Universidade Federal do Rio de Janeiro
Escola Politécnica

ESTUDO SOBRE UMA ESTRATÉGIA DE CONTROLE ADAPTATIVO SERVO VISUAL E SUA APLICAÇÃO EM UM ROBÔ INDUSTRIAL

LUCAS CARVALHO DE LIMA

Projeto de Graduação apresentado ao Curso de Engenharia de Controle e Automação da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientador:
Prof. Fernando Cesar Lizarralde, D.Sc.

Rio de Janeiro, RJ - Brasil

Março de 2013

ESTUDO SOBRE UMA ESTRATÉGIA DE CONTROLE ADAPTATIVO SERVO VISUAL E SUA APLICAÇÃO EM UM ROBÔ INDUSTRIAL

LUCAS CARVALHO DE LIMA

PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE CONTROLE E AUTOMAÇÃO

Aprovado por:

Prof. Fernando Cesar Lizarralde, D.Sc.

Prof. Alessandro Jacoud Peixoto, D.Sc.

Prof. Antonio Candea Leite, D.Sc.

Rio de Janeiro, RJ - Brasil

Março de 2013

Lima, Lucas Carvalho.
Estudo sobre uma estratégia de controle adaptativo servo visual e sua aplicação em um robô industrial
115 páginas
Projeto de Graduação UFRJ/ Escola Politécnica/Curso de Engenharia de Controle e Automação, 2013.
Orientador:
Prof. Fernando Cesar Lizarralde, D.Sc.

1. Servo Visão
2. Controle Adaptativo
3. Robótica

I. Lima, Lucas Carvalho. II. Universidade Federal do Rio de Janeiro, Escola Politécnica, Curso de Engenharia de Controle e Automação. III. Estudo sobre uma estratégia de controle adaptativo servo visual e sua aplicação em um robô industrial.

Agradecimentos

Gostaria de agradecer primeiramente a Deus, em seguida gostaria de agradecer a todos que contribuíram diretamente ou indiretamente para que este trabalho pudesse ter sido concluído. Em especial, aos meus familiares e amigos da turma de engenharia de controle e automação do ano de 2008.

Resumo do Projeto de Graduação apresentado à Escola Politécnica/ UFRJ como parte dos requisitos necessários para a obtenção do grau de Engenheiro de Controle e Automação.

ESTUDO SOBRE UMA ESTRATÉGIA DE CONTROLE ADAPTATIVO SERVO VISUAL E SUA APLICAÇÃO EM UM ROBÔ INDUSTRIAL

Lucas Carvalho de Lima

Março/2013

Orientador: Prof. Fernando Cesar Lizarralde, D.Sc.

Curso: Engenharia de Controle e Automação

Este trabalho apresenta um estudo sobre uma estratégia de controle adaptativo servo visual e a descrição de sua posterior implementação prática em um manipulador industrial, utilizando três graus de liberdade. A partir de uma câmera fixa não calibrada são obtidas as informações de posição utilizadas na estratégia de controle. O objetivo consiste em rastrear trajetórias de referência tridimensionais definidas na própria imagem. O movimento do manipulador é particionado em um movimento 2D em um plano ortogonal ao eixo da câmera e um movimento 1D paralelo ao mesmo eixo. Um esquema adaptativo servo visual utilizando uma abordagem cinemática, além de lidar com as incertezas paramétricas da câmera não calibrada, permite os rastreamentos em profundidade e planar, sem a necessidade de estimar as medidas de profundidade e velocidades da imagem. A convergência assintótica para zero dos erros de rastreamento planar e em profundidade são comprovados por uma abordagem baseada na teoria de Lyapunov e posteriormente os resultados teóricos são comprovados a partir de simulações e experimentos em um robô industrial real.

Palavras Chaves: Servo Visão, Controle Adaptativo, Robótica

Abstract of Undergraduate Project presented to POLI/UFRJ as a partial fulfillment of the requirements for the degree of Engineer.

STUDY OF AN VISUAL SERVOING ADAPTIVE CONTROL STRATEGY AND ITS APPLICATION IN A INDUSTRIAL ROBOT

Lucas Carvalho de Lima

March/2013

Advisor:
Prof. Fernando Cesar Lizarralde, D.Sc.
Course: Engenharia de Controle e Automação

This work presents an study about an visual servoing adaptive control strategy and an description of its practical implementation in a industrial manipulator, using three degrees of freedom. Position informations are obtained from an uncalibrated fixed camera and they are used in an control strategy. Therefore the strategy objective will be tracking 3D reference trajetories that are defined in an image. The robot motion will be decomposed into a 2D motion in a plan orthogonal to the camera optical axis and a 1D motion in a axis that is paralell to this axis. An adaptive visual servoing scheme based on kinematic control besides coping with uncertainties in parameters of an uncalibrated camera will provide depth and planar tracking without measuring depth variations and image velocity. Asymptotic convergence of planar and depth tracking errors to zero are proved by an approach based on Lyapunov theory and after the convergence is proved by simulations and experimental results in a real industrial robot.
Keywords: Visual Servoing, Adaptive Control, Robotics

Conteúdo

RESUMO **4**

ABSTRACT **5**

LISTA DE FIGURAS **iv**

LISTA DE ABREVIATURAS **vi**

1 Introdução **1**

1.1 Servo Visão 2

1.1.1 Controle Servo Visual Baseado em Imagem 3

1.1.2 Controle Servo Visual Baseado em Posição 4

1.1.3 Controle Servo Visual Híbrido 5

1.1.4 Novas abordagens e métodos em Controle Servo Visual 6

1.2 Objetivos 7

1.3 Motivação 8

1.4 Organização do trabalho 9

2 Servo Visão Robótica **10**

2.1 Formulação do problema 10

2.2 Modelo Servo Visual 12

2.2.1 Manipulador Antropomórfico (3 DOF's) 12

2.2.2 Sistema de Coordenadas e Cinemática Diferencial 13

2.2.3 Modelo de Projeção de Câmeras 16

2.2.4 Modelo do Sistema Servo Visual Baseado em Imagem (IBVS) 17

2.3 Controle Adaptativo 19
2.3.1 Conceitos Básicos 20
2.3.2 Controle Adaptativo por Modelo de Referência (MRAC) 22
2.3.3 MRAC Servo Visual 26
2.4 Resultados de Simulações 45

3 Implementação Prática 57
3.1 Sistema de Controle Servo Visual 57
3.1.1 Visão Computacional 57
3.1.2 Componentes do Sistema de Controle Servo Visual 65
3.2 Interface Gráfica de Controle 71
3.2.1 Operação da Interface Gráfica 73
3.3 Resultados 75

4 Conclusões 79
4.1 Trabalhos Futuros 81

Referências Bibliográficas 82

A Manual de Operação da Plataforma Motoman DIA10 84
A.1 A Plataforma Motoman DIA10 84
A.1.1 Os Manipuladores 84
A.1.2 Controlador NX100 88
A.1.3 Programming Pendant 88
A.1.4 Modos de Operação 92
A.1.5 High-speed Synchronous Controller (HSC) 94
A.1.6 MotomanSimulinkHost 96
A.1.7 Integração dos sistemas 97
A.1.8 Operação em modo *Teach* - Passo a passo 98
A.1.9 Operação por Computador Usuário - Passo a passo 99

B Prova da positividade real estrita (SPR) para a função de transferência do erro aumentado 103

Lista de Figuras

1.1 (a) Ilustração do sistema eye-in-hand. (b) Ilustração do sistema eye-to-hand . . 2

2.1 Representação simplificada de um Sistema Servo Visual 11

2.2 Manipulador Antropomórfico com 3 DOF's 13

2.3 Representação dos ângulos das juntas do manipulador 15

2.4 Representação das medidas do manipulador 16

2.5 Diagrama de blocos básico do Controle Adaptativo 21

2.6 Diagrama de blocos básico do MRAC 22

2.7 Gráfico das trajetórias real e de referência para o rastreamento planar, obtido em simulação com ganhos de adaptação $\gamma_1 = 4.0$, $\gamma_2 = 60.0$ 47

2.8 Gráfico do erro de rastreamento planar e em profundidade obtido em simulação com ganhos de adaptação $\gamma_1 = 4.0$, $\gamma_2 = 60.0$, $\gamma_3 = 10^6$ 48

2.9 Gráfico da estimação dos parâmetros Θ_1 e Θ_2 desconhecidos, obtido em simulação com ganhos de adaptação $\gamma_1 = 4.0$, $\gamma_2 = 60.0$ 49

2.10 Gráfico da estimação do parâmetro θ_3 desconhecido, obtido em simulação com ganho de adaptação $\gamma_3 = 10^6$ 50

2.11 Gráfico das trajetórias real e de referência para o rastreamento planar, obtido em simulação com ganhos de adaptação $\gamma_1 = 0.5$, $\gamma_2 = 60.0$ 52

2.12 Gráfico do erro de rastreamento planar e em profundidade, obtido em simulação com ganhos de adaptação $\gamma_1 = 0.5$, $\gamma_2 = 60.0$, $\gamma_3 = 10^{10}$ 53

2.13 Gráfico da estimação dos parâmetros Θ_1 e Θ_2 desconhecidos, obtido em simulação com ganhos de adaptação $\gamma_1 = 0.5$, $\gamma_2 = 60.0$ 54

2.14 Gráfico da estimação do parâmetro θ_3 desconhecido, obtido em simulação com ganho de adaptação $\gamma_3 = 10^{10}$ 55

3.1 Representação do sistema de coordenadas O_f relacionado a matriz da imagem capturada e o sistema de coordenadas da projeção da imagem O_v 63
3.2 Ilustração do sistema robótico Motoman DIA10 67
3.3 Diagrama representativo com as juntas utilizadas na implementação prática . . 68
3.4 Interface Gráfica (GUI) desenvolvida para a simulação e o teste experimental do controle servo visual 73
3.5 Gráfico das trajetórias real e de referência para o rastreamento planar, obtido em teste experimental com ganhos de adaptação $\gamma_1 = 0.8$, $\gamma_2 = 0.8$ 76
3.6 Gráfico do erro de rastreamento planar, obtido em teste experimental com ganhos de adaptação $\gamma_1 = 0.8$, $\gamma_2 = 0.8$ 77
3.7 Gráfico da estimação dos parâmetros Θ_1 e Θ_2 desconhecidos, obtido em teste experimental com ganhos de adaptação $\gamma_1 = 0.8$, $\gamma_2 = 0.8$ 78

A.1 Sistema robótico Motoman DIA10 85
A.2 Diagrama representativo das juntas, nomes e respectivos eixos de rotação do Motoman DIA10 86
A.3 Diagrama representativo com as juntas e medidas de interesse 87
A.4 Diagrama básico das conexões em uma operação através do *Programming Pendant* 88
A.5 Controlador Manual *Programming Pendant* - Visão Geral 91
A.6 Controlador Manual *Programming Pendant* - Imagem ampliada dos botões relevantes 92
A.7 Diagrama básico das conexões em uma operação com HSC 94
A.8 Diagrama do processo interno de uma operação com HSC 95
A.9 Imagem mostrada no monitor do HSC, indicando a comunicação bem sucedida com o computador usuário 100

Lista de abreviaturas

IBVS - Controle Servo Visual Baseado em Imagem

PBVS - Controle Servo Visual Baseado em Posição

DOF - *Degree Of Freedom* (Grau de liberdade)

MIMO - *Multiple Inputs / Multiple Outputs* (Múltiplas entradas / Multiplas saídas)

SISO - *Single Inputs / Single Outputs* (Uma entrada / uma saída)

BIBO - *Bounded Input / Bounded Output* (Entrada limitada / Saída limitada)

MRAC - *Model Reference Adaptive Control* (Controle por Modelo de Referência)

I/O - *Inputs/Outputs* (Entradas/Saídas)

1D - Monodimensional

2D - Bidimensional

3D - Tridimensional

HD - *High Definition* (Alta Definição)

Capítulo 1

Introdução

O desenvolvimento de áreas como a robótica e visão computacional, vêm fomentando, por anos, pesquisas que apresentam como foco a integração de ambos conceitos, permitindo desta maneira o surgimento de novos sistemas com a mais elevada tecnologia.

A robótica é um ramo de estudos multidisciplinar que envolve áreas como Engenharia Mecânica, Elétrica, Eletrônica, Computação, Física, Matemática e Biologia. Ao integrar os campos de estudo mencionados, tornou-se possível desenvolver a robótica a tal ponto em que é possível aplicá-la nas mais diversas áreas.

Um sistema robótico consiste no desenvolvimento de tecnologia que apresenta, dentre outros, componentes principais como: sistema energético, sensores, sistema de controle e atuadores.

Os avanços da engenharia Robótica que observamos hoje, são incentivados em grande parte pela sua flexibilidade em aplicações e pelos benefícios que a sua introdução oferece.

A visão computacional é a ciência que procura desenvolver algoritmos e tecnologias, através dos quais sistemas computacionais são capazes de interpretar imagens ou quaisquer outros dados multidimensionais e extrair informações em um determinado contexto. A visão computacional integra áreas como processamento de imagens e inteligência artificial, podendo ser aplicada em alguns campos do conhecimento como agronomia, astronomia, biologia, biometria, medicina e robótica.

Atualmente, existe um grande número de pesquisas que se ocupam da integração dos conceitos de robótica e visão computacional. Fruto destes estudos, emergiu o que denotamos Controle Servo Visual ou na língua inglesa *Visual Servoing Control*.

1.1 Servo Visão

O conceito básico do Controle por Servo Visão consiste em utilizar dados da Visão Computacional para controlar o movimento de um sistema robótico. De forma geral, o aprendizado, planejamento, percepção e a ação, são as principais atividades que o sistema servo visual deverá executar.

A maior parcela dos sistemas em servo visão são compostos por uma ou mais câmeras que fornecem os dados visuais e um manipulador robótico com um conjunto definido de graus de liberdade (DOF). As abordagens servo visuais utilizam, principalmente, duas configurações distintas para o posicionamento das câmeras [2]: posição "Olho na mão"(*Eye in hand*) e "Olho para mão"(*Eye to hand*).

Na configuração *eye in hand* a câmera responsável pela coleta de dados é posicionada na extremidade do braço robótico que ao se mover, induz um movimento na câmera. Já a montagem *eye to hand* apresenta uma câmera estacionária montada no espaço de trabalho que captura o movimento do braço robótico. As duas configurações podem ser observadas na figura 1.1.

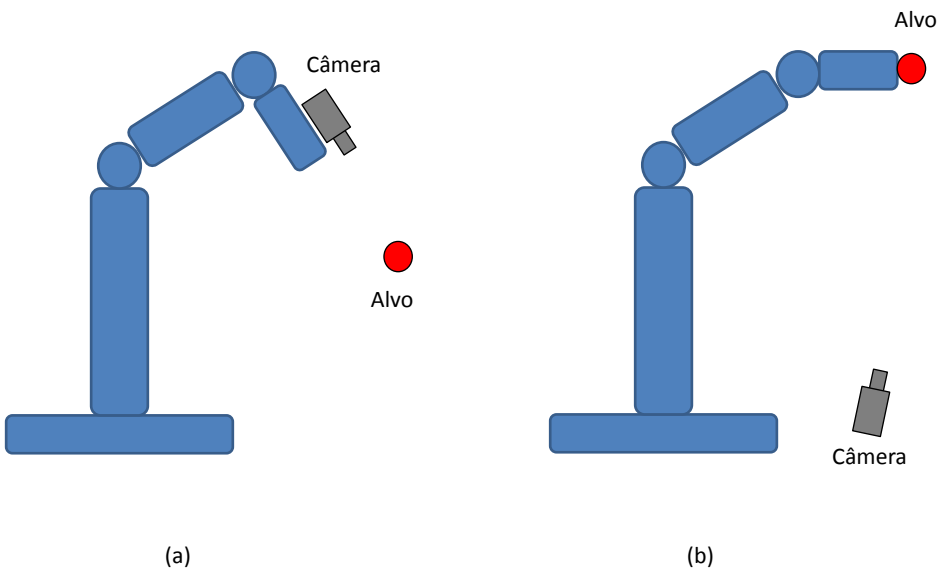

Figura 1.1: (a) Ilustração do sistema eye-in-hand. (b) Ilustração do sistema eye-to-hand

Assim como na maioria das estratégias de controle, o objetivo, a essência do Controle Servo Visual consiste na minimização do erro $e(t)$ que é definido como:

$$e(t) = s(m(t), a) - s^* \tag{1.1}$$

A equação (1.1) é uma generalização conceitual que pode ser utilizada para os mais diversos tipos de abordagem e aplicações em Controle Servo Visual. O parâmetro m é um vetor que armazena medidas da imagem e é utilizado para construção do vetor $s(m(t), a)$ que possui um conjunto de características da imagem. A forma como obtêm-se o parâmetro s está diretamente ligada ao tipo de abordagem e metodologia utilizada na construção do Controle Servo Visual. Já a variável a representa informações adicionais do sistema que poderiam ser previamente fornecidas, como o formato do objeto utilizado como alvo ou a distância focal da câmera utilizada. Por fim, s^* consiste no valor desejado do vetor de características da imagem.

Atualmente, o Controle por Servo Visão é foco de diversas pesquisas para solucionar o problema de rastreamento de objetos, controlando-se o movimento de sistemas robóticos como manipuladores industriais. Deste modo, existem três abordagens clássicas [2] utilizadas no desenvolvimento do Controle Servo Visual, das quais podemos destacar:

- Controle Servo Visual Baseado em Imagem (IBVS)
- Controle Servo Visual Baseado em Posição (PBVS)
- Controle Servo Visual Híbrido

A existência de restrições inerentes às propostas clássicas incentivou o surgimento de inúmeras outras metodologias, objetivando minimizar tais limitações relacionadas ao problema de rastreamento utilizando o Controle Servo Visual. Deste modo, ampliou-se a capacidade e as possibilidades de aplicações para os Sistemas Servo Visuais.

1.1.1 Controle Servo Visual Baseado em Imagem

Nesta metodologia a definição do parâmetro s advém de informações diretamente fornecidas da imagem capturada. Desta maneira, usualmente são utilizados as coordenadas de um conjunto de pontos do plano da imagem para a construção do vetor s.

Posteriormente, o IBVS define a matriz de interação L_s que também é conhecida como Jacobiano da imagem, a partir da qual será calculada a pseudo-inversa estimada, denotada $\hat{L_s}$. A estimação é necessária, pois na prática se torna impossível conhecer completamente a matriz de interação, assim a lei de controle será baseada na matriz pseudo inversa estimada, no erro $e(t)$ e na utilização de um ganho proporcional. Em [2] a estabilidade assintótica local do erro $e(t)$ é provada a partir da teoria de Lyapunov.

