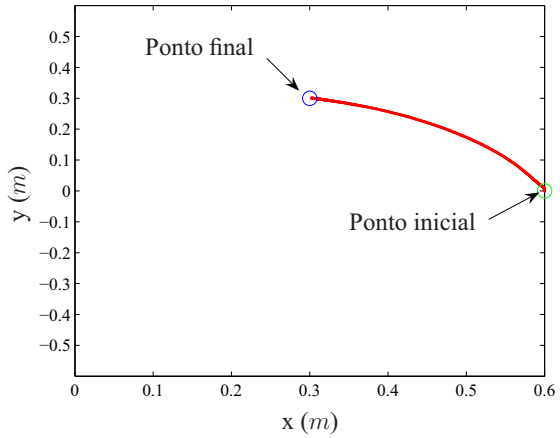


Fig. 5.9: Caminho descrito pela ponta do robô durante a tarefa.

1 segundo, e que o objetivo foi alcançado com erro nulo. A figura 5.10 mostra como foram os movimentos das juntas durante a execução dessa tarefa.

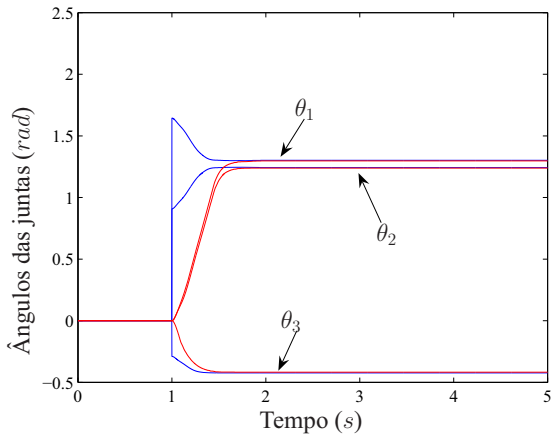


Fig. 5.10: Respostas das juntas do robô durante a tarefa.

Comparando-se a figura 5.10 com a figura 5.8 pode-se ver claramente que a máquina em ambas as situações atingiu os objetivos em mesmos tempos mas, como esperado, realizaram movimentos distintos. Em observação mais atenta é possível verificar que como forma de compensar a ponderação dada à junta 1, as juntas 2 e 3 realizaram maior movimentação em relação ao movimento que a máquina realizou sem tal ponderação.

5.4.3 Rastreamento de Trajetórias Cartesianas

O terceiro experimento teve por finalidade comprovar a capacidade do método de busca para rastrear trajetórias cartesianas repetitivas. Ele também teve como objetivo verificar experimentalmente

alguns bons resultados que haviam sido anteriormente obtidos via simulação e publicados em [93].

Para tal, o robô foi comandado a realizar circunferências periódicas de freqüências 0,5 Hz com raio de 0,1 m. A configuração inicial das juntas do robô foi $\mathbf{q} = [0 \;\; 0 \;\; 0]^T$ e o tempo total da tarefa foi fixado em 50 s. A seqüência de juntas usada foi {3, 2, 1}, sem imposição de limitação aos seus movimentos, e a tolerância da busca igual a 10^{-9} m. A figura 5.11 mostra as trajetórias de referência no espaço de trabalho e a realizada pelo robô.

A figura 5.12 mostra que o erro de rastreamento ficou inferior a 5mm em módulo. É importante observar que esse erro é causado pelo baixo desempenho dos controladores de juntas utilizados.

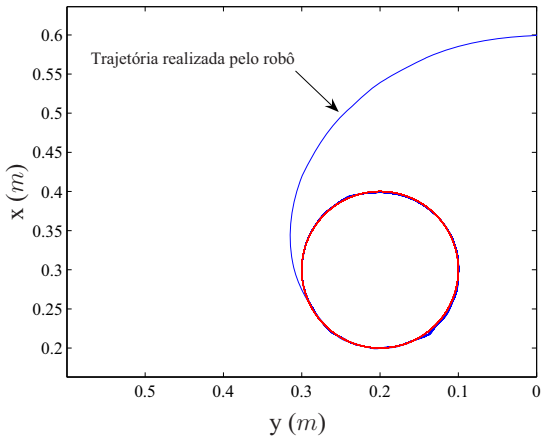


Fig. 5.11: Trajetória de referência e trajetória realizada pelo robô.

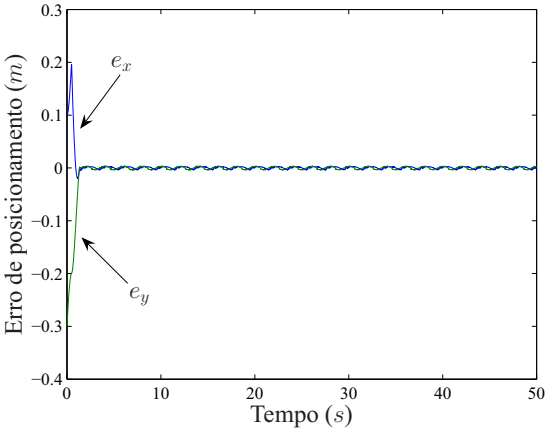


Fig. 5.12: Evolução do erro de rastreamento durante a tarefa.

A figura 5.13 mostra as respostas em posição das juntas durante a execução das trajetórias circulares da ponta do robô, e pode-se notar que são periódicas.

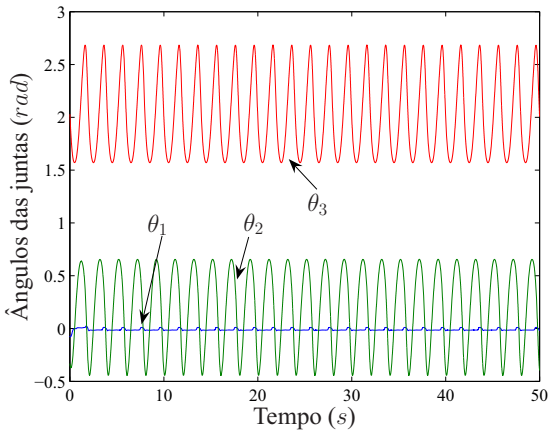


Fig. 5.13: Respostas das juntas do robô.