Recentes pesquisas em IBVS demonstraram resultados relevantes, comprovando sua eficácia, como é o caso da trabalho realizado em [4]. No estudo referido, um esquema de controle sem calibração foi desenvolvido para guiar um robô de solda industrial com 6 DOF's para rastreamento de alvo via IBVS. A lei de controle em malha fechada proposta, é baseada em informações visuais obtidas de uma câmera em configuração *eye in hand* e sua implementação permite que o robô rastreie ou seja guiado sem o conhecimento total do modelo cinemático do robô ou da calibração da câmera. A análise de estabilidade foi efetuada com base na teoria de Lyapunov e simulações efetuadas com um robô 6 DOF's, comprovaram a capacidade de rastreamento de objetos e a estabilidade assintótica do sistema controlado.

Apesar de apresentar resultados consistentes, o IBVS possui algumas restrições que devem ser destacadas. A primeira delas é a necessidade de estimar ou aproximar o valor da profundidade de um determinado ponto em relação ao sistema de coordenadas da câmera. Ressalta-se ainda, a possibilidade da matriz de interação se tornar singular para algumas configurações, comprometendo assim a estabilidade do sistema. Outra restrição consiste na estabilidade assintótica local que garante a convergência do erro $e(t)$ para zero somente para pequenas diferenças entre o vetor de características da imagem desejado e o atual. Por fim, o IBVS quando aplicado, apresenta um inesperado movimento translacional de recuo na direção do eixo ótico da câmera, comprometendo assim a movimentação do sistema robótico.

1.1.2 Controle Servo Visual Baseado em Posição

A metodologia PBVS consiste em efetuar uma série de medidas a partir da imagem capturada, a fim de estimar um conjunto de parâmetros 3D relacionados a posição da câmera. Desta maneira, a parametrização desta posição em relação a um referencial global será representada

na variável definida como s.

Para o cálculo da posição é necessário o conhecimento de parâmetros intrínsecos da própria câmera e valores do modelo 3D do objeto a ser observado. Parâmetros estes que serão agora representados na variável a.

Da mesma maneira, assim como ocorreu no IBVS, uma nova matriz de interação será calculada e a estimativa desta matriz será utilizada para definição da lei de controle baseada também em um ganho proporcional e no erro $e(t)$. Assim como no método anterior, a estabilidade assintótica local do erro é comprovada a partir da teoria de Lyapunov.

O PBVS também apresenta algumas restrições que devem ser ressaltadas. A lei de controle utilizada apresenta quantidades estimadas que são função da calibração de parâmetros do sistema, desta maneira pequenos erros de estimação podem levar a erros de reconstrução que impactam em grande nível a acurácia do sistema após a convergência.

1.1.3 Controle Servo Visual Híbrido

A abordagem híbrida do controle servo visual consiste em unir boas características do IBVS e PBVS para produzir uma nova metodologia com maior acurácia e que minimize as restrições.

Uma das estratégias desenvolvidas é denominada 2 e 1/2D Visual Servo, esta técnica utiliza uma câmera em configuração *eye in hand* e consiste basicamente em estimar o parâmetro que define o ângulo de rotação da câmera, em seguida é desenvolvida uma lei de controle com duas parcelas: uma consiste no controle rotacional e a outra é o controle translacional, assim como aquele desenvolvido no IBVS. Desta maneira o controle é particionado, combinando-se o PBVS e o IBVS. Ressalta-se que esta metodologia, em virtude de suas características e dos resultados obtidos por simulação [3], se aproxima do método PBVS e em perfeitas condições a estabilidade assintótica global é assegurada. Em alguns casos considera-se a presença de erros de calibração e a estabilidade pode ser analisada abordando técnicas de estimação.

Outros esquemas híbridos também foram desenvolvidos de maneira que os graus de liberdade translacionais da câmera são utilizados para realizar movimentos 3D em linha reta, enquanto seus graus de liberdade rotacionais são utilizados para realizar movimentos 2D de

forma a compensar o movimento 2D do centro de gravidade do alvo na imagem. Por fim, algumas abordagens híbridas combinam características 2D e 3D do sistema para produzir a estratégia de controle.

1.1.4 Novas abordagens e métodos em Controle Servo Visual

As pesquisas na área de Servo Visão, permitiram o surgimento de uma série de metodologias de Controle Servo Visual inovadoras com diferentes abordagens e aplicações de novos conceitos.

Um dos problemas enfrentados pelos estudiosos do Controle Servo Visual é o desconhecimento das medidas de profundidade nas informações obtidas da imagem. Em [11] é proposto um Controle IBVS Adaptativo em que a matriz de interação é independente da informação de profundidade, consequentemente as coordenadas 3D desconhecidas aparecerão de forma linear no sistema dinâmico em malha fechada. O controle adaptativo empregado estima, durante a execução da malha de controle, os parâmetros 3D desconhecidos e em conjunto com um controlador proporcional e derivativo clássico, permite ao manipulador efetuar o rastreamento do alvo. A análise de estabilidade é produzida com base na teoria de Lyapunov e experimentos práticos em um robô com três graus de liberdade exemplificam e comprovam a convergência assintótica do erro entre referência e a saída do sistema.

Recentes estudos [16] desenvolveram um observador de estados [7] não linear para a estimação da trajetória desconhecida de alvos e assim permitir o rastreamento. A estratégia de controle em questão foi desenvolvida para lidar com o movimento 3D desconhecido de um alvo e com as não linearidades da dinâmica de um manipulador com 3 DOF's. Ao utilizar a matriz de interação independente da informação de profundidade, os parâmetros 3D desconhecidos aparecem linearmente na equação dinâmica, deste modo o observador de estados é desenvolvido para estimar , durante a execução da malha de controle, a trajetória 3D desconhecida do alvo e para compor o controle dinâmico. O controle dinâmico utiliza esta estimativa do observador de estados em sua construção de forma a compensar forças de inércia, forças não-lineares centrífugas de Coriolis e a força gravitacional, além de compensar o erro de referência entre a posição atual do alvo na imagem e a posição desejada. A convergência para zero dos erros de

referência na imagem e da estimação dos parâmetros desconhecidos é demonstrada à partir da teoria de Lyapunov e experimentos em um manipulador com 3 DOF's comprovam a eficácia da estratégia utilizada.

Outros autores estudaram alternativas à utilização de estimativas da informação de profundidade necessária ao método IBVS como pode ser observado nos trabalhos [17] e [10].

Em [10], desenvolveu-se um controle adaptativo IBVS que permite o rastreamento em 3D de trajetórias em superfícies suaves sem a necessidade de obter medidas de profundidade ou velocidades na imagem e sob a presença de incertezas paramétricas tanto no robô quanto na câmera. No estudo em questão utilizou-se apenas uma câmera estática para visualização do manipulador, o que caracteriza a abordagem *eye to hand* e o objetivo consistia em criar uma estratégia de controle menos sensível à falta de informações sobre parâmetros do sistema ou ambiente. A modelagem do movimento robótico foi decomposto em uma parcela 2D e 1D, que representam a locomoção em um plano ortogonal ao eixo ótico da câmera e a variação de profundidade paralela ao eixo ótico da câmera, respectivamente. Desta maneira, a ideia principal do estudo consiste em utilizar o centroide da imagem para prover o rastreamento planar em 2D e a área da imagem para o rastreamento em profundidade 1D.

A pesquisa sobre o Sistema Servo Visual utilizando controle adaptativo efetuada em [10] foi a base para o estudo desenvolvido neste documento.

1.2 Objetivos

O objetivo deste trabalho consiste em efetuar um estudo sobre uma proposta de controle adaptativo servo visual para rastreamento de trajetórias, desenvolvido em [10]. Entretanto, para este trabalho será implementado apenas a estratégia controle adaptativo cinemático, desconsiderando as incertezas cinemáticas e dinâmicas do robô. Inicialmente será efetuada uma descrição teórica da estratégia de controle e a convergência dos erros de rastreamento será comprovada com base em uma abordagem utilizando a teoria de Lyapunov. Posteriormente, o comportamento da estratégia de controle será analisado em uma primeira instância através de simulações, utilizando o software Matlab © e em seguida, a avaliação será efetuada a partir de experimentos em um ambiente real dispondo de uma câmera web HD e da plataforma Motoman DIA10

©.

1.3 Motivação

As técnicas de controle servo visual estão se tornando foco de estudo e dedicação para diversos pesquisadores da área.

O interesse em se criar sistemas robóticos com mais autonomia e consequentemente menor dependência da ação humana, têm direcionado grande parte das pesquisas como em ([1]) para a aplicação de metodologias de controle que utilizam informações obtidas de imagens para direcionar a locomoção de sistemas robóticos.

As técnicas de Controle Servo Visuais em sua grande maioria são avaliadas através de simulações e por vezes através de experimentos reais baseados em manipuladores robóticos industriais como ocorre nos trabalhos ([16], [8]). Este fato se deve a larga utilização de manipuladores robóticos em diversos segmentos da indústria alimentícia, automobilística, farmacêutica, visto que a sua implantação oferece mais eficiência, rapidez e robustez que a ação humana para um conjunto diversificado de tarefas.

As restrições relacionadas a falta de informações sobre os parâmetros da câmera, robô ou ambiente, na aplicação do Controle Servo Visual, incentivou a realização de inúmeros trabalhos ([17], [10], [5],[11]). Inclusive, alguns deles abordaram ainda a implementação de técnicas de controle que não utilizam medidas de profundidade para a locomoção dos manipuladores robóticos.

Neste trabalho, por sua vez, será efetuado um estudo sobre uma técnica de Controle Adaptativo Servo Visual proposta por ([10]) através de simulações e de testes experimentais, utilizando um braço robótico em configuração antropomórfica com 3 graus de liberdade (3DOF's) e uma Câmera Web HD estacionária. Desta maneira, além de dar prosseguimento aos estudos dos Sistemas de Controle Servo Visuais, os interessados no assunto poderão compreender com mais detalhes o sistema de controle servo visual estudado e sua aplicação prática.

1.4 Organização do trabalho

Este trabalho será dividido em quatro capítulos. O primeiro deles descrito até o momento, introduz conceitos básicos como robótica, visão computacional e servo visão. Assuntos importantes e que contextualizam o tema central deste trabalho.

O segundo capítulo apresenta uma descrição do estudo produzido abordando a definição do problema, descrição do modelo servo visual e do sistema de controle servo visual. Ressalta-se ainda que, serão apresentados temas importantes como Controle Adaptativo e sua aplicação ao problema proposto, a análise de estabilidade por Lyapunov, além da apresentação da simulação com ilustrações gráficas dos resultados.

O terceiro capítulo discorrerá a implementação prática da estratégia de controle utilizada na resolução do problema proposto. Desta maneira, serão detalhados os componentes do sistema de controle servo visual, a programação na plataforma Matlab © e a descrição da interface gráfica desenvolvida para poder efetuar o controle do manipulador, além da apresentação dos resultados experimentais.

Por fim, no quarto capítulo serão comentadas as conclusões obtidas sobre a descrição teórica da abordagem servo visual utilizada neste trabalho, além das conclusões baseadas na análise dos resultados das simulações e dos testes experimentais. Finalizando o capítulo serão comentadas as possibilidades de trabalhos futuros relacionados ao tema tratado.

Capítulo 2

Servo Visão Robótica

Em todas as áreas do conhecimento é primordial efetuar um aprofundamento teórico sobre o tema que está sendo estudado, desta maneira o capítulo que segue irá tratar questões teóricas, fornecendo ao leitor as ferramentas necessárias para o entendimento do trabalho produzido.

Serão abordadas questões como a formulação do problema, descrições do modelo servo visual, assim como da estratégia de controle utilizada. Em seguida, serão discutidos temas importantes como o Controle Adaptativo por Modelo de Referência e sua aplicação para resolução do problema, além da análise de estabilidade do sistema controlado por Lyapunov. Ao final do capítulo, os resultados das simulações serão apresentados.

2.1 Formulação do problema

Como fonte de motivação para todo estudo, há que se propor um problema e definir as suas características e restrições.

O problema proposto consiste em efetuar o controle de um manipulador robótico com três graus de liberdade (3 DOF's) com o objetivo de produzir o rastreamento de trajetórias tridimensionais criadas em imagens. Como pode ser observado na figura 2.1 a ideia principal é posicionar uma câmera não calibrada no ambiente de forma a fornecer uma realimentação visual da posição do efetuador que poderá realizar o rastreamento nas direções dos três eixos coordenados (x,y,z). Para auxiliar a identificação da posição do efetuador, um alvo vermelho de formato esférico será fixado na extremidade do manipulador.

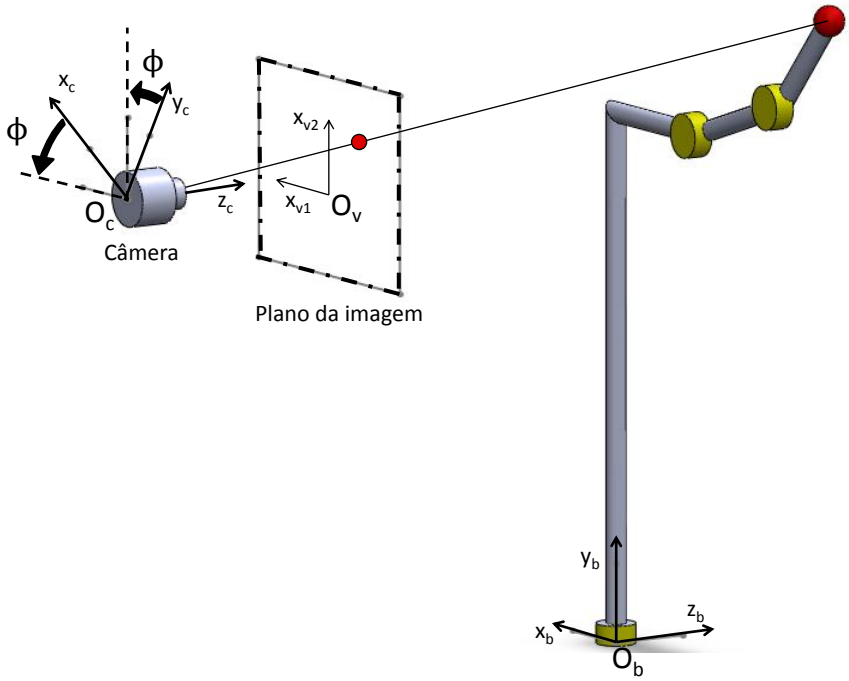

Figura 2.1: Representação simplificada de um Sistema Servo Visual

Além da restrição definida como a incerteza na calibração da câmera, o problema proposto não permite a medição de profundidade do alvo, que está diretamente relacionada à posição do efetuador, como também a medições de velocidades obtidas da imagem. Assim, é necessário criar uma alternativa adequada ao problema em questão.

Em virtude da trajetória ser definida em um ambiente tridimensional, optou-se por separar as informações obtidas da imagem em duas etapas simultâneas. Na primeira etapa, será utilizada a medida do centroide do alvo vermelho para se obter seu posicionamento 2D e efetuar o controle do rastreamento da trajetória planar na imagem. Simultaneamente, o rastreamento em profundidade será efetuado a partir da variação da área do alvo, desta maneira a característica desejada da imagem será fornecida como um valor de área em pixels^2.

Apesar da separação da trajetória em um movimento 2D no plano ortogonal ao eixo ótico da câmera e um movimento 1D paralelo ao eixo ótico, será observado que ambos os movimentos interagem significativamente. Contudo, a utilização da estratégia de controle servo visual baseada em imagem [10], irá lidar com esta interação e com o problema de controle não linear gerado pela variação de profundidade sem a possibilidade de medição direta.

2.2 Modelo Servo Visual

O trabalho de modelagem consiste em entender o comportamento dos mecanismos e processos em um sistema real. Ao receber uma entrada $u(t)$, o sistema real produz uma resposta de saída específica $y(t)$. Desta maneira, ao efetuar a modelagem do sistema, o engenheiro será capaz de representar por equações matemáticas as características I/O de um sistema.

O desenvolvimento de um modelo perfeito é praticamente impossível, devido em grande parte à complexidade dos sistemas físicos. Contudo, mesmo havendo uma descrição ideal do modelo, podem surgir dimensões infinitas ou sua descrição ser não linear ou variante no tempo. Assim, a etapa de modelagem se configura como um trabalho desafiador e complexo, visto que o engenheiro de controle têm que reproduzir um sistema com entradas e saídas que se aproximem o máximo da planta real e ao mesmo tempo simples o suficiente para permitir o desenvolvimento da estratégia de controle.

Após a definição do problema, iniciou-se a construção do modelo matemático do sistema a ser controlado. Desta maneira, serão reproduzidas as suas características, permitindo a implementação da estratégia de controle definida e a posterior simulação do sistema completo, servo visão e controle.

2.2.1 Manipulador Antropomórfico (3 DOF's)

Um manipulador antropomórfico consiste em um sistema robótico com juntas rotacionais, o que lhes permite uma mobilidade semelhante àquela da articulação humana. Os três graus de liberdade serão definidos pela utilização de três juntas rotacionais, em uma configuração de maneira que o eixo de rotação de uma das juntas é ortogonal a dois outros eixos de rotação paralelos das juntas restantes.

No trabalho proposto, as juntas estão dispostas de acordo com a figura 2.2, o que permitirá o movimento nas três direções do sistema cartesiano e consequentemente possibilitará o rastreamento da trajetória desejada.

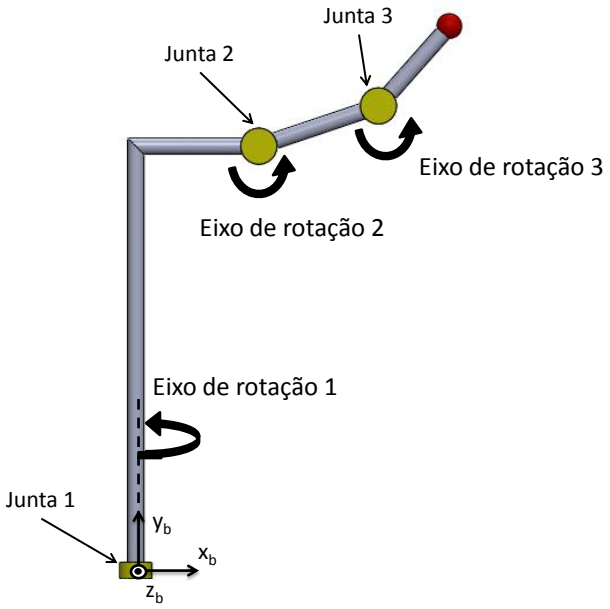

Figura 2.2: Manipulador Antropomórfico com 3 DOF's

2.2.2 Sistema de Coordenadas e Cinemática Diferencial

Existem diversas formas de representação de um ponto ou vetor no espaço, dentre as quais podemos destacar o sistema de coordenadas cartesiano e o sistema de coordenadas polares. Neste trabalho, todos os pontos relevantes serão representados no sistema cartesiano em virtude das características do problema proposto e do sistema apresentado.

Em muitos problemas é comum utilizar os conceitos para transformação de um sistema de coordenadas em outro, facilitando desta maneira os cálculos. Por este motivo, será introduzido um conceito geral de transformação de coordenadas que se estenderá em diversas etapas da modelagem do sistema servo visual.

Toda transformação de coordenadas pode ser representada por uma rotação elementar em relação a um eixo pré-determinado e uma translação [14]. Para o problema proposto os dois sistemas de coordenadas serão o sistema de referência fixado na base do manipulador, denotado O_b e o sistema de referência da câmera fixa, denotado O_c. Assim, denotando (x_c, y_c, z_c) as coordenadas do centroide do alvo representado no sistema O_c e (x_b, y_b, z_b) as coordenadas do mesmo ponto representadas no sistema O_b, a operação que relaciona as duas coordenadas segue

abaixo:

$$\begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} = R(\phi) \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} + T_{bc}^b \tag{2.1}$$

$$R(\phi) = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

Efetuando uma simples manipulação algébrica para isolar as coordenadas do alvo no sistema de referência da câmera, temos:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R(\phi)^T (\begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} - T_{bc}^b) \tag{2.3}$$

$$R^T(\phi) = \begin{bmatrix} \cos(\phi) & \sin(\phi) & 0 \\ -\sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

Deve ser ressaltado que para o problema proposto o eixo z dos dois sistemas de referência O_c e O_b são paralelos.

Desta maneira, $R(\phi) \in SO(3)$ é a matriz de rotação elementar em torno do eixo z, T_{bc}^b é o vetor de translação que representa a posição do centro O_c em relação ao sistema de referência O_b e $\phi \in (-\frac{\pi}{2}, \frac{\pi}{2})$ é o ângulo de rotação do sistema O_c em relação ao sistema O_b, efetuado em torno do eixo z. Desta maneira, ϕ representa o desalinhamento da câmera em relação à base do manipulador (figura 2.1).

A cinemática diferencial consiste na forma de se descrever o movimento de um corpo rígido, desconsiderando-se a dinâmica do sistema. Para definir a coordenadas da posição do efetuador em relação às coordenadas da base do manipulador, pode-se utilizar a convenção Denavit-Hartenberg [14] ou efetuar os cálculos diretamente. Desta maneira, denotando

(q_1, q_2, q_3) os ângulos das juntas do manipulador e (x_b, y_b, z_b) as coordenadas do alvo fixado na ponta do efetuador, representadas no sistema de referência da base. Com o auxílio das figuras (2.3 e 2.4), a transformação que relaciona os ângulos das juntas e a posição da alvo é dada por:

$$x_b = \cos(q_1)[l_1 + l_2\cos(q_2) + l_3\cos(q_2 + q_3)] \tag{2.5}$$

$$y_b = [l_0 + l_2\sin(q_2) + l_3\sin(q_2 + q_3)] \tag{2.6}$$

$$z_b = -\sin(q_1)[l_1 + l_2\cos(q_2) + l_3\cos(q_2 + q_3)] \tag{2.7}$$

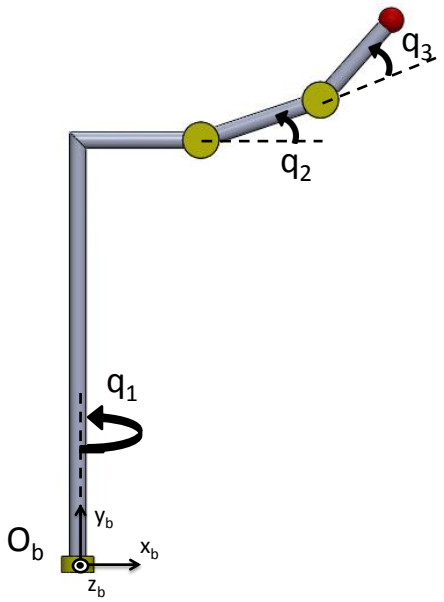

Figura 2.3: Representação dos ângulos das juntas do manipulador

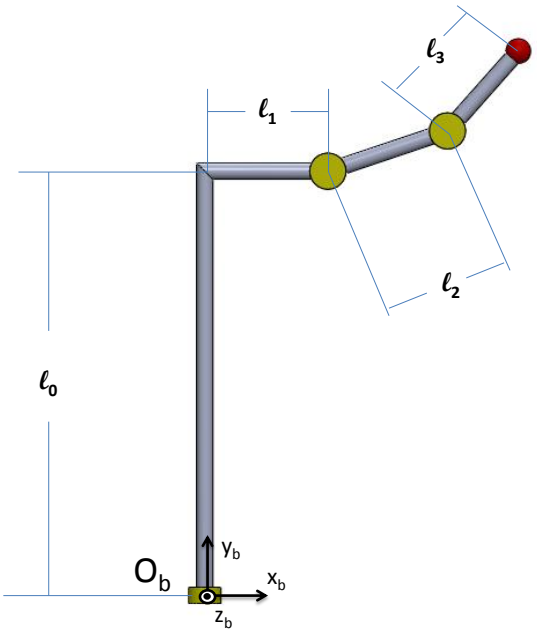

Figura 2.4: Representação das medidas do manipulador

Diferenciando as três coordenadas (x_b, y_b, z_b) em relação ao tempo, obtemos a matriz conhecida como a Matriz Jacobiana [14], que relaciona as velocidades das juntas com as velocidades translacionais do efetuador representadas por v. Desta maneira:

$$v = \begin{bmatrix} \dot{x_b} \\ \dot{y_b} \\ \dot{z_b} \end{bmatrix} = J(q) \begin{bmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix} \tag{2.8}$$

Assim como proposto por [10], o controle servo visual adotado utilizará $v = \begin{bmatrix} \dot{x_b} & \dot{y_b} & \dot{z_b} \end{bmatrix}^T$ como variável de controle a ser desenvolvida, onde $v \in \mathbb{R}^3$. Portanto, para obter as velocidades das juntas desejadas a partir do sinal de controle v, teremos:

$$\begin{bmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix} = J(q)^{-1}\, v \tag{2.9}$$

2.2.3 Modelo de Projeção de Câmeras

Com o objetivo de simplificar a modelagem da projeção da câmera, será considerada uma câmera pinhole ideal. As câmeras pinhole, rudimentarmente falando, são como uma caixa com

um pequeno orifício que permite a passagem de luz. A luz, por sua vez, ao penetrar pelo orifício produz uma projeção da imagem do ambiente na parede oposta à parede onde se encontra o furo. Desta maneira, de acordo com [2] e [12], denotando $(x_{v1}, x_{v2}) \in \mathbb{R}^2$ como as coordenadas do centroide do alvo na referência da imagem, a projeção perspectiva será dada pela seguinte transformação:

$$\begin{bmatrix} x_{v1} \\ x_{v2} \end{bmatrix} = \frac{f}{z_c} \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \end{bmatrix} \tag{2.10}$$

Na equação descrita, o termo $f > 0$ representa a distância focal da lente da câmera e os termos $\alpha_1, \alpha_2 > 0$ são fatores de escala da câmera, relacionando as medidas da imagem em pixels com as medidas reais em milímetros.

2.2.4 Modelo do Sistema Servo Visual Baseado em Imagem (IBVS)

A modelagem do sistema servo visual baseada em imagem tem como objetivo básico representar o funcionamento do manipulador robótico e a extração das características da imagem, unindo ambos em um único elemento que definirá o sistema global. Desta maneira, o modelo resultante fornecerá as saídas que serão dadas no espaço da imagem, a partir das entradas inseridas no espaço das velocidades do efetuador. Para executar tal tarefa, o movimento do robô será dividido em duas parcelas: o movimento 2D efetuado em um plano ortogonal ao eixo da câmera de visualização e um movimento em profundidade 1D efetuado em um eixo paralelo ao eixo da câmera.

Subsistema planar 2D

A partir do modelo de projeção da câmera representado pela equação (2.10) e da transformação de coordenadas da equação (2.3), a cinemática diferencial do manipulador representado no sistema de coordenadas da projeção da imagem denotado O_v será dada por:

$$\begin{bmatrix} \dot{x}_{v1} \\ \dot{x}_{v2} \end{bmatrix} = \frac{1}{z_c} \begin{bmatrix} f\,\alpha_1 & 0 & -x_{v1} \\ 0 & f\,\alpha_2 & -x_{v2} \end{bmatrix} R(\phi)^T \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{z}_b \end{bmatrix} \tag{2.11}$$

sendo $(\dot{x}_b, \dot{y}_b, \dot{z}_b)$ as velocidades de translação do alvo em relação à base do manipulador e $(\dot{x}_{v1}, \dot{x}_{v2})$ as velocidades de translação do alvo representadas no sistema de coordenadas da imagem O_v.

Subsistema de profundidade 1D

Após a descrição do movimento planar e a sua representação no sistema de coordenadas da imagem, é necessário descrever a variação de profundidade do efetuador sem efetuar sua medida direta. Assim, a descrição da profundidade será efetuada a partir da variação da área projetada do alvo no plano da imagem. Da referência [10], obtêm-se que a área projetada do alvo A_v é descrita pela seguinte equação dinâmica:

$$\dot{A}_v = \left(\frac{-2A_v}{z_c} \right) \dot{z}_b \,. \tag{2.12}$$

da equação (2.3) conclui-se que $\dot{z}_c = \dot{z}_b$. Neste momento algumas hipóteses são necessárias:

H1 - O objeto alvo apresenta uma geometria esférica. Deste modo, a área projetada no espaço da imagem se torna invariante para rotações do objeto no ambiente tridimensional.

H2 - A área projetada do objeto A_v é limitada e satisfaz a inequação $A_{min} < A_{v(t)} < A_{max}$ para qualquer tempo t pertencente ao intervalo $[0, \infty)$ onde $A_{min} > 0$.

H3 - Assume-se que o sinal de z_c é conhecido e constante, porém seu valor é desconhecido. Assim, sem perda de generalidade será adotado $z_c > 0$.

Modelo translacional completo

A união dos subsistemas planar e de profundidade permite descrever o modelo translacional completo do sistema proposto. Tendo em vista a relação $\dot{z}_c = \dot{z}_b$ facilmente obtida da

equação (2.3), a cinemática tridimencional do efetuador mapeada no espaço da imagem será dada pela seguinte relação:

$$\begin{bmatrix} \dot{x}_{v1} \\ \dot{x}_{v2} \\ \dot{A}_v \end{bmatrix} = \frac{1}{z_c} \begin{bmatrix} f\,\alpha_1 & 0 & -x_{v1} \\ 0 & f\,\alpha_2 & -x_{v2} \\ 0 & 0 & -2A_v \end{bmatrix} R(\phi) \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{z}_b \end{bmatrix} \tag{2.13}$$

Em virtude do Controle Servo Visual ser baseado no vetor de características da imagem $s(m(t), a)$ mencionado no capítulo 1 , define-se $s_T = [x_c^T \quad Av]$ como o vetor de características da imagem para o problema proposto. Denotando $v = \begin{bmatrix} \dot{x}_b & \dot{y}_b & \dot{z}_b \end{bmatrix}^T$ como a entrada do sistema, obtêm-se:

$$\dot{s}_T = \frac{1}{z_c} L_T(s_T) v, \tag{2.14}$$

em que

$$\begin{aligned} L_T(s_T) &= L_0(s_T) R(\phi)^T \\ &= \begin{bmatrix} f\alpha_1 \cos\phi & f\alpha_1 \sin\phi & -x_{v1} \\ -f\alpha_2 \sin\phi & f\alpha_2 \cos\phi & -x_{v2} \\ 0 & 0 & -2A_v \end{bmatrix} \end{aligned} \tag{2.15}$$

A matriz $L_T(s_T)$ é conhecida como o Jacobiano da Imagem ou Matriz de Interação.

2.3 Controle Adaptativo

O desenvolvimento de um sistema de controle é completado, somente após a construção da estratégia de controle. Até chegar neste ponto, o engenheiro de controle executou etapas como a definição do problema e suas características, além da modelagem do sistema a ser controlado.

O controle é desenvolvido de forma a atender alguns requerimentos de performance para a planta real. Como mencionado anteriormente, por mais rigorosa que possa ser a modelagem matemática, não é possível reproduzir perfeitamente as características do sistema real. Assim,

a definição e implementação do controle se torna uma importante e difícil etapa do processo, visto que podem existir elevadas discrepâncias entre as performances do controle desenvolvido no sistema modelado e real.

Quando algumas características do modelo são consideradas completamente desconhecidas, variantes com o tempo ou com as condições de operação variantes, a implementação do controle clássico na maior parte dos casos apresenta performance deteriorada. Em uma planta com características variantes, ao efetuar o controle com as técnicas clássicas, os pólos da malha fechada possivelmente irão ser tornar instáveis em algum momento, em virtude da variação das condições de operação do sistema, determinando assim que o controle clássico não apresente a capacidade em se adequar à esse tipo de variação. Desta maneira, muitos estudos se ocupam do desenvolvimento de Sistemas Adaptativos para lidar com tais condições adversas.

2.3.1 Conceitos Básicos

O controle Adaptativo como o nome sugere, consiste basicamente em desenvolver um sistema de realimentação sofisticado que a partir de informações de saída do sistema, "aprenda"e acomode os ganhos de controle de acordo com alterações dos parâmetros do sistema. Em [9] a exemplificação do conceito é efetuada da seguinte maneira, considere um sistema modelado pela equação dinâmica (2.16) abaixo:

$$\begin{aligned} \dot{x} &= A_i x + B_i u, \quad x(0) = x_0 \\ y &= C_i^T x + D_i u \end{aligned} \tag{2.16}$$

Considerando os parâmetros A_i, B_i, C_i, D_i variantes como funções de cada ponto de operação i, o diagrama de blocos que representa o controle adaptativo para o sistema descrito é demonstrado na figura 2.5.

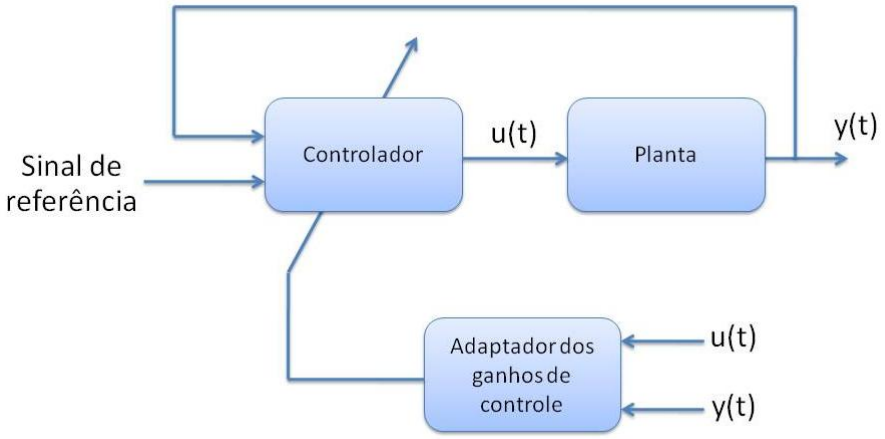

Figura 2.5: Diagrama de blocos básico do Controle Adaptativo

Alterações nos valores dos parâmetros A_i, B_i, C_i, D_i produzirão alterações na resposta de saída $y(t)$, desta maneira $y(t)$ carrega consigo informações do estado $x(t)$ do sistema que serão utilizadas para prover a adaptação dos ganhos de controle durante a execução da malha de realimentação.

O modo como os valores dos ganhos são alterados em resposta às alterações da planta, diferenciam um esquema adaptativo do outro. Dentre as diversas estratégias de controle adaptativo, destacam - se:

- Controle Robusto
- *Gain Scheduling*
- Controle Adaptativo Direto e Indireto
- Controle Adaptativo por Modelo de Referência (MRAC)

Não é objetivo deste trabalho descrever todos estes esquemas de controle adaptativo. Assim, objetivando o melhor entendimento do leitor da estratégia de controle implementada, a próxima subseção fornecerá os conceitos básicos do Controle Adaptativo por Modelo de Referência (MRAC) e posteriormente descreverá a sua implementação para a resolução do problema proposto.

2.3.2 Controle Adaptativo por Modelo de Referência (MRAC)

No MRAC, primeiramente é definido um modelo totalmente conhecido que descreve, em termos de entradas e saídas (I/O), o comportamento desejado que a planta em malha fechada deve apresentar. Este modelo é denominado modelo de referência. Deste modo, o objetivo básico do MRAC consiste em desenvolver uma lei de controle em malha fechada que modifique aspectos estruturais e dinâmicos da planta de forma que as suas propriedades (I/O) sejam iguais as do modelo de referência construído.

O esquema geral do MRAC esta representado na figura 2.6. O modelo de referência foi desenvolvido para produzir a saída desejada y_m a qual a planta deve seguir, sendo o erro e_1 a discrepância entre a saída desejada e a saída atual. O sistema controlado, por sua vez, apresentará um mecanismo de ajuste para a estimação do parâmetro de controle $\theta(t)$ e um controlador $C(\theta)$ que fará parte do controle em malha fechada aplicado à planta. Deste modo, o MRAC irá desenvolver o mecanismo de estimação do parâmetro de controle e o controlador de forma que a saída y_p do sistema controlado seja limitada e o mais próxima possível da saída desejada y_m.