5.4.4 Rastreamento Servo-Visual em Sistema com Parâmetros Incertos

Neste quarto experimento procurou-se mostrar alguns importantes efeitos da inclusão de sensor externo ao robô como auxiliar nas buscas heurísticas.

No caso foi assumido que os parâmetros do modelo foram determinados incorretamete, e com erros grosseiros. Ou ainda, que foi admitida grande incerteza de parâmetros.

Os comprimentos dos elos do robô foram considerados como tendo comprimentos errados: $l_1 = 0,22$m, $l_2 = 0,19$m e $l_3 = 0,17$m, sabendo-se que por construção todos possuem comprimentos de 0,20 m. Também foi assumido desconhecimento do modelo da câmera utilizada na experimentação (webcam CCD *Labtec WCPRO*).

Empiricamente, o modelo inverso da câmera foi assumido como sendo

$$\tilde{\mathbf{C}} = \begin{bmatrix} 200 & 0 & 0 \\ 0 & 300 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5.7}$$

$$^{0}\tilde{\mathbf{A}}_{c} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1.1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.8}$$

Foram usados os seguintes ganhos para os controladores PID utilizados no controle dos servomecanismos de juntas:

Foi considerado um tempo de amostragem de 40ms na aquisição dos sinais de vídeo (25 quadros

Junta	1	2	3
P	12,0	12,0	12,0
I	3,0	3,0	3,0
D	12,0	10,4	9,6

Tab. 5.1: Ganhos dos controladores PID para o experimento.

por segundo), porém o sistema experimental montado e que realiza a captura das imagens e o seu processamento para a determinação do centróide referido à ponta do robô, não possibilita realizar essa taxa de forma precisa e em tempo real porque utilizou o sistema operacional *Windows*. A resolução usada no sistema de visão computacional foi de 640×480 $pixels$ e os parâmetros do processo de classificação de cor (vermelho) associada à ponta do são os seguintes:

$$\mathbf{m}_i = \begin{bmatrix} 40,91 \\ 25,59 \end{bmatrix} \tag{5.9}$$

e

$$\mathbf{K}_i^{-1} = \begin{bmatrix} 0,0573 & -0,0174 \\ -0,0174 & 0,0310 \end{bmatrix}, \tag{5.10}$$

com $t_i = 10$.

A figura 5.14 mostra as configurações inicial e final do robô sendo controlado com realimentação visual. Essas imagens foram extraídas da própria câmera de controle.

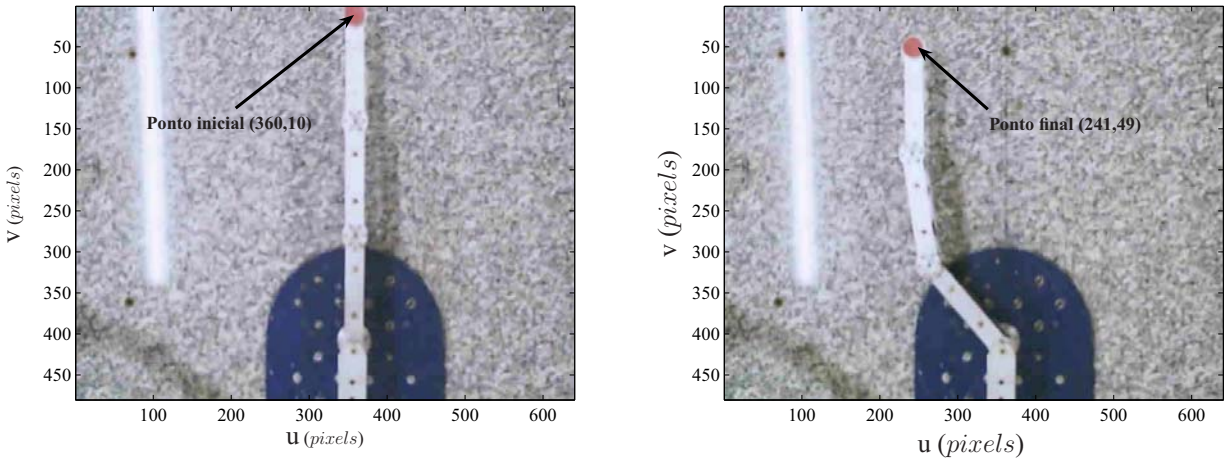


Fig. 5.14: Vista superior (câmera de controle) dos posicionamentos inicial e final do robô.

A figura 5.15 mostra a trajetória descrita pela garra do robô no plano de imagem, sendo que os

marcadores indicam as posições nas quais o sistema de controle recebeu a informação do sistema de visão sobre a posição da garra.

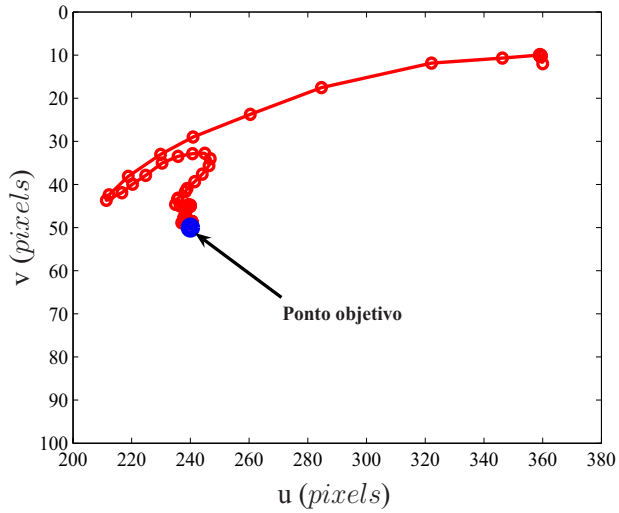


Fig. 5.15: Trajetória descrita pela garra do no plano de imagem

Enquanto que a figura 5.16 mostra a evolução individual de cada junta na direção do ponto de referência. Dessa figura pode-se notar que o sistema levou aproximadamente 20 s para estabilizar. Isso se deve à baixa taxa de quadros obtida no sistema de visão computacional ($\sim$ 10). O erro em regime permanente para essa tarefa foi de 1 $pixel$ em módulo mesmo não se conhecendo com precisão os parâmetros do robô e câmera.

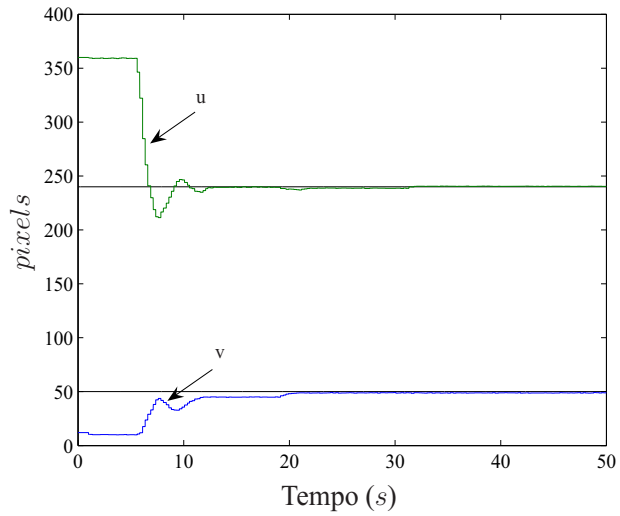


Fig. 5.16: Evolução individual das coordenadas da garra no plano de imagem

As referência corrigidas fornecidas à busca são mostradas na figura 5.17, indicando uma forte

atuação do sistema na correção dos parâmetros e controle da máquina.

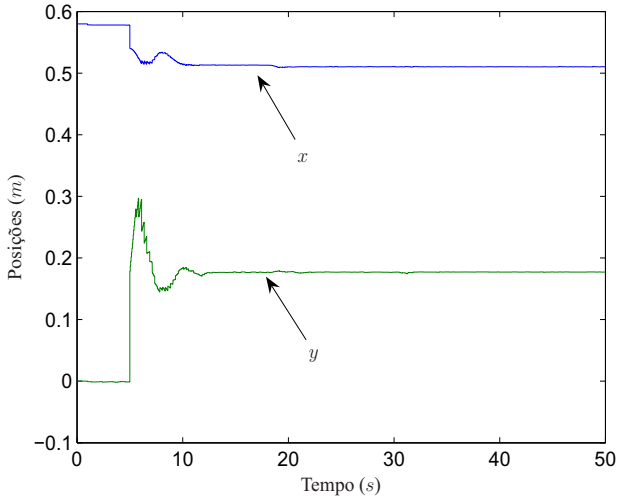


Fig. 5.17: Referências para a busca alteradas pelo sistema de realimentação visual

A título de ilustração, a figura 5.18 mostra as referências de junta geradas pelo algoritmo de busca (linhas pretas) e as respostas conseguidas no controle de juntas.

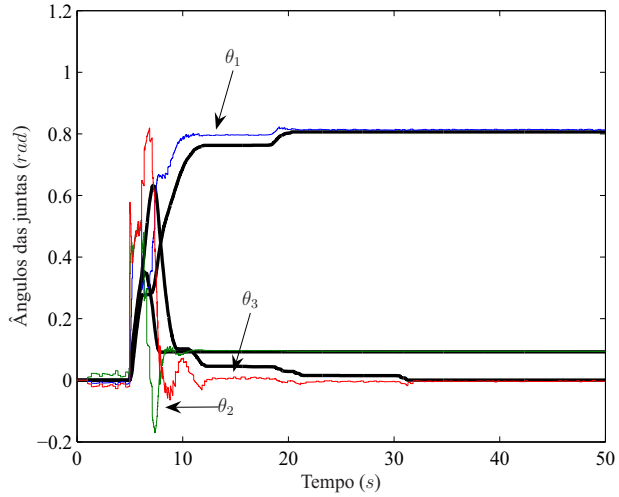


Fig. 5.18: Evolução das referências de juntas geradas pelo processo de busca e movimentações executadas por cada junta.

Capítulo 6

Conclusões e Perspectivas

Neste trabalho, foi estudada e proposta uma técnica alternativa para a solução do problema de cinemática inversa para manipuladores robóticos de cadeia serial aberta. O método visa resolver o problema em questão sem o uso de inversão de matrizes, linearizações, etapas de treinamento, adaptação de parâmetros etc. Tais intentos foram alcançados buscando-se a solução inversa apenas como o modelo direto, variando-se, numa etapa simulada, apenas uma junta de cada vez e determinado-se a melhor contribuição de cada junta para o solução final. Tal estratégia transforma o problema não linear n-dimensional em n problemas unidimensionais com solução analítica simples, cujas expressões gerais para modelos de cinemática direta baseados na formulação de Denavit-Hartenberg foram elaboradas neste estudo, tanto para posição quanto para orientação. A solução geral é obtida por um processo iterativo no qual cada junta participa de acordo em cada ponto rastreado com uma ponderação pre-definida. O método é convergente para quaisquer pontos a serem rastreado dentro do espaço de trabalho da máquina. O estudo da convergência foi elaborado pela teoria de busca em espaços de estados usando funções de controle de Lyapunov.