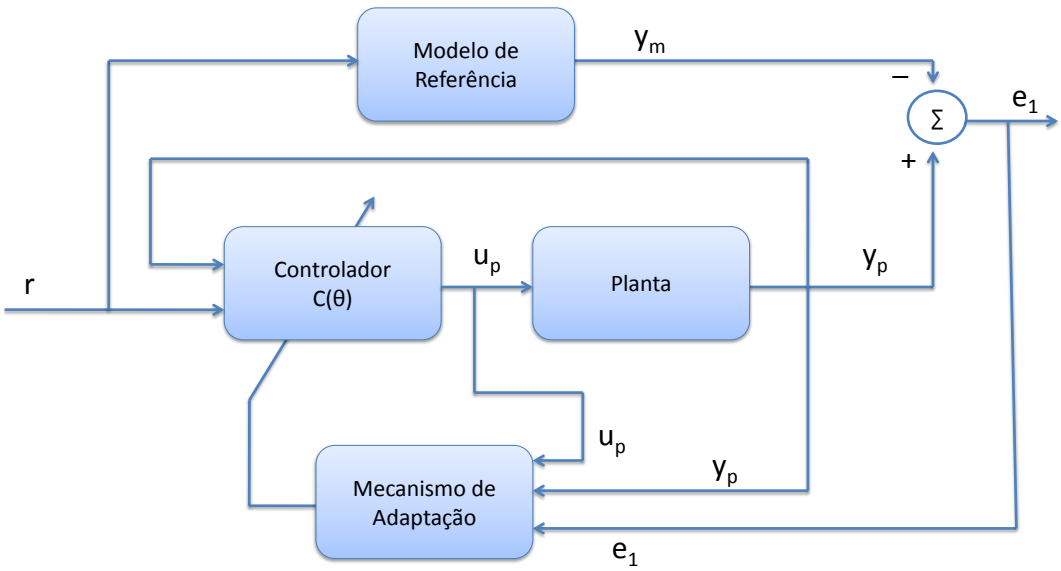

Figura 2.6: Diagrama de blocos básico do MRAC

Os esquemas MRAC podem ser caracterizados como diretos ou indiretos e ainda com leis de adaptação normalizadas e não normalizadas. Para fins práticos, as próximas subseções se aterão ao método direto, pois seu conceito apresenta grande importância para o entendimento da estratégia de controle implementada neste estudo. De maneira que, caso o leitor queira se aprofundar nas outras metodologias MRAC um estudo detalhado pode ser encontrado em [9].

MRAC Direto Simples

O esquema MRAC Direto consiste basicamente em atualizar o vetor de parâmetros θ do controlador $C(\theta)$ diretamente a partir da lei de adaptação. Desta maneira, o parâmetro θ será estimado e a partir da determinação da função $C(\theta)$ o sinal de controle $u(t)$ será, então, obtido.

As leis de adaptação do MRAC Direto Simples podem ser classificadas como normalizadas e a não normalizadas. A lei normalizada é baseada em uma normalização do erro de estimação e em uma apropriada parametrização da planta que envolve os parâmetros de controle desconhecidos. Já as leis não normalizadas não utilizam nenhuma normalização para o erro paramétrico, apresentando assim características distintas. Desta maneira, seguindo o estudo proposto por [10], será utilizada uma lei de adaptação não normalizada.

Para a construção da lei de adaptação, em [9] e [15] é adotada uma abordagem que utiliza o conceito de funções de transferência estritamente reais positivas (SPR) e a teoria de Lyapunov para garantir a convergência do erro de rastreamento para zero.

A seguir, será descrito todo o procedimento para aplicação do MRAC Direto Simples em uma planta não linear escalar. Assim, considere o sistema abaixo:

$$\dot{x} = af(x) + bg(x)u, \quad x \in \mathbb{R}$$

onde a , b são escalares desconhecidos, $f(x)$, $g(x)$ são funções conhecidas com $g(x) > c > 0 \ \forall x \in \mathbb{R}$ e para alguma constante $c > 0$. O sinal de b é conhecido porém seu valor é desconhecido e $f(x)$ é limitado para x limitado. Assumindo $x > 0$, convenientemente escolheremos $f(x) = 0$, $g(x) = x$, assim teremos:

$$\dot{x} = xbu, \quad x \in \mathbb{R} \tag{2.17}$$

Agora, deseja-se que x rastreie o estado x_m do modelo de referência abaixo

$$\dot{x}_m = -\lambda_m x_m + \lambda_m r \tag{2.18}$$

para algum sinal de referência limitado r. Segundo [9], a lei de controle ideal que permite o rastreamento perfeito é

$$u^* = \frac{1}{x}[k^* x + l^* r] \tag{2.19}$$

$$k^* = \frac{-\lambda_m}{b} \ , \ l^* = \frac{\lambda_m}{b}$$

denotando $\theta^* = \frac{\lambda_m}{b}$ e substituindo em u , a equação ficará na forma:

$$u^* = \theta^* \big(\frac{r-x}{x}\big) \tag{2.20}$$

em virtude do desconhecimento do parâmetro b e baseado no princípio da equivalência certa [9], propõe-se uma lei de controle com ganhos adaptativos da seguinte forma

$$u = \theta(t)\sigma \tag{2.21}$$

com $\sigma = \frac{r-x}{x}$, o parâmetro $\theta(t)$ será a estimativa do ganho do controlador desconhecido θ^*, gerado pela lei de adaptação.

Para descrever a lei de adaptação , primeiramente utiliza-se o artifício de somar e subtrair o termo $b\theta^*(r-x)$ da equação da planta (2.17), substituindo $b\theta^* = \lambda_m$ e $u = \theta(\frac{r-x}{x})$, obtêm-se

$$\begin{aligned} \dot{x} &= xbu + b\theta^*(r-x) - b\theta^*(r-x) \\ &= \lambda_m(r-x) + b(xu - \theta^*(r-x)) \\ &= \lambda_m(r-x) + b[(\theta - \theta^*)(r-x)] \end{aligned} \tag{2.22}$$

transformando a equação (2.22) e a equação do modelo de referência (2.18) para o domínio de Laplace, teremos

$$X = \frac{\lambda_m R}{s + \lambda_m} + \frac{b}{s + \lambda_m}[\,\mathcal{L}\{(\theta - \theta^*)(r - x)\}\,] \tag{2.23}$$

$$X_m = \lambda_m \frac{R}{s + \lambda_m} \tag{2.24}$$

em que X, X_m, R são as respectivas variáveis x, x_m, r no domínio de Laplace, e $\mathcal{L}$ representa a transformação de uma função para o domínio de Laplace. Substituindo (2.24) em (2.23) obtêm-se que:

$$X = X_m + \frac{b}{s + \lambda_m}[\,\mathcal{L}\{(\theta - \theta^*)(r - x)\}\,]\ , \tag{2.25}$$

transformando novamente para o domínio do tempo

$$\begin{aligned} \dot{x} + \lambda_m x &= \dot{x}_m + \lambda_m x_m + b[(\theta - \theta^*)(r - x)] \\ \dot{x} - \dot{x}_m &= -\lambda_m (x - x_m) + b[(\theta - \theta^*)(r - x)] \end{aligned} \tag{2.26}$$

definindo $\dot{e} = \dot{x} - \dot{x}_m$, $e = x - x_m$ e $\tilde{\theta} = \theta - \theta^*$ a dinâmica do erro do sistema controlado será:

$$\dot{e} = -\lambda_m e + b[\tilde{\theta}(r - x)] \tag{2.27}$$

Em [15] utiliza-se a abordagem SPR-Lyapunov para a definição da lei de controle. Considerando o sistema (2.27), temos que a função de transferência que o rege será dada por:

$$H(s) \ = \ \frac{1}{s \ + \ \lambda_m} \quad , \ \lambda_m \ > \ 0 \tag{2.28}$$

A partir de [15], obtêm-se que a função de transferência (2.28) é SPR. Podemos agora desenvolver a lei de adaptação sem normalização segundo [9]. Assim , define-se:

$$\dot{\tilde{\theta}} = \dot{\theta} = -\gamma\, e\,(r - x)\, sgn(b) \tag{2.29}$$

considerando a seguinte função de Lyapunov:

$$V(e, \tilde{\theta}) = \frac{e^2}{2} + \frac{\tilde{\theta}^2}{2\gamma}|b| \tag{2.30}$$

diferenciando a equação anterior e substituindo $\dot{\theta}$ e $\dot{e}$ obtêm-se

$$\begin{aligned}
\dot{V}(e, \tilde{\theta}) &= e\,\dot{e} + \frac{\tilde{\theta}\,\dot{\theta}\,|b|}{\gamma} \\
&= -\lambda_m\, e^2 + e\, b\,[\tilde{\theta}(r - x)] - \frac{\tilde{\theta}\,\gamma\, e\,(r - x)\, sgn(b)\,|b|}{\gamma} \\
&= -\lambda_m\, e^2 + e\, b\,\tilde{\theta}\,(r - x) - e\, b\,\tilde{\theta}\,(r - x) \\
\dot{V}(e, \tilde{\theta}) &= -\lambda_m e^2
\end{aligned} \tag{2.31}$$

assim, por Lyapunov as equações (2.30) e (2.31) demonstram que $e(t)$ e θ são globalmente limitados. Obtendo a derivada segunda da função de Lyapunov, teremos:

$$\ddot{V} = -2\lambda_m\, e\,\dot{e} \tag{2.32}$$

Desta maneira, com $e(t)$, $(r - x)$, e $\tilde{\theta}$ limitados, pela equação (2.27) temos que $\dot{e}$ será limitado e a partir de (2.32) garante-se que $\ddot{V}$ também será limitado, por este motivo a equação (2.31) terá continuidade uniforme. A partir do lema de Barbalat [15] conclui-se, então, que o erro $e(t)$ converge assintoticamente para zero.

2.3.3 MRAC Servo Visual

Nas seções anteriores, convenientemente descreveu-se as abordagens MRAC Direta para plantas não lineares. Esta ação teve por objetivo embasar o leitor com a teoria necessária para

o entendimento da estratégia de controle implementada. Entretanto, ao aplicar o MRAC para o sistema servo visual, algumas modificações e alterações foram efetuadas em virtude das restrições do problema.

Como já mencionado na seção 2.2, o modelo servo visual foi dividido em um movimento planar 2D cuja informação de posição do alvo foi extraída a partir do centroide da imagem e um movimento 1D no eixo ortogonal ao plano da imagem cuja informação foi extraída indiretamente a partir da variação da área do alvo na imagem. Pôde-se observar que, embora tenham sido modelados separadamente, os dois subsistemas são parcialmente acoplados, fato este que gerou algumas implicações na resolução do problema.

Ressalta-se ainda que, a impossibilidade de medida das velocidades na imagem resultaram em algumas modificações na estratégia de controle, fazendo com que o MRAC Servo Visual apresente diferenças em relação à abordagem tradicional.

Deste modo, para a resolução do rastreamento planar 2D utilizou-se o MRAC com fatoração SDU [10], por uma série de motivos que serão comentados ao longo de sua descrição. Já a resolução do rastreamento em profundidade se deu com base no MRAC Direto ([10] , [9]).

Rastreamento Adaptativo em Profundidade

Retomando a equação (2.12) que descreve a relação entre a variação da área projetada na imagem e a profundidade do alvo, têm-se:

$$\dot{A}_v = \left(\frac{-2A_v}{z_c}\right)\dot{z}_b \qquad \text{subseção 2.2.4 - equação 2.12}$$

Substituindo $\dot{z}_b$ por $\dot{z}_c$ e definindo A_{v0} como a área projetada em um instante inicial $t_0 < t$, com a respectiva medida de profundidade neste mesmo instante denotada z_{c0} . Ao integrar ambos os lados da equação (2.12) e eliminar o tempo, obtêm-se:

$$z_c(A_v)^{\frac{1}{2}} = z_{c0}(A_{v0})^{\frac{1}{2}} = \beta \tag{2.33}$$

reescrevendo a equação (2.12) com o novo parâmetro β, teremos

$$\dot{A}_v = -2\beta^{-1}(A_v)^{\frac{3}{2}}\, v_z \tag{2.34}$$

em que $v_z = \dot{z}_b$. Em posse da equação (2.33) e utilizando as hipóteses H2 e H3 mencionadas anteriormente, obtêm-se que $\beta^{-1} > 0$.

Lei de Controle

Em posse da equação (2.34), o objetivo agora é desenvolver uma lei de controle para $v_z(t)$ em malha fechada, que permita o rastreamento de A_{vm}, definido pelo modelo de referência abaixo:

$$\dot{A}_{vm} = -\lambda_m A_{vm} + \lambda_m A_v^*(t) \tag{2.35}$$

em que $A_v^*(t) \in \mathbb{R}$ é um sinal qualquer limitado e $\lambda_m > 0$. Desde que A_v seja positivo e limitado (suposição H2), pode-se definir erro de rastreamento em profundidade como:

$$e_a = (A_v)^{-\frac{1}{2}} - (A_{vm})^{-\frac{1}{2}} \tag{2.36}$$

com $A_{vm} > 0$. Derivando a equação do erro em relação ao tempo e substituindo as equações (2.34) e (2.35), obtêm-se a equação dinâmica do erro como:

$$\begin{aligned}
\dot{e}_a &= -\frac{1}{2}(A_v)^{-\frac{3}{2}}\, \dot{A}_v + \frac{1}{2}(A_{vm})^{-\frac{3}{2}}\, \dot{A}_{vm} \\
&= \frac{v_z}{\beta} + \frac{1}{2}(A_{vm})^{-\frac{3}{2}}\, \lambda_m (A_v^* - A_{vm}) \\
\dot{e}_a &= \frac{v_z}{\beta} + \frac{\lambda_m}{2}\,(A_{vm})^{-\frac{3}{2}}\,(A_v^* - A_{vm})
\end{aligned} \tag{2.37}$$

Neste momento, ocorre a modificação mencionada anteriormente em relação ao MRAC direto. Ao definir a lei de controle ideal v_z^*, surgirá o termo e_{af} que representa o erro e_a filtrado, fato ocasionado pela restrição imposta pela não utilização de medidas de velocidades da imagem. Assim, a partir de [10] teremos:

$$v_z^* = \underbrace{\beta}_{\theta^*} \underbrace{[-\lambda\, e_{af} - \frac{\lambda_m}{2}\,(A_{vm})^{-\frac{3}{2}}\,(A_v^* - A_{vm})]}_{w} \tag{2.38}$$

Utilizando o princípio da equivalência certa [15], definimos assim como anteriormente (subseção 2.3.2) $v_z^* = \theta^* w$, onde $\theta^* \in \mathbb{R}$ é o parâmetro incerto de adaptação e $w \in \mathbb{R}$ é o regressor escalar relacionado ao rastreamento em profundidade. Desta maneira, o sinal de controle estimado será $v_z = \theta\, w$.

O parâmetro e_{af} é obtido da seguinte forma

$$\dot{e}_{af} = -\lambda\, e_{af} + \lambda\, e_a \quad , \lambda > 0 \tag{2.39}$$

Utilizando o artifício de somar e subtrair v_z^* da equação (2.37) . A dinâmica do erro será dada por:

$$\begin{aligned}
\dot{e}_a &= (v_z + v_z^* - v_z^*)\beta^{-1} \;+\; \frac{\lambda_m}{2}\,(A_{vm})^{-\frac{3}{2}}\,(A_v^* - A_{vm}) \\
&= (v_z - v_z^*)\beta^{-1} \;+\; v_z^*\,\beta^{-1} \;+\; \frac{\lambda_m}{2}\,(A_{vm})^{-\frac{3}{2}}\,(A_v^* - A_{vm}) \\
&= \tilde{v}_z\,\beta^{-1} - \lambda\, e_{af} - \frac{\lambda_m}{2}\,(A_{vm})^{-\frac{3}{2}}\,(A_v^* - A_{vm}) + \frac{\lambda_m}{2}\,(A_{vm})^{-\frac{3}{2}}\,(A_v^* - A_{vm}) \\
\dot{e}_a &= -\lambda\, e_{af} + \beta^{-1}\,\tilde{v}_z
\end{aligned} \tag{2.40}$$

onde $\tilde{v}_z = v_z - v_z^*$. Assim, objetivando a análise da convergência do erro para zero através da abordagem SPR-Lyapunov, define-se o erro de rastreamento aumentado como $\xi_a \triangleq [e_{af} \quad e_a]^T$ e levando em consideração as equações (2.39) e (2.40) a dinâmica do erro aumentado será:

$$\dot{\xi}_a = A_a \xi_a + b_a\,\tilde{v}_z\,, \quad e_a = c_a \xi_a \tag{2.41}$$

$$A_a = \begin{bmatrix} -\lambda & \lambda \\ -\lambda & 0 \end{bmatrix}, \quad b_a = \begin{bmatrix} 0 \\ \beta^{-1} \end{bmatrix}, \quad c_a = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

Transformando o sistema de equação (2.41) para o domínio de Laplace, teremos:

$$(sI - A_a)\xi_a(s) = b_a\,\tilde{V}_z$$

$$\xi_a(s) = (sI - A_a)^{-1} b_a\,\tilde{V}_z$$

$$e_a(s) = \underbrace{c_a(sI - A_a)^{-1} b_a}_{H(s)}\,\tilde{V}_z$$

$$H(s) = c_a(sI - A_a)^{-1} b_a \tag{2.42}$$

substituindo os valores da equação (2.41) e efetuando os devidos cálculos, obtêm-se que:

$$H(s) = \frac{\beta^{-1}(s\,+\,\lambda)}{s^2\,+\,\lambda s\,+\,\lambda^2} \quad , \quad \lambda\,>\,0 \tag{2.43}$$

Neste momento, é necessário comprovar que a função de transferência $H(s)$ que rege o sistema do erro aumentado é SPR. Após alguns cálculos e com o auxílio da literatura [15] , foi possível comprovar que o sistema é de fato SPR. A prova completa pode ser vista no apêndice B.

Deste modo, novamente a partir de [15], obtêm-se que se a função de transferência $H(s)$ é SPR, então existem as matrizes $P_a = P_a^T > 0$ e $Q_a = Q_a^T > 0$ que satisfazem a relação:

$$A_a^T P_a + P_a A_a = -2Q_a\,, \quad P_a b_a = c_a^T \tag{2.44}$$

Este resultado será importante nas próximas etapas.