De forma a minimizar a influência das incertezas do modelo cinemático, a técnica foi expandida para incorporar informações de sensores externos que juntamente com a resolução da cinemática produzem um sinal de correção para a referência do processo de busca, possibilitando a correção das imprecisões nos parâmetros do modelo, tanto do robô quanto do sensor externo. Do ponto de vista puramente cinemático, esta modificação gera um sistema estável e convergente para uma precisão dentro do limite de tolerância estabelecido para a busca e pelo desempenho dos controladores de junta. Porém, do ponto de vista dinâmico, como o robô deixa de ser visto como um posicionador perfeito, o que é o caso na prática, os valores dos ganhos de correção e das incertezas no modelo do sensor externo influenciam no desempenho do sistema.

Nos experimentos de simulação, o método foi comparado com técnicas tradicionais de resolução de cinemática inversa e mostrou-se superior em aspectos como singularidades, manipulação de re-

dundâncias, repetibilidade, limitações nos movimentos das juntas, incerteza de parâmetros etc.

O resultados práticos ratificaram os resultados obtidos nas simulações, mostrando principalmente que a abordagem proposta é robusta a incerteza nos parâmetros cinemáticos do robô e também da câmera.

Dessa forma, poder-se-ia, por exemplo, adaptar uma máquina existente mantendo seu projeto de controladores original de forma com que ela tivesse seu desempenho geral melhorado, podendo inclusive trabalhar em ambientes menos controlados.

De modo a indicar linhas de continuidade do trabalho desenvolvido, são apresentadas a seguir perspectivas e sugestões de estudos que se mostraram pertinentes durante o desenvolvimento da tese e que ainda não foram abordados.

- Estudo do impacto da malha de realimentação externa na estabilidade do sistema de controle;
- Elaboração de um algoritmo para redimensionamento do tempo em tarefas que excedam a capacidade do robô;
- Estudo de técnicas de visão computacional mais avançadas para serem englobadas no sistema de busca;
- Utilização do método proposto para múltiplos robôs trabalhando em cooperação;
- Estudo de técnicas para otimizar o desempenho do método, possivelmente, por meio de técnicas como lógica nebulosa, algoritmos genéticos etc, alterando os pesos das contribuições das juntas;
- Estudo da aplicação do método para robôs tolerantes a falhas.
- Desenvolvimento de um sistema de visão computacional de alto desempenho para testes mais avançados da técnica;
- Desenvolvimento e elaboração de uma arquitetura de *hardware* para o sistema de busca.

Referências Bibliográficas

[1] L. Sciavicco e B. Siciliano. *Modeling and Control of Robot Manipulators*. McGraw-Hill, Inc., 1996. ISBN 0-07-114726-8. International Edition.

[2] Thomas R. Kurfess, editor. *Robotics and Automation Handbook*. CRC Press, 2004.

[3] J. M. Rosário. *Princípios de Mecatrônica*. Prentice Hall, Brasil., 2005.

[4] F. Chapelle e P. Bidaud. A closed form for inverse kinematics aproximation of general 6r manipulators using genetic programming. In *IEEE International Conference on Robotics and Automation*, pages 3364–3369, Seoul, Korea, May 2001.

[5] G. Antonelli e S. Chiaverini. A fuzzy approach to redundancy resolution for underwater vehicle-manipulator systems. *Control Engineering Practice*, 11(4):445–452, 2003.

[6] Y. Zhang, J. Wang, e Y. Xu. A dual neural network for bi-criteria kinematic control of redundant manipulators. *IEEE Transactions on Robotics and Automation*, 18(6):923–931, December 2002.

[7] M. K. Madrid e A. G. B. Palhares. Heuristic search method for continuous-path tracking optimization on high-performance industrial robots. *Control Engineering Practice*, 5(9):1261–1271, 1997.

[8] A. Campos, R. Guenther, e D. Martins. Differential kinematics of serial manipulators using virtual chains. *J. of the Braz. Soc. of Mech. Sci. and Eng.*, XXVII(4):345–356, 2005.

[9] K. Tchon. Repeatability of inverse kinematics algorithms for mobile manipulators. *IEEE Transactions on Automatic Control*, 47(8):1376–1380, 2002.

[10] Suguru Arimoto. A natural resolution of bernstein´s degrees-of-freedom problem in case of multi-joint reaching. In *IEEE International Conference on Robotics and Biomimetics*, volume CD proceedings, Shenyang, China, August 22-25 2004.

[11] H. Márquez. *Nonlinear Control Systems: Analysis and Design*. Wiley-Interscience, 2003.

[12] D. N. Nenchev, Y. Tsumaki, e M. Uchiyama. Real-time motion control in the neighborhood of singularities: a comparative study between the sc and the dls methods. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 506–511, Detroit, Michigan, 10-15 May 1999.

[13] F.B.M. Duarte e J.A.T. Machado. Chaos dynamics in the trajectory control of redundant manipulators. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, volume 4, pages 4109–4114, San Francisco, CA, USA, April 2000.

[14] G. Antonelli, S. Chiaverini, e G. Fusco. A new on-line algorithm for inverse kinematics of robot manipulators ensuring path tracking capability under joint limits. *IEEE Transactions on Robotics and Automation*, 19(1):162–167, February 2003.

[15] M. W. Spong e M. Vidyasagar. *Robot Dynamics and Control*. John Wiley and Sons, 1989.

[16] Y. H. Liu, H. Wang, C. Wang, e K. K. Lam. Uncalibrated visual servoing of robots using a depth-independent interaction matrix. *IEEE Transactions on Robotics*, 22(4):804–817, 2006.

[17] C. C. Cheah, M. Hirano, S. Kawamura, e S. Arimoto. Approximate jacobian robot control with uncertain kinematics and dynamics. In 692-702, editor, *IEEE Transactions on Robotics and Automation*, 4, 2003. 19.