Lei de Adaptação

Após definição da lei de controle, é necessário desenvolver a lei de adaptação que faça o parâmetro desconhecido θ rastrear o valor ideal θ^*. Assim, denotando $\tilde{\theta} = \theta - \theta^*$ e de acordo

com a abordagem descrita em [15], a lei de adaptação que além de garantir a convergência assintótica de $\xi_a(t)$, garante também que o sinais do sistema sejam uniformemente limitados, será da seguinte forma:

$$\dot{\theta} = -\gamma_a \, sgn(\beta^{-1}) w \, e_a$$

Como já mencionado, temos que $\beta^{-1} > 0$, logo

$$\dot{\theta} = -\gamma_a \, w \, e_a \tag{2.45}$$

Análise de Estabilidade

Neste momento, pretende-se comprovar a natureza convergente do sistema (2.41) a partir da seguinte função de Lyapunov:

$$2V(\xi_a \, , \, \tilde{\theta}) = \xi_a^T P_a \xi_a \quad + \quad \frac{\tilde{\theta}^2}{\gamma_a} \tag{2.46}$$

derivando a equação (2.46) e substituindo por (2.41),(2.45) e (2.44) teremos:

$$\begin{aligned}
2\dot{V}(\xi_a \, , \, \tilde{\theta}) &= \dot{\xi}_a^T P_a \xi_a + \xi_a^T P_a \dot{\xi}_a \quad + \quad \frac{2\tilde{\theta}\dot{\theta}}{\gamma_a} \\
&= (\xi_a^T A_a^T + b_a^T \, \tilde{v_z}) P_a \xi_a \quad + \quad \xi_a^T P_a (A_a \xi_a + b_a \tilde{v}_z) \quad + \quad \frac{2\tilde{\theta}(-\gamma_a \, w \, e_a)}{\gamma_a} \\
&= \xi_a^T A_a^T P_a \xi_a \; + \; \xi_a^T P_a A_a \xi_a \; + \; \underbrace{b_a^T \, \tilde{v_z} P_a \xi_a}_{escalar} \; + \; \underbrace{\xi_a^T P_a b_a \tilde{v}_z}_{escalar} \; -2\tilde{\theta} \, w \, e_a \\
&= \xi_a^T (A_a^T P_a \; + \; P_a A_a) \xi_a \; + \; 2 \underbrace{\xi_a^T P_a b_a}_{e_a} \tilde{v}_z \; - 2 \underbrace{\tilde{\theta} \, w}_{\tilde{v_z}} \, e_a
\end{aligned}$$

como pode ser observado, os dois últimos termos se cancelarão, pois são iguais. Substituindo $A_a^T P_a + P_a A_a = -2Q_a$, teremos:

$$2\dot{V}(\xi_a\,,\,\tilde{\theta}) = -2\xi_a^T(Q_a)\xi_a \;\leq 0 \tag{2.47}$$

Assim, Q_a sendo simétrica e positiva definida, as equações (2.46) e (2.47) garantem que e_a , e_{af} e θ sejam globalmente limitados. Da equação (2.38) obtêm - se que v_z é limitado e consequentemente a partir das equações (2.47) e (2.41) o sinal $\dot{\xi}_a$ também será limitado. Deste modo, obtendo-se a derivada segunda da função de Lyapunov, temos:

$$\ddot{V}(\xi_a\,,\,\tilde{\theta}) = -2\xi_a^T(Q_a)\dot{\xi}_a \tag{2.48}$$

Da equação anterior, observa-se que $\ddot{V}(\xi_a\,,\,\tilde{\theta})$ será limitado pois $\dot{\xi}_a$ e ξ_a são limitados, consequentemente a função $\dot{V}$ terá continuidade uniforme e pelo lema de Barbalat o erro e_a irá convergir assintoticamente para zero.

Rastreamento Adaptativo Planar

Garantida a convergência do rastreamento em profundidade, pode-se agora definir o controle que permitirá o rastreamento planar 2D. Retomando o modelo do subsistema planar (2.11), ao substituir (2.33) e $R(\phi)^T$, obtêm-se a seguinte equação:

$$\begin{bmatrix} \dot{x}_{v1} \\ \dot{x}_{v2} \end{bmatrix} = \frac{1}{z_c} \begin{bmatrix} f\,\alpha_1\cos\phi & f\,\alpha_1\sin\phi & -x_{v1} \\ -f\,\alpha_2\sin\phi & f\,\alpha_2\cos\phi & -x_{v2} \end{bmatrix} \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{z}_b \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_{v1} \\ \dot{x}_{v2} \end{bmatrix} = \beta^{-1}(A_v)^{\frac{1}{2}} \begin{bmatrix} f\,\alpha_1\cos\phi & f\,\alpha_1\sin\phi & -x_{v1} \\ -f\,\alpha_2\sin\phi & f\,\alpha_2\cos\phi & -x_{v2} \end{bmatrix} \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{z}_b \end{bmatrix} \tag{2.49}$$

rearranjando os termos do sistema, uma nova parametrização será dada por

$$\begin{bmatrix} \dot{x}_{v1} \\ \dot{x}_{v2} \end{bmatrix} = \beta^{-1}(A_v)^{\frac{1}{2}} f \begin{bmatrix} \alpha_1\cos\phi & \alpha_1\sin\phi \\ -\alpha_2\sin\phi & \alpha_2\cos\phi \end{bmatrix} \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \end{bmatrix} - \beta^{-1}(A_v)^{\frac{1}{2}} \begin{bmatrix} x_{v1} \\ x_{v2} \end{bmatrix} \dot{z}_b \tag{2.50}$$

renomeando as varáveis teremos:

$$\dot{x}_v = \beta^{-1}(A_v)^{\frac{1}{2}} K_p\, u + \delta(x_v, A_v, v_z) \tag{2.51}$$

com

$$K_p \triangleq f \begin{bmatrix} \alpha_1 \cos\phi & \alpha_1 \sin\phi \\ -\alpha_2 \sin\phi & \alpha_2 \cos\phi \end{bmatrix} \quad , \quad u \triangleq \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \end{bmatrix} \tag{2.52}$$

$$v_z \triangleq \dot{z}_b \quad , \quad \delta(x_v, A_v, v_z) \triangleq -\beta^{-1}(A_v)^{\frac{1}{2}}\, x_v\, v_z$$

Pode-se observar que o acoplamento entre os dois subsistemas (2D e 1D) ocorre pela presença dos parâmetros A_v e v_z na equação (2.51).

Lei de Controle

Após a parametrização do modelo do subsistema planar 2D, será desenvolvida a lei de controle $u(t)$ que faça x_v rastrear x_{vm}. Assim, pela abordagem MRAC padrão, define-se x_{vm} a partir do modelo de referência abaixo:

$$\dot{x}_{vm} = -\Lambda_M x_{vm} + \Lambda_M x_v^*(t) \tag{2.53}$$

com qualquer sinal limitado e contínuo $x_v^*(t) \in \mathbb{R}^2$ e $\Lambda_M = \lambda_M\, I$ onde $I \in \mathbb{R}^{2\text{x}2}$ é a matriz identidade e $\lambda_M \in \mathbb{R}$, $\lambda_M > 0$. A partir das equações (2.35) e (2.53) será definido o vetor aumentado das características desejadas da imagem como $\mathcal{X}_{vm} = [\, x_{vm}^T \quad A_{vm}]$. Após a adoção da trajetória desejada como a especificada em (2.53), define-se o erro de rastreamento planar e_v como:

$$e_v = x_v(A_v)^{-\frac{1}{2}} - x_{vm}(A_{vm})^{-\frac{1}{2}} \tag{2.54}$$

com $A_v > 0$ e limitado (proposição H2) e $A_{vm} > 0$ e limitado. Em posse das equações (2.51), (2.53), (2.34), (2.35) e diferenciando o erro de rastreamento, a sua dinâmica será dada por:

$$\dot{e}_v = \dot{x}_v\,(A_v)^{-\frac{1}{2}} - \frac{1}{2}(A_v)^{-\frac{3}{2}}\dot{A}_v\,x_v - \dot{x}_{vm}\,(A_{vm})^{-\frac{1}{2}} + \frac{1}{2}(A_{vm})^{-\frac{3}{2}}\dot{A}_{vm}x_{vm}$$

$$= \beta^{-1}K_p\,u - \beta^{-1}\,x_v\,v_z + \beta^{-1}\,x_v\,v_z - \Lambda_M(x_v^* - x_{vm})\,(A_{vm})^{-\frac{1}{2}} + \frac{1}{2}(A_{vm})^{-\frac{3}{2}}\lambda_m(A_v^* - A_{vm})x_{vm}$$

$$= \beta^{-1}K_p\,u \underbrace{- \Lambda_M(x_v^* - x_{vm})\,(A_{vm})^{-\frac{1}{2}} + \frac{1}{2}(A_{vm})^{-\frac{3}{2}}\lambda_m(A_v^* - A_{vm})x_{vm}}_{\eta(\mathcal{X}_{vm})}$$

definindo $\zeta_v \triangleq x_v^* - x_{vm}$ e $\zeta_a \triangleq (A_v^* - A_{vm})$. Assim,

$$\eta(\mathcal{X}_{vm}) = -\Lambda_M\,\zeta_v\,(A_{vm})^{-\frac{1}{2}} + \frac{1}{2}(A_{vm})^{-\frac{3}{2}}\lambda_m\,\zeta_a\,x_{vm}$$

e por fim a equação final da dinâmica do erro será:

$$\dot{e_v} = \beta^{-1}K_p\,u - \Lambda_M\,\zeta_v\,(A_{vm})^{-\frac{1}{2}} + \frac{1}{2}(A_{vm})^{-\frac{3}{2}}\lambda_m\,\zeta_a\,x_{vm}$$

$$\dot{e_v} = \beta^{-1}K_p\,u + \eta(\mathcal{X}_{vm}) \tag{2.55}$$

Tendo em vista a restrição definida como a impossibilidade de medição de velocidades obtidas da imagem, então, de acordo com [10], utiliza-se o artifício de construção do erro filtrado e_{vf}. O e_{vf} é obtido da seguinte equação:

$$\dot{e}_{vf} = -\Lambda\,e_{vf} + \Lambda\,e_v \quad , \quad \Lambda = \lambda\,I \quad , \quad \lambda > 0 \tag{2.56}$$

em que $I \in \mathbb{R}^{2\text{x}2}$ é a matriz identidade.

Assim como no rastreamento em profundidade, há que se definir o controle ideal que fornece o rastreamento planar perfeito entre x_v e x_{vm}, deste modo:

$$u^* = \underbrace{\beta\, K_p^{-1}}_{\Phi^*}\, \underbrace{[-\Lambda\, e_{pf} \quad - \eta(\mathcal{X}_{vm})]}_{w_p} \tag{2.57}$$

Utilizando novamente o princípio da equivalência certa [15], podemos parametrizar o controle ideal u^* como $u^* = \Phi^*\, w_p$, desta maneira

$$\Phi^* = \beta\, K_p^{-1}$$

$$w_p = [-\Lambda\, e_{pf} \quad - \eta(\mathcal{X}_{vm})]$$

em que Φ^* é a matriz de parâmetros incertos e w_p é o vetor regressor ligado diretamente ao rastreamento planar.

Usando o artifício da soma e subtração do termo u^* na equação dinâmica do erro de rastreamento planar (2.55) e substituindo pela equação (2.57), a nova dinâmica será:

$$\begin{aligned}
\dot{e_v} &= \beta^{-1} K_p\,(u + u^* - u^*) + \eta \\
&= \beta^{-1} K_p\,(u - u^*) + \beta^{-1} K_p\, u^* + \eta \\
&= \beta^{-1} K_p\,(u - u^*) + \beta^{-1} K_p\, \beta\, K_p^{-1}\,[-\Lambda\, e_{pf} \quad - \eta] + \eta \\
&= \beta^{-1} K_p\, \tilde{u} - \Lambda\, e_{pf} \quad - \eta + \eta \\
\dot{e_v} &= -\Lambda\, e_{pf} + \beta^{-1} K_p\, \tilde{u}
\end{aligned} \tag{2.58}$$

onde $\tilde{u} = u - u^*$.

Agora, seria natural definir o sinal de controle para o subsistema planar como $u = \Phi\, w_p$ onde Φ seria a matriz estimada, analogamente ao que foi efetuado para o rastreamento em profundidade, contudo em virtude das características do subsistema 2D e das restrições do problema proposto, uma nova parametrização que apresenta relação direta com a fatoração SDU será efetuada.

Parametrização via Fatoração SDU

Primeiramente, será definido a partir de e_v e e_{vf} o vetor do erro aumentado devido ao rastreamento planar $\xi_v = [e_{vf}^T \quad e_v^T]^T$. Desta maneira, tendo as equações (2.58) e (2.56), o sistema de erro aumentado completo será:

$$\dot{\xi}_v = A_v\,\xi_v + B_v\,K_p\,\tilde{u}\,, \quad e_v = C_v\,\xi_v \tag{2.59}$$

$$A_v = \begin{bmatrix} -\Lambda & \Lambda \\ -\Lambda & 0 \end{bmatrix}, \quad B_v = \begin{bmatrix} 0 \\ \beta^{-1} \end{bmatrix}, \quad C_v = \begin{bmatrix} 0 & I \end{bmatrix}$$

Deve-se ressaltar que, para facilitar os cálculos as matrizes A_v , B_v e C_v foram representadas em forma de blocos de matrizes. Desta maneira, cada elemento é na verdade uma matriz que pertence ao domínio $\mathbb{R}^{2\text{x}2}$. A seguir, temos o mesmo sistema (2.59) com os índices indicando o domínio de cada elemento.

$$\dot{\xi}_v = A_v\,\xi_v + B_v\,K_p\,\tilde{u}\,, \quad e_v = C_v\,\xi_v$$

$$A_v = \begin{bmatrix} -\Lambda_{2\text{x}2} & \Lambda_{2\text{x}2} \\ -\Lambda_{2\text{x}2} & 0_{2\text{x}2} \end{bmatrix}, \quad B_v = \begin{bmatrix} 0_{2\text{x}2} \\ I_{2\text{x}2}\,\beta^{-1} \end{bmatrix}, \quad C_v = \begin{bmatrix} 0_{2\text{x}2} & I_{2\text{x}2} \end{bmatrix}$$

Novamente será efetuada a análise do sistema de erro planar (2.59) , pela abordagem SPR - Lyapunov, para posteriormente desenvolver a lei de adaptação. O sistema também será transformado para o domínio de Laplace, contudo o leitor deve ter consciência que ao invés de se obter uma função de transferência, será obtida agora, uma matriz de transferência. Isto se deve ao fato do sistema ser MIMO. Deste modo, a transformação será:

$$(sI - A_v)\xi_v(s) = B_v\, K_p\, \tilde{V}_z$$

$$\xi_v(s) = (sI - A_v)^{-1}\, B_v\, K_p\, \tilde{V}_z$$

$$e_v(s) = \underbrace{C_v(sI - A_v)^{-1}B_v}_{H(s)}\, K_p\, \tilde{V}_z \tag{2.60}$$

$$H(s) = C_v(sI - A_v)^{-1}B_v \tag{2.61}$$

Observando as equações (2.45) , (2.41) e (2.43) do caso escalar para rastreamento em profundidade, ao se desenvolver a análise SPR-Lypunov([15],[9]) para a obtenção da lei de adaptação do caso planar 2D, o leitor chegaria a conclusão de que é necessário obter o valor do sinal da matriz K_p, tendo em vista que o sinal do parâmetro β^{-1} é positivo. De acordo com [9], a generalização do conceito MRAC para o caso MIMO exige a existência de uma matriz S_p, tal que $K_p\, S_p = K_p\, S_p^T$. Esta condição se torna difícil de se alcançar, em virtude do desconhecimento dos elementos da matriz K_p, visto que a calibração da câmera é desconhecida.

Portanto, a generalização da condição de sinal do caso escalar para o caso matricial se torna um fator limitante para a aplicação do MRAC tradicional.

Com o objetivo de solucionar esta questão, em [6] foi desenvolvido um estudo, aplicando a parametrização por fatoração SDU. O estudo mencionado, prova que se K_p apresenta seus menores principais maiores que zero, então existe uma fatoração do tipo $K_p = S\, D\, U$ em que a matriz S é simétrica e positiva definida, a matriz D é diagonal com entradas unitárias positivas ou negativas e U é uma matriz triangular superior com os elementos diagonais unitários.
Do estudo efetuado em [6], obtemos que para executar a fatoração SDU os seguintes procedimentos devem ser efetuados:

Primeiro aplicar fatoração LDU à matriz K_p, de modo que

$$K_p = L_1 D_p L_2^T \ , \tag{2.62}$$

onde L_1 e L_2 são matrizes triangulares inferiores e para não haver equívocos, a matriz U da fatoração LDU será representada por L_2^T. Assim, obtêm-se que $D_p = diag\{\Delta_1, \frac{\Delta_2}{\Delta_1}, ..., \frac{\Delta_m}{\Delta_{m-1}}\}$. Onde $\Delta_1, ..., \Delta_m$ são os menores principais de K_p.

Fatorando D_p como

$$D_p = D_+ D \tag{2.63}$$

onde D_+ é uma matriz com entradas positivas, assim podemos reescrever a equação (2.62) como $K_p = L_1 D_+ L_1^T L_1^{-T} D L_2^T$, deste modo a fatoração SDU pode ser satisfeita por

$$S = L_1 D_+ L_1^T \quad , \ U = D^{-1} L_1^{-T} D L_2^T \tag{2.64}$$

Ressalta-se que, D é uma matriz diagonal com entradas $+1$ ou -1. Assim, a fatoração $K_p = SDU$ não será única pois a matriz D_+ introduzida em (2.63) pode apresentar qualquer parâmetro positivo em sua diagonal principal.

Assim, denotando

$$K_p = \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \tag{2.65}$$

Da equação (2.62) temos:

$$L_1 = \begin{bmatrix} 1 & 0 \\ l_1 & 1 \end{bmatrix} \ , \ D_p = \begin{bmatrix} \Delta_1 & 0 \\ 0 & \frac{\Delta_2}{\Delta_1} \end{bmatrix} \ , \ L_2 = \begin{bmatrix} 1 & 0 \\ l_2 & 1 \end{bmatrix} \ , \tag{2.66}$$

onde $l_1 = \frac{k_{21}}{\Delta_1}$ e $l_2 = \frac{k_{12}}{\Delta_1}$. Escolhendo um D_+ e D tal que

$$D_+ = \begin{bmatrix} d_1^+ & 0 \\ 0 & d_2^+ \end{bmatrix}$$

$$D = \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix}$$

a fatoração SDU produz $D = D_+^{-1} D_p$, tal que

$$S = \begin{bmatrix} d_1^+ & d_1^+ \, l_1 \\ d_1^+ \, l_1 & d_2^+ \, + \, d_1^+ \, l_1^2 \end{bmatrix} \quad , \quad D \; = \begin{bmatrix} \frac{\Delta_1}{d_1^+} & 0 \\ 0 & \frac{\Delta_2}{d_2^+ \Delta_1} \end{bmatrix} \quad , \quad U = \begin{bmatrix} 1 & l_2 \, - \, \frac{d_1^+ l_1 \Delta_2}{d_2^+ \Delta_1^2} \\ 0 & 1 \end{bmatrix} \tag{2.67}$$

Substituindo K_p em (2.65) por (2.52), teremos

$$\Delta_1 = f \alpha_1 \cos \phi \quad , \quad \Delta_2 = f^2 \alpha_1 \alpha_2 \tag{2.68}$$

Ao analisar os elementos da matriz D na equação (2.67), em virtude dos sinais de d_1^+ e d_2^+ serem positivos, observa-se que os sinais dos elementos d_1 e d_2 da matriz D dependerão unicamente dos menores principais da matriz K_p , assim

$$sgn(d_1) = sgn\left(\frac{\Delta_1}{d_1^+}\right) \; = \; sgn(\Delta_1)$$

$$sgn(d_2) = sgn\left(\frac{\Delta_2}{d_2^+ \Delta_1}\right) \; = \; sgn\left(\frac{\Delta_2}{\Delta_1}\right)$$

substituindo Δ_1 e Δ_2 , obtêm-se que

$$sgn(d_1) = sgn(f\alpha_1 \cos\phi)$$

$$sgn(d_2) = sgn\left(\frac{f\alpha_2}{\cos\phi}\right)$$

como já havia sido definido anteriormente, f , α_1 , $\alpha_2 > 0$ e $\phi \in]-\frac{\pi}{2}, \frac{\pi}{2}[$. Assim, teremos que $0 < \cos\phi \leq 1$ e consequentemente

$$sgn(d_1) = +1 \quad , \quad sgn(d_2) = +1 \tag{2.69}$$

independente dos valores dos parâmetros f , α_1 , α_2 e ϕ, desde que satisfaçam as condições impostas. O resultado obtido em (2.69) será importante na definição da lei de adaptação.