[18] C. Cheah, C. Liu, e J. Slotine. Approximate jacobian adaptive control for robot manipulators. In *Proceedings of IEEE conference on robotics and automation*, pages 3075–3080, New Orleans, USA, 2004.

[19] A. R. L. Zachi, L. Hsu, R. Ortega, e F. Lizarralde. Dynamic control of uncertain manipulators through immersion and invariance adaptive visual servoing. *The Int. Journal of Robotics Research*, 25(11):1149–1159, 2006.

[20] W. E. Dixon. Adaptive path-constrained control of a robotic manipulator in a task space. *Robotica*, 25:43–61, 2006.

[21] W. E. Dixon. Adaptive regulation of amplitude limited robot manipulators with uncertain kinematics and dynamics. *IEEE Transactions on Automatic Control*, 52(3):488–493, 2007.

[22] B. Espiau, F. Chaumette, e P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–326, 1992.

[23] S. Hutchinson, G. Hager, e P. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, 1996.

[24] E. Malis e F. Chaumette. Theoretical improvements in the stability analysis of a new class of model-free visual servoimg methods. *IEEE Transactions on Robotics and Automation*, 18(2): 176–186, 2002.

[25] V. Lippiello, B. Siciliano, e L. Villani. Position-based visual servoing in industrial multi-robot cells using a hybrid camera configuration. *IEEE Transactions on Robotics*, 23(1):73–86, Feb. 2007.

[26] E. Malis, F. Chaumette, e S. Boudet. 2 1/2 d visual servoing. *IEEE Transactions on Robotics and Automation*, 15(2):238–250, 1999.

[27] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In D. Kriegman, G. Hager, e S. Morse, editors, *The confluence of Vision and Control*, volume 237, pages 66–78. Springer-Verlag, 1998.

[28] L. T. Wang e C. C. Chen. A combined optimization method for solving the inverse kinematics problem of mechanical manipulators. *IEEE Transactions on Robotics and Automation*, 7:489–499, 1991.

[29] M. K. Madrid e F. M. H. Maciá. Búsqueda heurística en tiempo real para la generación da trayectorias de robots manipuladores. In *V Conferencia de la Asociación Española para la Inteligencia Artificial*, pages 269–279, Madrid, 1993.

[30] Chris Welman. Inverse kinematics and geometric constraints for articulated figure manipulation. Master's thesis, Simon Fraser University, School of Computing Science, 1993.

[31] J. M. Ahuactzin e K. K. Gupta. The kinematic roadmap: a motion planning based global approach for inverse kinematics of redundant robots. *IEEE Transctions on Robotics and Automation*, 15(4):653–669, August 1999.

[32] D. L. Pieper. *The Kinematics of Manipulators Under Computer Control*. PhD thesis, Stanford University, Santanford, CA, 1969.

[33] C. Mavroidis, F. B. Quezdou, e P. Bidaud. Inverse kinematics of six degree of freedom general ans special manipulators using symbolic computation. *Robotica*, 1993.

[34] D. E. Whitney. Resolved motion rate control of manipulators and human prostheses. *IEEE Trans. Man-Mach. Syst.*, MMS-10:47–53, 1969.

[35] Y. Nakamura e H. Hanafusa. Inverse kinematics solution with singularities robustness for robot manipulator control. *J. Dynam. Syst., Meas., Contr.*, 108:163–171, 1986.

[36] C. W. Wampler. Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. *IEEE Trans. Syst., Man, Cybern.*, 16:93–101, January 1986.

[37] S. Chiaverini, B. Siciliano, e O. Egeland. Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator. *IEEE Transactions on Control Systems Technology*, 2(2):123–134, June 1994.

[38] Denny Oetomo, Marcelo H. Ang Jr, e Tao Ming Lim. Singularity robust manipulator control using virtual joints. In *IEEE International Conference on Robotics and Automation*, pages 2418–2423, Washington, DC, May 2002.

[39] Guangyu Lian, Qingjie Zhao, e Zengqi Sun. Path-constrained trajectory planning of robot arm passing through singularities. In *IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering - TENCON'02*, volume 3, pages 1558 – 1561, Beijing, China, October 2002.

[40] J. E. Lloyd. Removing the singularities of serial manipulators by transforming the workspace. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 2935–2940, Leuven, Belgium, 16-20 May 1998.

[41] A. Balestrino, G. De Maria, e L. Sciavicco. Robust control of robot manipulators. In *Proc. of the* 9^{th} *IFAC World Congress*, volume 6, pages 80–85, Budapest, Hungary, July 1984.

[42] W.A. Wolovich e H. Elliott. A computacional tecnique for inverse kinematics. In *Proc. of the* 23^nd *IEEE Conf. Decision and Control*, pages 1359–1363, Las Vegas, NV, December 1984.

[43] S. Chiacchio, S. Chiaverini, L. Sciavicco, e B. Siciliano. Closed-loop inverse kinematics schems for constrained redundant manipulators with task-space augmentation and task-priority strategy. *Int. J. Robot. Res.*, 10:410–425, 1991.

[44] F. Caccavale, S. Chiaverini, e B. Siciliano. Second-order kinematic control of robot manipulators with jacobian damped least-squares inverse: Theory and experiments. *IEEE/ASME Trans. Mechatron.*, 2:188–194, 1997.

[45] C.W. Wampler e L. J. Leifer. Aplications of damped least-squares methods to resolved-rate and resolved-acceleration control of manipulators. *J. Dynam. Syst., Meas. Contr.*, 110:31–38, 1988.

[46] I. A. Gravagne e I. D. Walker. On the structure of minimun effort solutions with application to kinematic redundancy resolution. *IEEE Trans. on Robotics and Automation*, 16(6):855–863, Dec. 2000.

[47] S. Chiaverini. Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 13(3):398–410, June 1997.