Após o desenvolvimento da fatoração SDU, o sistema (2.59) ficará da seguinte forma:

$$\dot{\xi}_v = A_v\,\xi_v + B_v\,S\,D\,U\,\tilde{u}\,, \quad e_v = C_v\,\xi_v \tag{2.70}$$

A partir de [6], conclui-se que a fatoração SDU não é única e pode ser escolhida de modo que exista um S que faça a matriz de transferência, obtida a partir da equação (2.70), se tornar SPR. A função de transferência em questão será regida pelas matrizes (A_v , $B_v\,S$ e C_v). Garantida a condição SPR para o sistema do erro aumentado de rastreamento planar, novamente existirão matrizes $P_t = P_t^T > 0$ e $Q_t = Q_t > 0$ que satisfazem a relação:

$$A_v^T P_t \,+\, P_t\,A_v \,=\, -2Q_t \quad , \quad P_t\,B_v\,S \,=\, C_v^T \tag{2.71}$$

Em virtude das modificações no sistema (2.59) é necessário efetuar uma nova parametrização para o sinal de controle, segundo [6]. Com $Uu = (I - I + U)u = Iu - (I - U)u = u - (I - U)u$ nós teremos que $U(u - u^*) = u - U\Phi^*\,w_p - (I - U)u$. Assim a nova parametrização para a entrada ideal será dada por:

$$\underbrace{\begin{bmatrix}\Theta_1^{*T}\,\Omega_1\\ \Theta_2^{*T}\,\Omega_2\end{bmatrix}}_{u^*} \;=\; U\Phi^*\,w_p + (I-U)u \tag{2.72}$$

onde $\Omega_1 \triangleq [w_p^T \quad u_2]^T$ e $\Omega_2 \triangleq w_p$ são os vetores regressores onde $\Omega_1 \in \mathbb{R}^3$ e $\Omega_2 \in \mathbb{R}^2$, já $\Theta_1^* \in \mathbb{R}^3$ e $\Theta_2^* \in \mathbb{R}^2$ são os vetores de parâmetros desconhecidos e u_2 é o segundo elemento do vetor u. Assim, ao inserir a parametrização o sistema (2.70) ficará da seguinte forma:

$$\dot{\xi}_v = A_v\,\xi_v + B_v\,S\,D\,(u\,-\,\Theta^{*T}\Omega)\,, \quad e_v = C_v\,\xi_v \tag{2.73}$$

onde $\Theta^* = [\Theta_1^{*T} \quad \Theta_2^{*T}]^T$, $\Omega = [\Omega_1^T \quad \Omega_2^T]^T$ e $\tilde{u} = u - \Theta^{*T}\Omega$.

Como consequência da nova parametrização para o controle ideal u^* e da fatoração SDU, temos que o controle real será dado por:

$$u \;=\; \left[\Theta_1^T\,\Omega_1 \quad \Theta_2^T\,\Omega_2\right]^T \;=\; \Theta^T\Omega \tag{2.74}$$

assim, Θ será a estimativa do parâmetro Θ^*.

Lei de Adaptação

Após a definição da lei de controle e efetuada a parametrização com base na fatoração SDU, agora será definida a lei de adaptação que fará o vetor de parâmetros estimados Θ rastrear Θ^* . Da mesma maneira, como foi definido no rastreamento em profundidade, denotaremos $\tilde{\Theta} = \Theta - \Theta^*$.

Deve-se ressaltar que, a necessidade de obtenção do sinal da matriz K_p para o desenvolvimento da lei de adaptação, foi flexibilizada de maneira perspicaz através da fatoração SDU. Portanto, com a matriz D diagonal unitária, a extração do sinal se torna trivial.

Analogamente ao rastreamento em profundidade, utilizou-se para a lei de adaptação a abordagem SPR-Lyapunov. Assim, para garantir que ξ_v seja uniformemente limitado e a sua

convergência assintótica, teremos:

$$\dot{\Theta}_1 \;=\; -\gamma_1\, sgn(d_1)\, e_{v1}\, \Omega_1$$

$$\dot{\Theta}_2 \;=\; -\gamma_2\, sgn(d_2)\, e_{v2}\, \Omega_2 \tag{2.75}$$

tal que (d_1 , d_2) são os elementos da diagonal da matriz D e (γ_1 , γ_2) representam os ganhos de adaptação. Substituindo (2.69) em (2.75), obtêm-se

$$\dot{\Theta}_1 \;=\; -\gamma_1\, e_{v1}\, \Omega_1$$

$$\dot{\Theta}_2 \;=\; -\gamma_2\, e_{v2}\, \Omega_2 \tag{2.76}$$

Análise de Estabilidade

Após a definição das leis de controle e adaptação, é necessário efetuar uma análise sobre a convergência do sistema (2.73). Novamente, utilizando a teoria de Lyapunov, introduz-se a seguinte função:

$$2V(\xi_v\,,\,\tilde{\Theta}) = \xi_v^T P_t \xi_v \quad + \quad \sum_{i=1}^{2} \frac{1}{\gamma_i}\, |d_i|\, \tilde{\Theta}_i^T\, \tilde{\Theta}_i \tag{2.77}$$

derivando a equação (2.77) e substituindo por (2.70) , (2.75) e (2.71) teremos:

$$2\dot{V}(\xi_v\,,\,\tilde{\Theta}) = \dot{\xi_v^T} P_t \xi_v \;+\; \xi_v^T P_t \dot{\xi_v} \;+\; \sum_{i=1}^{2} \frac{2}{\gamma_i}\, |d_i|\, \tilde{\Theta}_i^T\, \dot{\Theta}_i$$

$$= (\xi_v^T A_v^T \;+\; \tilde{u}^T D\, S\, B_v^T) P_t \xi_v \;+\; \xi_v^T P_t (A_v \xi_v \;+\; B_v S D \tilde{u}) \;+\; \sum_{i=1}^{2} \frac{1}{\gamma_i}\, |d_i|\, 2\tilde{\Theta}_i^T\, (-\gamma_i sgn(d_i) e_{vi} \Omega_i)$$

$$= \xi_v^T A_v^T P_t \xi_v \;+\; \xi_v^T P_t A_v \xi_v \;+\; \underbrace{\tilde{u}^T D\, S\, B_v^T P_t \xi_v}_{escalar} \;+\; \underbrace{\xi_v^T P_t B_v S D \tilde{u}}_{escalar} \;-\; 2\sum_{i=1}^{2} d_i\, e_{vi} \tilde{\Theta}_i^T\, \Omega_i$$

$$= \xi_v^T (A_v^T P_t \;+\; P_t A_v) \xi_v \;+\; 2\xi_v^T \underbrace{P_t B_v S}_{C_v^T} D\tilde{u} \;-\; 2\sum_{i=1}^{2} e_{vi}\, d_i\, \tilde{\Theta}_i^T\, \Omega_i \tag{2.78}$$

tendo em vista que:

$$P_t B_v S = C_v^T \tag{2.79}$$

$$\xi_v^T C_v^T = e_v^T \tag{2.80}$$

$$\tilde{u} = \tilde{\Theta}^T \Omega \tag{2.81}$$

manipulando o termo $\xi_v^T P_t B_v S D \tilde{u}$ utilizando as equações anteriores, teremos:

$$\xi_v^T P_t B_v S D \tilde{u} = \xi_v^T C_v^T D \tilde{u}$$

$$= e_v^T D \tilde{u}$$

$$= e_v^T D \tilde{\Theta}^T \Omega$$

$$\xi_v^T P_t B_v S D \tilde{u} = \sum_{i=1}^{2} e_{vi}\, d_i\, \tilde{\Theta}_i^T\, \Omega_i \tag{2.82}$$

Retomando a função de Lyapunov (2.78) e utilizando a equação (2.82) , os dois últimos termos se cancelarão e substituindo o termo $A_v^T P_t + P_t A_v = -2Q_t$, o resultado será:

$$2\dot{V} = -2\xi_v^T (Q_t) \xi_v$$

$$\dot{V} = -\xi_v^T (Q_t) \xi_v \leq 0 \tag{2.83}$$

Como havia sido definido em (2.71) a matriz Q_t é simétrica e definida positiva. Assim a derivada da função candidata de Lyapunov será sempre menor ou igual a zero, garantindo assim a estabilidade para o sistema composto por (2.73),(2.76). Pela teoria de Lyapunov as equações

(2.77) e (2.83) demonstram que o erro e_v e o parâmetro de adaptação Θ são globalmente limitados, ressalta-se ainda que em virtude do sistema (2.56) ser BIBO estável então e_{vf} também será limitado. A partir das definições de Ω_1 e Ω_2 efetuadas anteriormente, têm-se que ambos são limitados, consequentemente pela equação (2.74) temos que u também será limitado.

Calculando a derivada segunda da função de Lyapunov (2.77), obtêm-se:

$$\begin{aligned} \ddot{V} &= -\dot{\xi_v}^T Q_t \xi_v - \xi_v^T Q_t \dot{\xi}_v \\ &= -2\xi_v^T Q_t \dot{\xi}_v \end{aligned} \tag{2.84}$$

Deve-se ressaltar que, o sistema composto por (2.70),(2.76) não só garante a limitação de e_v , como também a limitação de todo o estado ξ_v [15], como pode ser observado pela equação (2.83).

Em posse dessas duas informações, a partir da equação (2.70) temos que $\dot{\xi_v}$ será limitado e observando a equação (2.84) temos que $\ddot{V}$ também será limitada, consequentemente $\dot{V}$ apresenta continuidade uniforme.

Após obter o resultado da continuidade uniforme da função $\dot{V}$, aplica-se o lema de Barbalat e assim concluímos que o erro e_v converge assintoticamente para zero. Desta maneira, as leis de controle e de adaptação aplicadas, permitem o rastreamento planar 2D do alvo.

Deve-se ressaltar que, tanto para o rastreamento planar quanto para o rastreamento em profundidade, as leis de adaptação e de controle não garantem que os erros paramétricos convirjam para zero, garantindo apenas a sua limitação. De acordo com [15] a convergência para zero só ocorrerá se os sinais de entrada dos modelos de referência apresentarem uma condição denominada "excitação persistente".

Após o desenvolvimento das estratégias de controle tanto para o rastreamento planar quanto para o rastreamento em profundidade, observou-se, através da abordagem SPR-Lyapunov, que o sistema servo visual completo apresenta convergência assintótica para zero em ambos os erros de rastreamento, comprovando assim o estudo desenvolvido.

2.4 Resultados de Simulações

Assim como em diversos campos do conhecimento, uma das maneiras mais utilizadas para se compreender o comportamento de um sistema dinâmico e validar os resultados de um estudo é através da simulação. O processo de simulação exige primeiramente que o modelo do sistema de interesse esteja representado através das equações dinâmicas que o regem. Quanto mais próxima da realidade for o modelo do sistema de interesse, mais próximos do sistema real serão os resultados da simulação. Após a etapa de modelagem , deve-se utilizar alguma ferramenta computacional de alto nível que permita efetuar a resolução das equações em função das entradas a serem definidas. Um dos softwares mais comumente usados para simulação é o já mencionado Matlab .

Desta maneira para consolidar e avaliar o estudo proposto por [10], foi efetuada a simulação do sistema servo visual completo, isto é , modelo e controle adaptativo servo visual, através do software Matlab.

Inúmeras simulações foram efetuadas até que os resultados fossem satisfatórios, confirmando assim o estudo proposto. A escolha dos parâmetros iniciais, ganhos de controle dentre outros, foram definidos empiricamente, através das simulações e da observação dos resultados. A seguir, serão demonstradas três simulações consideradas relevantes tanto para a compreensão do estudo proposto, quanto para a comprovação da convergência da estratégia de controle implementada.

Os parâmetros comuns a todas as simulações são apresentados abaixo:

Os comprimentos dos elementos de ligação entre as juntas são $l_0 = 1337mm$, $l_1 = 380mm$, $l_2 = 390mm$, $l_3 = 330mm$

Coordenadas da posição da câmera em relação à referência da base do manipulador:

$T = [-571.4 \quad 1216.3 \quad -1760]^T$

$q(0) = [-\pi \quad -\frac{\pi}{4} \quad \frac{\pi}{2}]\ rad$

$f = 2\ mm$

$\alpha_1 = 401.28\ pixel/mm$

$\alpha_2 = 401.28\ pixel/mm$

$w_n = \frac{\pi}{10} \quad rad/s$

$z_{c0} = 1760\ mm$

$A_{v0} = 760\ pixel^2$

$rp = 60\ pixel$

$ra = 100\ pixel$

$\lambda_m = 1$

$\lambda_M = 1$

A primeira simulação demonstrada tem como característica básica a utilização de uma referência constante para o rastreamento em profundidade e uma referência senoidal com duas componentes de frequência para o rastreamento planar. Os parâmetros referentes a esta simulação seguem abaixo:

Sinal de referência em profundidade : $A_v^* = 840\ pixels^2$

Sinal de referência planar: $x_v^* = r_p\,[\sin(2w_n\,t) \quad \cos(w_n\,t)\,]^T + [\,-77.9794 \quad -87.3446\,]^T\ pixels$

Tempo de simulação: $45\ s$

Parâmetros utilizados:

$x_{v0} = [-87.97 \quad -47.34]\ pixel$

$A_{vm0} = 840\ pixel^2$

$x_{vm0} = [-77.97 \quad -27.34]\ pixel$

$\lambda = 10$

$\Theta_1(0) = [40.45 \quad 0.0 \quad 0.0]$

$\Theta_2(0) = [0.0 \quad 60.45]$

$\Theta_3(0) = [9704.0]$

$\gamma_1 = 4.0$

$\gamma_2 = 60.0$

$\gamma_3 = 10^6$

$\phi = -\frac{\pi}{6}\ rad$

Ao analisar o gráfico de rastreamento planar da trajetória de referência representado na figura 2.7 , observa-se em uma primeira instância que estratégia de controle adotada permitiu o rastreamento planar após um transiente, contudo a convergência dos erros de rastreamento podem apenas ser comprovados a partir dos gráficos da figura 2.8. Como pode ser observado o erro de rastreamento planar apresentou valores em torno de $2\ pixels$ em módulo o que para fins práticos nos permite assumir a convergência assintótica para zero e o erro de rastreamento em profundidade convergiu visivelmente para zero. Fato que comprova a eficácia da lei de controle servo visual aplicada.

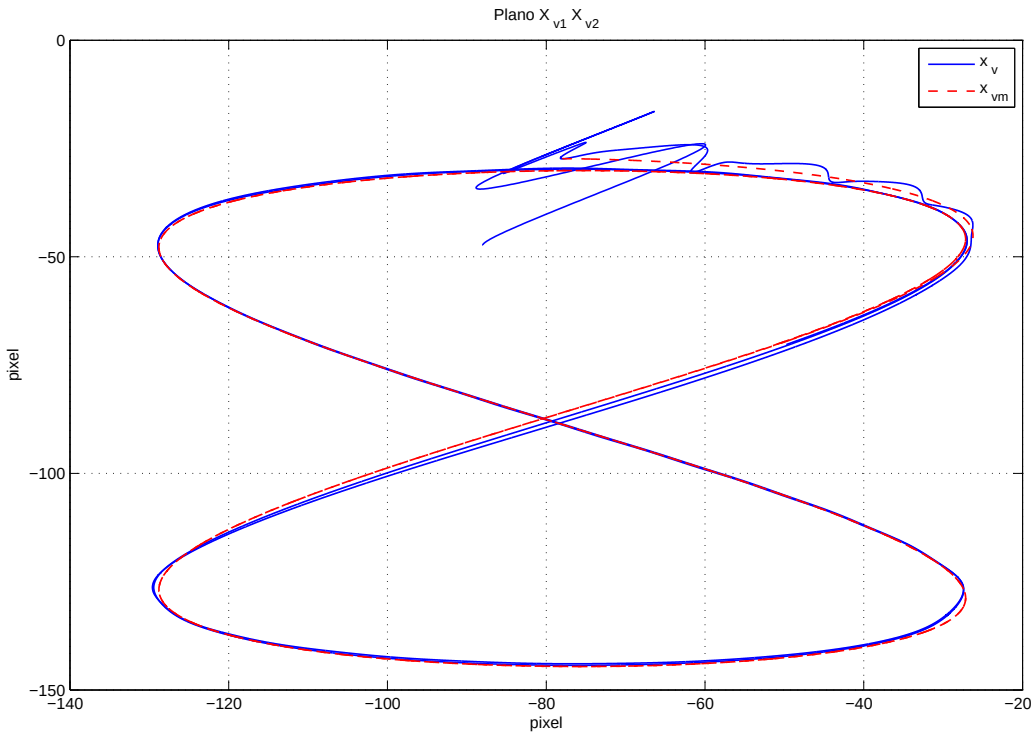

Figura 2.7: Gráfico das trajetórias real e de referência para o rastreamento planar, obtido em simulação com ganhos de adaptação $\gamma_1 = 4.0$, $\gamma_2 = 60.0$

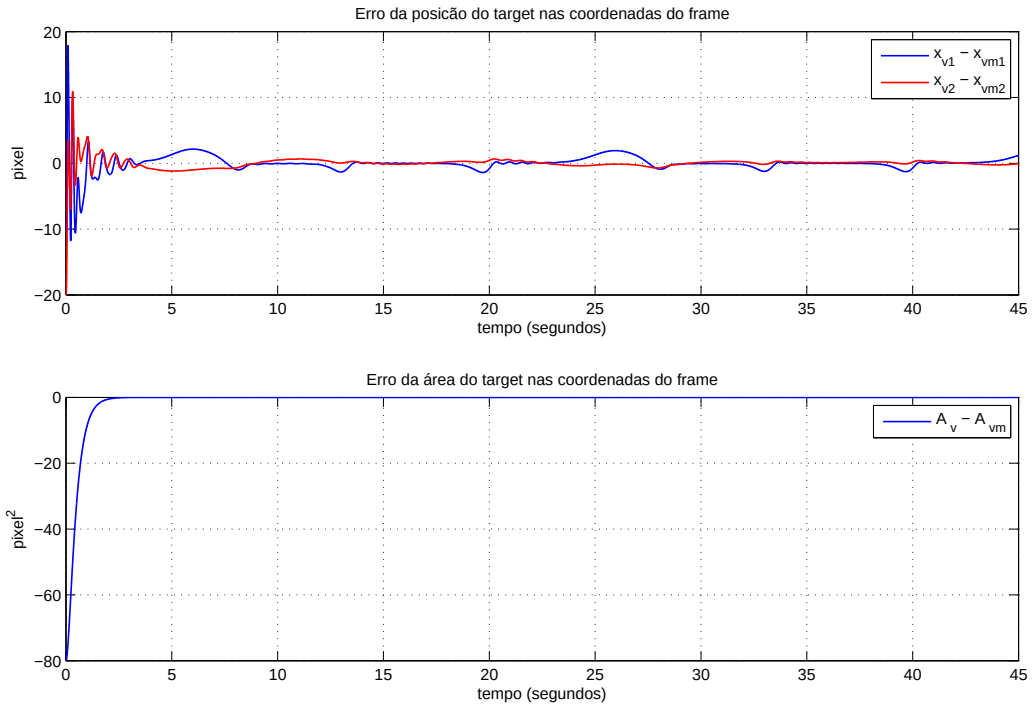

Figura 2.8: Gráfico do erro de rastreamento planar e em profundidade obtido em simulação com ganhos de adaptação $\gamma_1 = 4.0$, $\gamma_2 = 60.0$, $\gamma_3 = 10^6$

A partir dos gráficos dos parâmetros de adaptação Θ_1 e Θ_2 (figura 2.9), observa-se que seus valores tenderam a convergir para patamares constantes. O gráfico do parâmetro de adaptação θ_3 (figura 2.10) apresentou uma rápida convergência para um patamar constante. Portanto, a limitação dos parâmetros de adaptação e consequentemente a eficácia das leis de adaptação para o rastreamento planar e em profundidade são comprovadas.

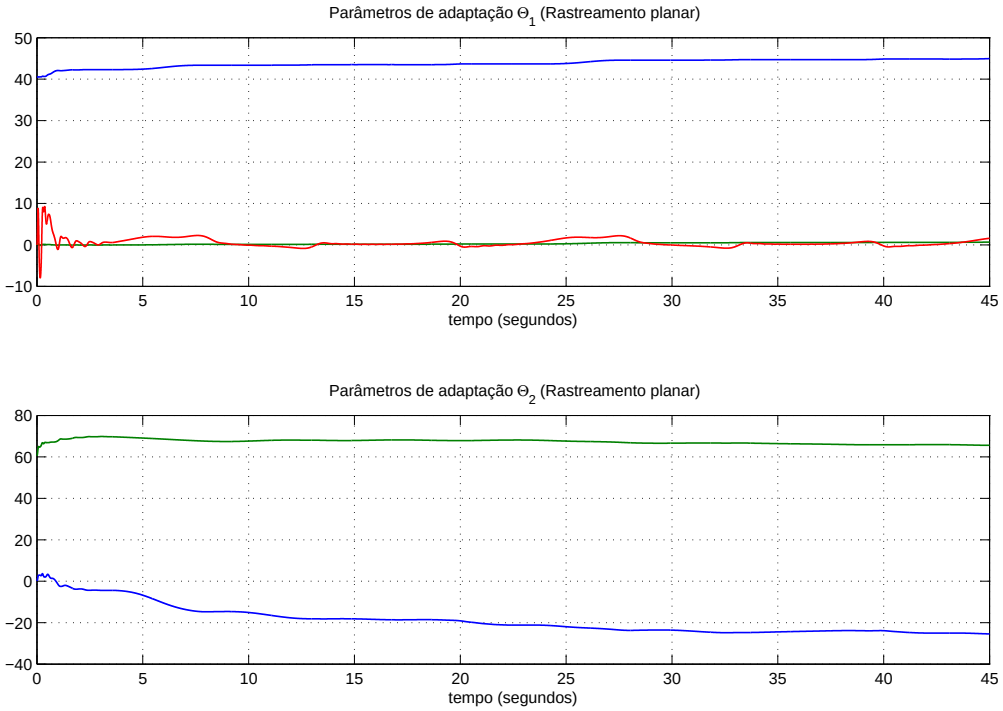

Figura 2.9: Gráfico da estimação dos parâmetros Θ_1 e Θ_2 desconhecidos, obtido em simulação com ganhos de adaptação $\gamma_1 = 4.0$, $\gamma_2 = 60.0$

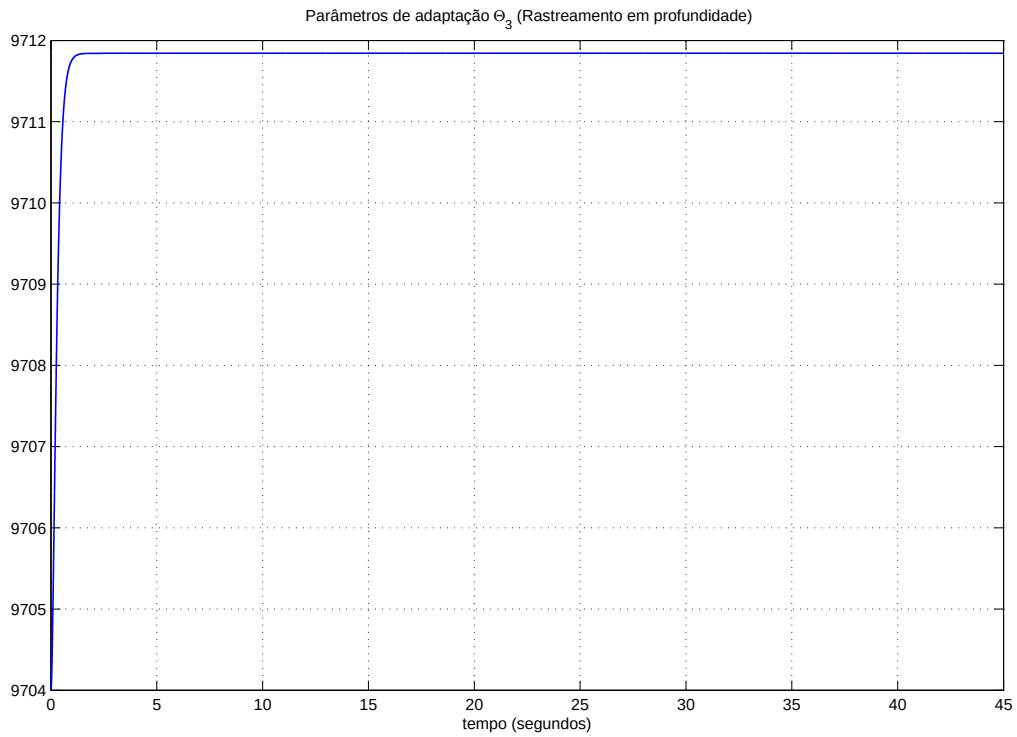

Figura 2.10: Gráfico da estimação do parâmetro θ_3 desconhecido, obtido em simulação com ganho de adaptação $\gamma_3 = 10^6$

A segunda simulação demonstrada tem como característica básica a utilização de uma referência senoidal com apenas uma componente de frequência para o rastreamento em profundidade e outra referência senoidal com duas componentes de frequência para o rastreamento planar. Para as novas condições, os parâmetros referentes a esta simulação seguem abaixo:

Sinal de referência em profundidade : $A_v^* = r_a \, \sin(\frac{w_n}{2} \, t) \; + \; 80 \; pixel^2$

Sinal de referência planar: $x_v^* = r_p \, [\sin(2w_n \, t) \quad \cos(w_n \, t)\,]^T + [\,-57.9794 \quad -87.3446\,]^T \; pixel$

Tempo de simulação: $25 \; s$

Parâmetros utilizados:

$x_{v0} = [-87.97 \quad -47.34] \; pixel$

$A_{vm0} = 840 \; pixel^2$

$x_{vm0} = [-57.97 \quad -27.34]\ pixel$

$\lambda = 10$

$\Theta_1(0) = [60.45 \quad 0.0 \quad 0.0]$

$\Theta_2(0) = [0.0 \quad 60.45]$

$\Theta_3(0) = [9704.0]$

$\gamma_1 = 0.5$

$\gamma_2 = 60.0$

$\gamma_3 = 10^{10}$

$\phi = -\frac{\pi}{6}\ rad$

O gráfico de rastreamento planar da trajetória de referência representado na figura 2.11 demonstra novamente a presença de um transiente e logo após observa-se que o rastreamento planar foi efetuado com sucesso. A convergência dos erros de rastreamento são comprovados a partir dos gráficos da figura 2.12, visto que para o rastreamento planar os erros apresentaram valor em torno de $2\ pixels$ em módulo e o erro de rastreamento em profundidade apresentou valor menor que $1\ pixel^2$. A partir destes resultados comprova-se a convergência dos erros de rastreamento para zero e consequentemente a eficácia da estratégia de controle servo visual aplicada.

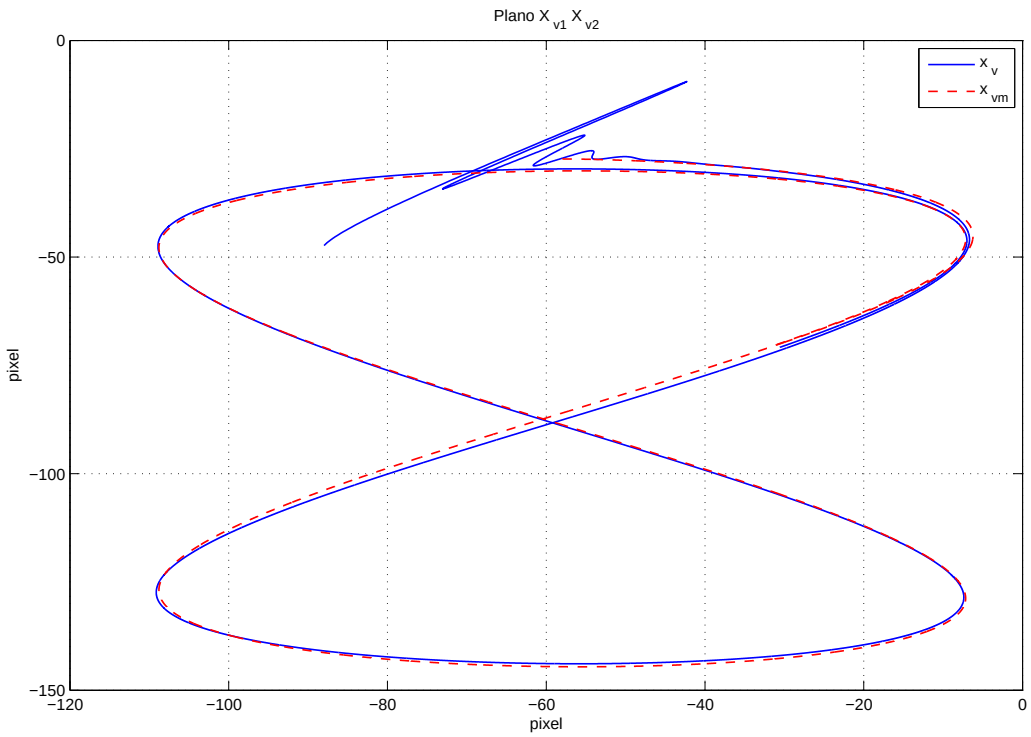

Figura 2.11: Gráfico das trajetórias real e de referência para o rastreamento planar, obtido em simulação com ganhos de adaptação $\gamma_1 = 0.5$, $\gamma_2 = 60.0$

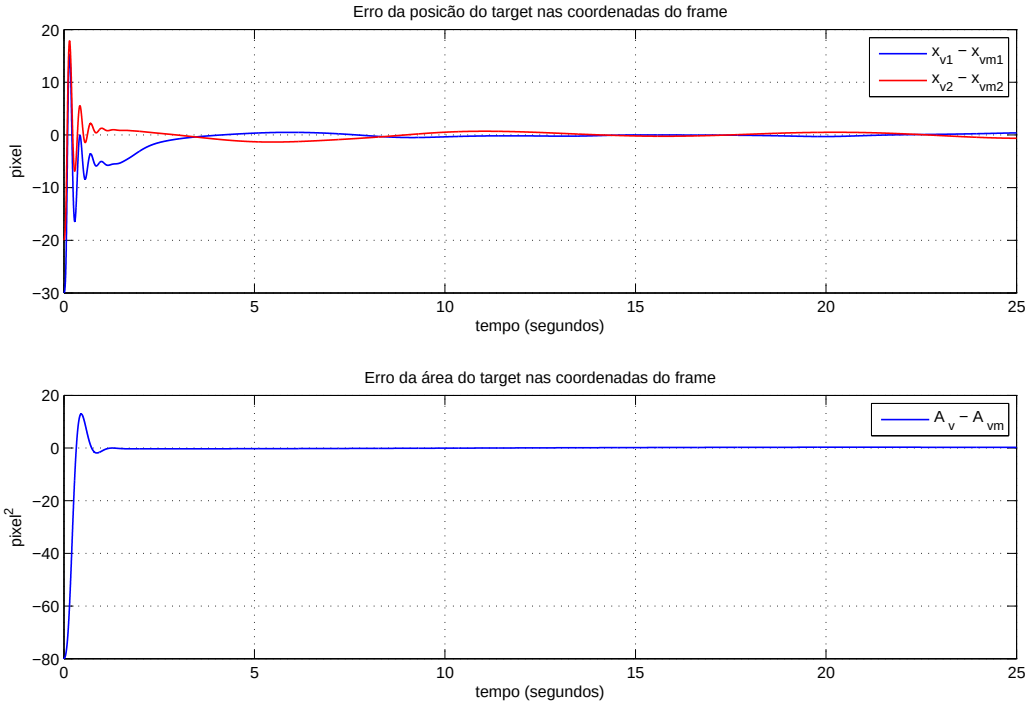

Figura 2.12: Gráfico do erro de rastreamento planar e em profundidade, obtido em simulação com ganhos de adaptação $\gamma_1 = 0.5$, $\gamma_2 = 60.0$, $\gamma_3 = 10^{10}$

Para as novas condições, os gráficos dos parâmetros de adaptação Θ_1 e Θ_2 observados na figura 2.13, demonstram que seus valores novamente tendem a convergir para patamares constantes. O gráfico do parâmetro de adaptação Θ_3 (figura 2.14) apresentou uma rápida convergência para um valor constante. Desta maneira, a limitação dos parâmetros de adaptação e a efetividade das leis de adaptação podem ser validadas.

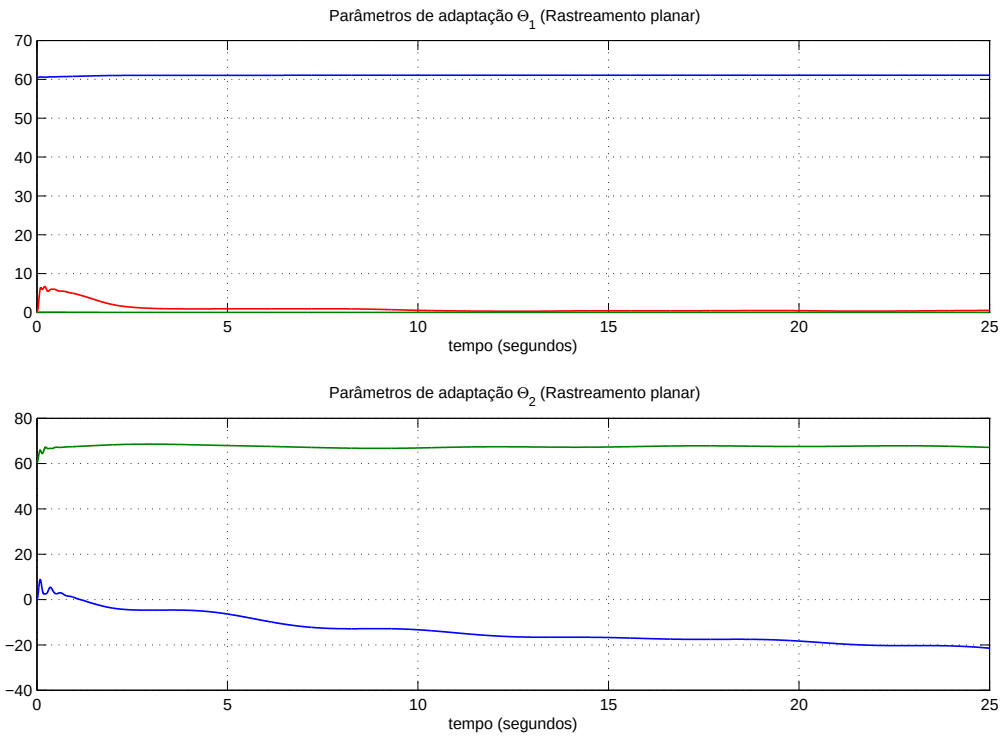

Figura 2.13: Gráfico da estimação dos parâmetros Θ_1 e Θ_2 desconhecidos, obtido em simulação com ganhos de adaptação $\gamma_1 = 0.5$, $\gamma_2 = 60.0$

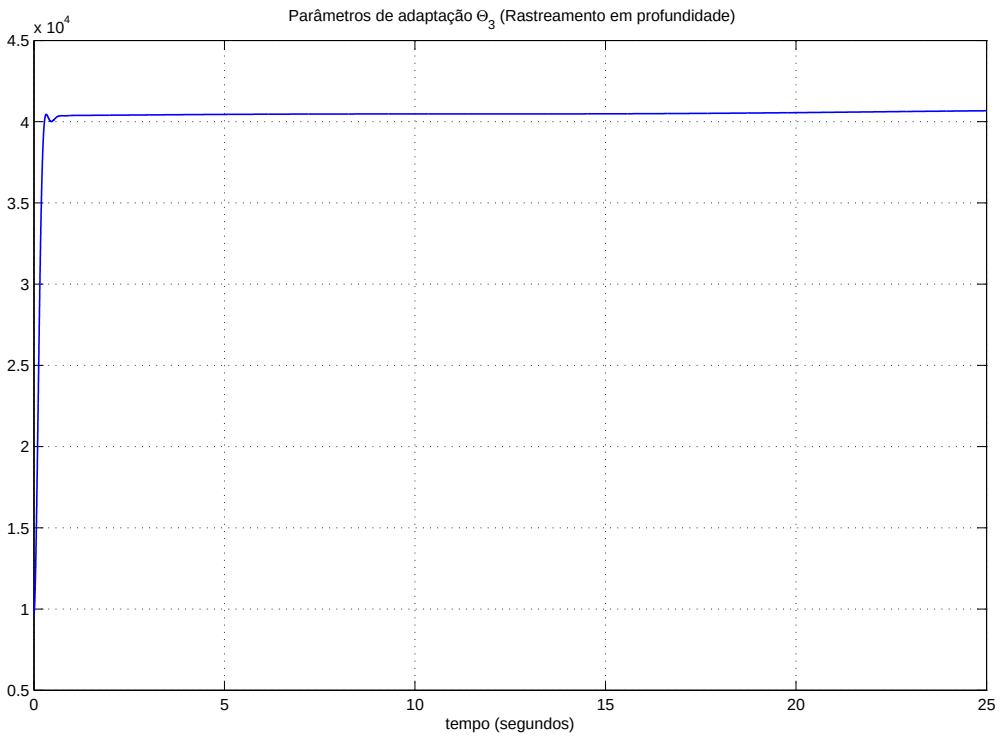

Figura 2.14: Gráfico da estimação do parâmetro θ_3 desconhecido, obtido em simulação com ganho de adaptação $\gamma_3 = 10^{10}$

Como pôde ser observado através dos gráficos das simulações demonstrados anteriormente, a abordagem de Controle Servo Visual proposta, apresentou resultados satisfatórios tanto para o rastreamento planar quanto para o rastreamento em profundidade.