[48] J. Park, Y. Choi, W. K. Chung, e Y. Youm. Multiple task kinematics using weigted pseudo-inverse for kinematically redundant manipulators. In *IEEE International Conference on Robotics and Automation*, pages 4041–4047, Seoul, Korea, May 2001.

[49] Veljko Potkonjak, Dragan Kostic, Milan Rasic, e Goran S. Dordaevic. Motion in human and machine: A virtual fatigue approach. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, 32(5):582–595, 2002.

[50] Veljko Potkonjak, S. Tzafestas, D. Kostic, G. Djoudjevic, e M. Rasic. The handwriting problem. *IEEE Robotics and Automation Magazine*, pages 35–46, March 2003.

[51] A. D'Souza, S. Vijayakumar, e S. Schaal. Learning inverse kinematics. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 298–303, Maui,Hawaii, USA, 2001.

[52] Z. Wei. The inverse kinematics for the orientation of a robot arm based on neural network. *J. of Nanjing University of Aeronautics and Astronautics*, 29(1):46–50, 1997.

[53] W. S. Tang; J. Wang. A recurrent neural network for minimum infinity-norm kinematic control of redundant manipulators with an improved problem formulation and reduced architecture complexity. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 31(1):98–105, February 2001.

[54] Y. Xu e M. C. Nechyba. Fuzzy inverse kinematic mapping: Rule generation, efficiency and implementation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 911–918, Yokohama, Japan, July 1993.

[55] M. G. Her, C. Y. Chen, Y. C. Hung, e M. Karkoub. Approximating a robot inverse kinematics solution using fuzzy logic tuned by genetic algorithms. *The International Journal of Advanced Manufacturing Technology*, 20:375–380, 2002.

[56] A. Canutescu e R. L. Dunbrack. Cyclic coordinate descent: A robotics algorithm for protein loop closure. *Protein Science*, 12:963–972, 2003.

[57] Y. Shirai e H. Inoue. Guiding a robot by visual feedback in assembling tasks. *pattern Recognition*, 5:99–108, 1973.

[58] A. C. Sanderson e L. E. Weiss. Image-based visual servo control using relational graph error signals. *Proc. IEEE*, pages 1074–1077, 1980.

[59] Peter Corke e S. A. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 17(4):507–515, August 2001.

[60] G. Chesi, K. Hashimoto, D. Prattichizzo, e A. Vicino. Keeping features in the field of view in eye-in-hand visual servoing: A switching approach. *IEEE Transactions on Robotics and Automation*, 20(5):908–913, 2004.

[61] J. Pagès, C. Collewet, F. Chaumette, e J. Salvi. Optimizing plane-to-plane positioning tasks by image-based visual servoing and structured light. *IEEE Transactions on Robotics*, 22(5): 1000–1010, 2006.

[62] E. Malis. Visual servoing invariant to changes in camera intrinsic parameters. *IEEE Transactions on Robotics and Automation*, 20(1):72–81, 2004.

[63] M. R. Akella. Vision-based adaptive tracking control of uncertain robot manipulators. *IEEE Transactions on Robotics*, 24(4):747– 753, Aug 2005.

[64] J. Denavit e R. S. Hartenberg. A kinematic notation for lower pair mechanisms based on matrices. *ASME J. App. Mech.*, 77:215–221, 1955.

[65] R. Tsai. A versatile camera calibration technique for high accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Transactions on Robotics and Automation*, 4 (4):323–344, 1987.

[66] R. Lens e R. Tsai. A new technique for fully autonomous and efficient 3d robotics hand/eye calibration. *IEEE Transactions on Robotics and Automation*, 5(3):345–358, 1989.

[67] Paulo Jorge Sequeira Gonçalves. *Controlo Visual de Robôs Manipuladores*. PhD thesis, Universidade Técnica de Lisboa - Instituto Superior Técnico, 2005.

[68] J. Hill e J. T. Park. Real time control of a robot with a mobile camera. In *Proc. of the* 9^{th} *International Simposium on Industrial Robots*, pages 233–246, Washington, DC, USA, 1979.

[69] P.I. Corke. A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3(1): 24–32, March 1996.

[70] William J. Wilson, Carol C. Williams Hulls, e Graham S. Bell. Relative end-effector control using cartesian osition based visual servoing. *IEEE Transactions on Robotics and Automation*, 12(5):684–696, 1996.

[71] Benoit Thuilot, Philippe Martinet, Lionel Cordesses, e Jean Gallice. Position based visual servoing: keeping the object in the field of vision. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pages 1624– 1629, Washington, DC, USA, May 2002.

[72] G. Taylor e L. Kleeman. Hybrid position-based visual servoing with online calibration for a humanoid robot. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 686– 691, Sendai, Japan, October 2004.

[73] G. N. DeSouza e A. C. Kak. A subsumptive, hierarchical, and distributed vision-based architecture for smart robotics. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 34(5): 1988–2002, 2004.

[74] N. Gans, S. Hutchinson, e Peter Corke. Performance tests for visual servo control systems, with application to partitioned approaches to visual servo control. *Int. J. Robotics Research*, 22(10): 955–981, October 2003.

[75] D. DeMenthon e L. S. Davis. Model-based object pose in 25 lines of code. In *European Conference on Computer Vision*, pages 335–343, 1992.

[76] Y. Motai e A. Kak. An interactive framework for acquiring vision models of 3-d objects from 2-d images. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 34(1):566–578, 2004.

[77] V. Lippiello, B. Siciliano, e L. Villani. Adaptive extended kalman filtering for visual motion estimation of 3d objects. *Control Engineering Practice*, 15:123–134, 2007.

[78] R. Horaud. New methods for matching 3d objects with single perspective view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(3):401–412, 1987.

[79] L. F. Deng, F. Janabi-Sharifi, e W. J. Wilson. Hybrid motion control and planning strategies for visual servoing. *IEEE Transactions on Industrial Electronics*, 52(4):1024–1040, 2005.

[80] J. Pearl. *Heuristics - Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley Publishing Company, Inc., Mass., USA, 1984.

[81] S. J. Russell e P. Norvig. *Artiffitial Intelligence: A Moderne Approach*. Prentice Hall, Inc., New Jersey, 1995.

[82] T. J. Perkins. *Lyapunov Methods for Safe Intelligent Agent Design*. PhD thesis, University of Massachusetts Amherst, Department of Computer Science, 2002.

[83] R. E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42(2-3):189–211, 1990.

[84] T. J. Perkins e A. G. Barto. Heuristic search in infinite state spaces guided by lyapunov analysis. In *Seventeenth International Joint Conference on Artificial Intelligence*, pages 242–247, San Francisco, CA, 2001.

[85] F. Nicolato e M. K. Madrid. Trajectory following technique for serial-chain manipulators at singularities: an iterative search appoach. In *XV Congresso Brasileiro de Automática*, volume CD proceedings, Gramado, RS, Brazil, 21-24 September 2004.

[86] L. Zlajpah. Simulation of n-r planar manipulators. *Simulation Practice and Theory*, 6(3):305–321, 1998.

[87] Target xPC. xpc target, 2007. Disponível em: www.mathworks.com.

[88] Real Time WorkShop. Real time workshop, 2007. Disponível em: www.mathworks.com.

[89] A. M. Rigato. Desenvolvimento de hardware eletrônico para aplicações em robótica usando eletrônica reconfigurável (fpga). Iniciação Científica, 2003. PIBIC/CNPq.

[90] Luiz Eduardo Filho Guardia. Sistema para controle de máquinas robotizadas utilizando dispositivos lógicos programaveis. Master's thesis, Faculdade de Engenharia Elétrica e de Computação - Universidade Estadual de Campinas, 2005.

[91] Fabrício Nicolato. Arquitetura de hardware para a extração em tempo real de características de múltiplos objetos em imagens de vídeo: Classificação de cores e localização de centróides. Master's thesis, Faculdade de Engenharia Elétrica e de Computação - UNICAMP, 2002.

[92] RS232 BlockSet. Rs232 blockset, 2007. Disponível em: http://digilander.libero.it/LeoDaga/Simulink/RS232Blockset.htm.

[93] F. Nicolato e M.K. Madrid. *Recursive algorithm for the inverse kinematics of redundant robotic manipulators*. Proceedings of 16th IFAC World Congress, 2005.

Apêndice A

Aspectos Construtivos do Ambiente Experimental

São apresentadas e descritas as partes básicas do ambiente experimental desenvolvido. É apresentada uma visão geral da construção, implementação do sistema

O robô desenvolvido é formado por três elos de alumínio de $0,2\ m$ de comprimento cada, figura A.5. A transmissão dos movimentos dos motores, que ficam sob a base do robô, é feita por polias e correias sincronizadas de forma a minimizar folgas nos movimentos das juntas.

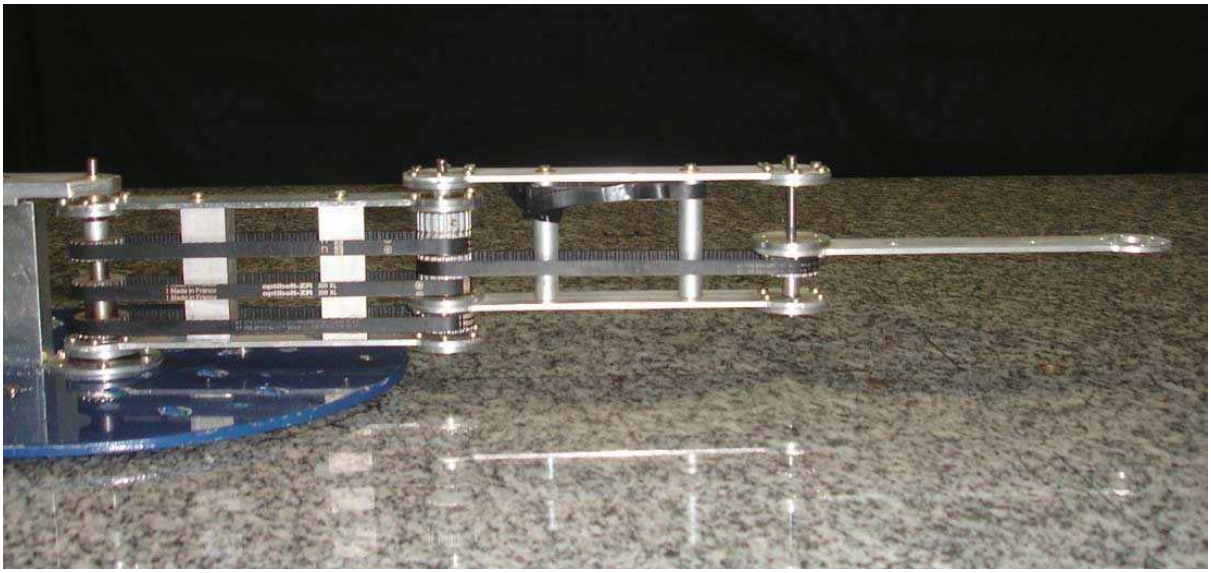


Fig. A.1: Foto do robô protótipo.

Cada junta é acionada por um motor de corrente contínua que é acoplados ao robô por meio de uma caixa de redução, veja a figura A.2. Cada motor do robô tem acoplado ao seu eixo um *encoder*

incremental de 500 pulsos por rotação.

Fig. A.2: Motores acoplados à estrutura mecânica do robô.

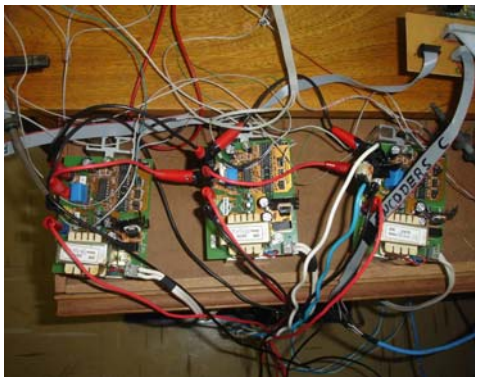


Fig. A.3: Base de acionamento dos motores.

O acionamento dos motores é feito pelos circuitos de potência mostrada na figura A.3, que implementa cada um, uma ponte H [1].

A geração dos sinais PWM, assim como a contagem dos pulsos dos *encoders* é feita por meio de circuitos eletrônicos, que geram uma precisão de 16 $bits$ para o ciclo de trabalho do PWM como uma freqüência de chaveamento de aproximadamente 6 KHz, e uma resolução de 16 bits para a leitura dos encoders. Com o efeito provocado pela redução mais o circuito de contagem, cada encoder é capaz de gerar 36000 a cada $\pi\ rad$ de ângulo nos eixos dos motores. Esses circuitos e também todo o esquema

[1]Estas placas foram desenvolvidas pela equipe do Prof. Antenor Pomílio do Laboratório de Condicionamento de Energia - DSCE/FEE/Unicamp - e foram fornecidos como cortesia para esse projeto

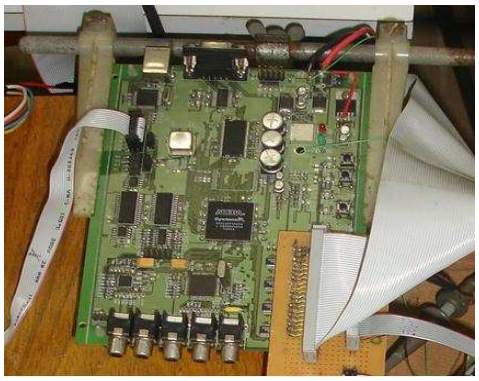


Fig. A.4: Placa de processamento e controle baseada em PLD.

de comunicação com o computador target foi implementado na placa de processamento baseada em PLD mostrada na figura A.4.

A associação de de eixos coordenados para o robô é mostrada na figura A.5. Por questões de construção mecânica, os ângulos das juntas medidos pelos *encoders* são sempre em relação ao eixo x_0. Com isso, a os parâmetros DH desse robô ficam conforme mostrado na tabela A.1. Isso resulta numa

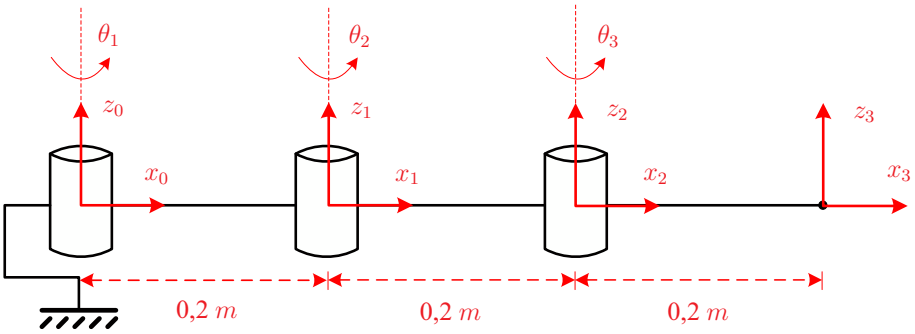


Fig. A.5: Associação de eixos coordenados.

expressão para a cinemática direta pela equação A.1, da qual pode-se notar que o robô é redundante para posição e não redundante para orientação, pois a orientação é dada sempre por θ_3 que, por sua vez, é absoluto em relação ao eixo x_0.

$$
{}^{0}\mathbf{T}_3 = \begin{bmatrix} \cos\theta_3 & -\operatorname{sen}\theta_3 & 0 & a_1\cos\theta_1 + a_2\cos\theta_2 + a_3\cos\theta_3 \\ \operatorname{sen}\theta_3 & \cos\theta_3 & 0 & a_1\operatorname{sen}\theta_1 + a_2\operatorname{sen}\theta_2 + a_3\operatorname{sen}\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.1}
$$

Da forma com que o sistema está montado, dois tipos de sistema de visão podem ser adicionados

Elo	$a_i\ (m)$	$\alpha_i\ (rad)$	$d_i\ (m)$	$\theta_i\ (rad)$
1	0,2	0	0	θ_1
2	0,2	0	0	$\theta_2 - \theta_1$
3	0,2	0	0	$\theta_3 - \theta_2$

Tab. A.1: Parâmetros dos elos para o robô planar de 3 DOF.

externamente ao robô de modo, por exemplo, a informar a posição da garra e/ou outras partes do robô, além de possíveis obstáculos. O primeiro modo, que não foi usado no trabalho, é usar uma câmera analógica ligada diretamente na placa de processamento que possui um decodificador de video nos formatos NTSC e/ou PAL-M e conseqüente conversão desse sinal para digital. O segundo modo consiste em se ligar uma câmera digital diretamente à uma porta paralela do computador *host* e realizar o processamento das imagens usando uma outra instância do MatLab trabalhando no modo de simulação acelerada (compilada) e comunicando-se com o *target* via $RS232$. Este foi o esquema usado para implementar o sistema de classificação de cor e rastreamento de centróide usados no trabalho. Como resultado, a taxa de quadros por segundo ficou em torno de 10. Vale ainda salientar que o host trabalha baseado no sistema operacional *windows* o que implica que não há garantia de processamento em tempo real.