Ao simular o sistema completo, modelo e controle é necessário dar a devida atenção ao espaço de trabalho do manipulador, isto é, caso a trajetória de referência ultrapasse a região máxima a qual o manipulador pode alcançar, o sistema instabilizará. Para verificar se a trajetória de referência se localizava dentro do espaço de trabalho do manipulador, criou-se uma rotina no Matlab que implementa o sistema servo visual e o modelo cinemático do manipulador. Em seguida, é desenhado em um mesmo gráfico, a trajetória de referência da imagem convertida para o sistema de coordenadas real da base do manipulador e o trajeto máximo que o manipulador pode alcançar. Assim, caso a trajetória de referência ultrapasse a trajetória máxima do manipulador, então significa que o caminho de referência está fora do espaço de trabalho do manipulador, fazendo com que o sistema caia em uma singularidade, divergindo.

Outro aspecto importante observado nas simulações, foi que ao definir os valores inici-

ais dos ganhos de adaptação com uma discrepância muito elevada em relação aos valores de convergência, também é notória a influência destes valores iniciais no aumento das oscilações da posição do alvo e do erro de rastreamento. Assim, a sintonia dos ganhos de adaptação não são triviais de se definir, já que há um compromisso entre aumentar os valores dos ganhos para tentar alcançar a convergência dos parâmetros e erros de rastreamento e ao mesmo tempo evitar que este aumento produza oscilações indesejáveis ou até mesmo provoque a instabilização do sistema.

Por fim, observou-se que ao aumentar o parâmetro λ, presente nos respectivos erros de rastreamentos filtrados, a performance do controle também apresentava uma melhora expressiva, resultando em erros de rastreamentos menores com uma redução do tempo de convergência.

Capítulo 3

Implementação Prática

A etapa de implementação prática de um estudo representa a consolidação da análise que esta sendo efetuada e a comprovação dos resultados obtidos no desenvolvimento teórico. Desta maneira, o capítulo atual irá abordar os componentes que compõem o sistema servo visual, além de descrever sua aplicação e integração, permitindo a implementação prática do estudo proposto. Por fim, serão demonstrados e analisados os resultados dos experimentos.

3.1 Sistema de Controle Servo Visual

O sistema servo visual consiste nos elementos que oferecem suporte e permitem a aplicação real de estratégia de controle servo visual estudada.

Como já discutido em tópicos anteriores, um dos componentes principais da Servo Visão é aquele que fornece os dados do ambiente para a realimentação do controle, a câmera. Desta maneira, a primeira ação executada na implementação prática consiste em extrair as informações úteis da imagem rudimentar. Sendo assim, esta seção abordará conceitos importantes como visão computacional e suas características, além de abordar os componentes do sistema servo visual reais utilizados na implementação prática deste estudo.

3.1.1 Visão Computacional

A visão computacional consiste em aplicar técnicas e métodos que permitem sistemas computacionais interpretar imagens e utilizá-las em um contexto específico. A interpretação de uma imagem pode ser definida como transformação dos dados digitais da representação da imagem em dados semânticos úteis ao sistema. Analogamente, no Sistema Servo Visual

Baseado em Imagem, muitos dos métodos e técnicas em visão computacional são utilizados, sendo o objetivo desta subseção comentar os métodos utilizados na implementação prática do problema proposto.

Definição do Sensoriamento

A imagem a ser analisada pelas técnicas e procedimentos em computação visual são obtidas das mais diversas formas. De acordo com a aplicação e com suas características, é que podem ser definidas o tipo de sensor a ser utilizado.

Os mais comumente usados são as câmeras sensíveis a luz, equipamentos de tomografia, radiologia, radar, câmeras ultrassônicas e infravermelho. Cada tipo de sensor mencionado irá produzir uma imagem em 2D com suas peculiaridades e informações importantes, que posteriormente serão extraídas para um fim específico.

Para a servo visão os sensores usados quase que em sua totalidade são as câmeras coloridas sensíveis a luz. Isto se deve à proximidade com a realidade que a imagem produzida oferece e a sua relativa simplicidade, desta maneira pode-se definir cores e regiões de interesse na imagem para a o posterior processamento e obtenção das características relevantes.

Pré-Processamento de imagem

Após a aquisição da imagem, a partir do sensor escolhido é necessário verificar se a imagem atende a alguns requisitos mínimos que permitam a posterior extração das informações relevantes. A imagem bruta extraída diretamente dos sensores, podem apresentar uma série de imperfeições como: ruídos oriundos do sensor, contraste reduzido, problemas de escala, dentre outros. Deste modo, a etapa de pré-processamento tem por objetivo utilizar métodos para eliminar ou reduzir estas imperfeições e fornecer uma imagem melhorada que atenda às necessidades que a aplicação exige.

Para os Sistemas Servo Visuais de um modo geral as principais ações realizadas na etapa de pré-processamento são a redução do ruído proveniente da câmera e melhorias da qualidade da imagem através de alterações no contraste e luminosidade. Contudo, cada tipo de abordagem Servo Visual e cada modelo de câmera utilizada irão implicar em diferentes necessidades.

Como já discutido anteriormente, o controle servo visual aplicado neste trabalho exige a

extração da área e centroide do alvo projetado na imagem. Assim, em virtude do ruído inerente a qualquer câmera, o problema proposto exigirá que a imagem seja filtrada de forma a impedir que este ruído esteja inserido nas informações extraídas.

Existem inúmeros métodos para filtragem de imagens e consequentemente melhoria da qualidade. Entretanto, não é foco deste trabalho detalhá-los. Desta maneira, será descrito um importante método para redução ruídos em imagem que foi utilizado na etapa de implementação prática, o filtro mediano.

O filtro mediano é um filtro não linear muito utilizado na redução de ruídos provenientes dos detectores das câmeras, os ruídos do tipo *"salt and peper"* ou traduzido para língua portuguesa ruídos "sal e pimenta". Este tipo de ruídos são pequenas granulações pontuais que surgem na imagem digitalizada e que são comuns em sensores de câmeras.

O filtro mediano consiste basicamente em classificar os pixels em uma vizinhança de acordo com seu valor de brilho (*brightness*) e após isto, definir a intensidade do brilho do pixel central como o valor da mediana dos valores classificados na primeira operação [13]. Desta maneira, ao aplicar o método a todos os pixels da imagem, aqueles que acidentalmente haviam apresentado uma modificação extrema no seu valor de luminosidade serão alterados por um valor "razoável", a mediana da vizinhança. A forma como se escolhe a vizinhança de pixels pode apresentar diferentes padrões como quadrados ou octais de diferentes quantidades de pixels. Estes padrões por sua vez irão influenciar na eficiência da redução do ruído e no nível de alteração que a imagem sofrerá.

Assim, o filtro mediano é uma boa alternativa para reduzir ruídos aleatórios e a sua principal vantagem reside no fato de que sua ação efetua o mínimo de degradação nas bordas presentes na imagem. Por este motivo, na implementação prática do estudo proposto foi utilizado um filtro mediano, visto que a presença de ruídos do tipo mencionado degradariam a precisão das medidas do centroide e da área do objeto alvo. O padrão de pixels da vizinhança foi escolhido empiricamente, preocupando-se sempre em reduzir os ruídos e não degradar as bordas da imagem.

Alguns ajustes de luminosidade e contraste também foram efetuados na implementação prática, todos utilizando o método empírico.

Finalizada a etapa de pré-processamento em que a imagem é alterada e melhorada de forma a atender às necessidades da aplicação, é iniciado o processo para extração das características relevantes.

Detecção de cor e Segmentação

A detecção de cor e a segmentação são processos extremamente importantes para o posterior reconhecimento e extração de informações das imagens. A detecção de cor consiste em determinar quais as regiões da imagem apresentam a coloração dentro de um intervalo específico de tons vermelhos, por exemplo. Após a definição das cores de interesse é necessário classificar e agrupar os pixels das regiões de interesse, processo este denominado segmentação.

Com o objetivo de facilitar a detecção de cor, a maior parte dos sistemas servo visuais utilizam um alvo pintado com uma das três cores primitivas, mais comumente o vermelho. Para este trabalho não foi diferente, utilizou-se uma esfera homogênea de cor vermelha como alvo.

A detecção de cor foi desenvolvida, utilizando o algoritmo da distância euclidiana. Neste método, os valores RGB de cada pixel serão os elementos de um vetor com três posições, desta maneira os seguintes passos serão executados:

1º etapa: Define-se o valor RGB de referência para a cor vermelha, neste caso:

$$v_r = \begin{bmatrix} 255 \\ 0 \\ 0 \end{bmatrix} \tag{3.1}$$

2º etapa: Denota-se p_{ij} o vetor com os valores RGB do pixel que apresenta as coordenadas (j, i) do sistema de referência da matriz que representa a imagem. Após, calcula-se a distância euclidiana D_{ij} de cada pixel da imagem, da seguinte maneira:

$$D_{ij} = ||v_r - p_{ij}|| \tag{3.2}$$

3º etapa: Após, cálculo da distância euclidiana de cada pixel, define-se um valor de referência R, para comparar as distâncias e mapear os pixels em valores binários denominados t_{ij}. Caso o D_{ij} de cada pixel seja maior que a distância de referência R, atribuímos o valor 0 a

este pixel, caso contrário atribuímos 1. A representação matemática deste procedimento segue abaixo:

$$t_{ij} = \begin{cases} 1 \quad , \quad se \quad D_{ij} \leq R; \\ 0 \quad , \quad se \quad D_{ij} > R; \end{cases}$$

Assim, a projeção da área do alvo na imagem será representada pelos pixels com valor 1 e o restante da imagem por pixels com valor 0.

Reconhecimento e extração de informações em imagem

Após a separação da imagem em regiões de interesse bem definidas, é necessário escolher quais características serão extraídas para poder utilizá-las de acordo com as necessidades da aplicação.

Desta maneira, características geométricas como perímetro, área, centroide ou até mesmo análises em espectro da frequência podem ser obtidas da imagem utilizando métodos e técnicas distintas para cada informação.

Como já foi discutido na etapa de modelagem do sistema servo visual, neste estudo as duas informações mais importantes que devem ser obtidas da imagem serão a posição do centroide e área do alvo esférico colorido, o que nos permitirá, consequentemente, efetuar o rastreamento planar e em profundidade. Desta maneira, a partir de [13] escolheu-se o método dos momentos da imagem para a extração de ambas as características.

Para uma função 2D contínua $f(x, y)$ o momento de ordem zero M_0 é calculado da seguinte maneira:

$$M_0 = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) dx dy \tag{3.3}$$

Analogamente, denotando como $I(j, i)$ a função que mapeia as coordenadas (j, i) da imagem nos valores binários 0 ou 1, atribuídos de acordo com a segmentação efetuada no passo anterior, temos que o momento da imagem m_0 será:

$$m_0 = \sum_{0}^{L} \sum_{0}^{H} I(j, i) \tag{3.4}$$

Assim, o valor m_0 será a área da alvo em $pixels^2$. Já as coordenadas (cx, cy) do centroide, representadas no sistema de referência O_f da imagem capturada, serão obtidas a partir de:

$$m_{10} = \sum_{0}^{L} \sum_{0}^{A} jI(j,i) \tag{3.5}$$

$$m_{01} = \sum_{0}^{L} \sum_{0}^{A} iI(j,i) \tag{3.6}$$

$$cx = \frac{m_{10}}{m_0} \tag{3.7}$$

$$cy = \frac{m_{01}}{m_0} \tag{3.8}$$

Em que L e A representam as dimensões em pixels da largura e altura do quadro da imagem respectivamente.

Observação: Quando a imagem é capturada, as informações das cores de cada pixel são armazenadas em uma matriz com as mesmas dimensões do quadro da imagem, para o nosso caso $(640\text{x}480)$ pixels. Desta maneira, a posição de cada pixel da imagem capturada será dado em relação à extremidade superior esquerda do quadro, sendo denotado como o sistema de referência O_f.

Ao efetuar o cálculo do centroide, haverá a necessidade de aplicar uma simples transformação de coordenadas para representar esta informação no sistema de coordenadas O_v utilizado na modelagem, assim como no controle servo visual desenvolvido. A representação dos dois sistemas de coordenadas pode ser observada na figura 3.1. Assim, a transformação dos sistemas será dada por:

$$\begin{bmatrix} x_{v1} \\ x_{v2} \end{bmatrix} = \begin{bmatrix} 320 - j \\ 240 - i \end{bmatrix} \tag{3.9}$$

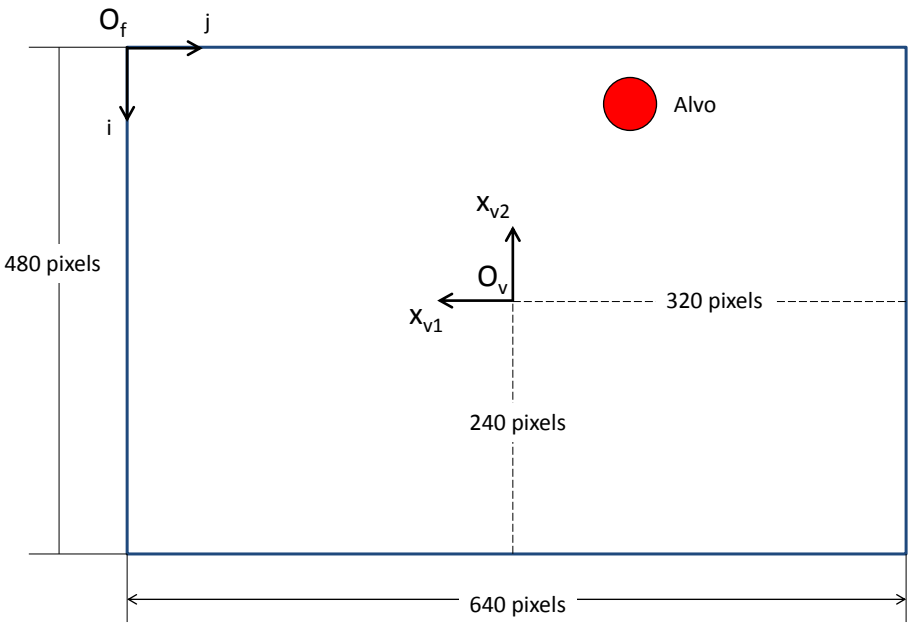

Figura 3.1: Representação do sistema de coordenadas O_f relacionado a matriz da imagem capturada e o sistema de coordenadas da projeção da imagem O_v

Deve ser ressaltado que na implementação prática, algumas ações foram executadas para aumentar a eficiência no processo de segmentação, detecção de cor e extração das informações. Para poder eliminar possíveis objetos com cor vermelha no fundo da imagem que pudessem interferir na extração das informações, duas imagens consecutivas com o alvo em posições diferentes foram capturadas. Em seguida, foi efetuada a detecção de cor e segmentação de ambas e o procedimento de extração das informações foi desenvolvida na imagem resultante da subtração das duas imagens binárias. Desta maneira, os possíveis objetos de cor vermelha presentes no fundo da imagem desaparecem na operação de subtração, restando apenas o alvo na imagem resultante.

Outra ação para a melhoria da eficiência, consistiu em utilizar quadros de apenas $(100\text{x}100)$ pixels para processar a imagem. Para realizar tal tarefa, após definir o centroide e a área da imagem segmentada resultante da subtração, reduz-se o quadro, originalmente de $(640\text{x}480)$, para uma janela de $(100\text{x}100)$ cujo centro se encontra na mesma posição do centroide do alvo. Desta maneira, a cada iteração o quadro reduzido se move em conjunto com o movimento do centroide do alvo, e todo o procedimento de processamento da imagem é efetuado no quadro reduzido, o

que aumenta consideravelmente a eficiência computacional.

Na etapa de implementação prática, observou-se ainda que mesmo após a etapa de pré-processamento, as informações da área e do centroide extraídas, apresentaram ruídos oriundos da captura da imagem. Para os testes de rastreamento planar, observou-se que a magnitude do ruído não degradava a ação do controle, contudo a magnitude dos ruídos eram maiores na informação da área extraída. Desta maneira, para o rastreamento em profundidade seria necessário a utilização de um filtro para a minimização dos ruídos indesejados.

Ao longo dos experimentos, observou-se que o ruído em questão apresentava algumas frequências bem definidas, desta maneira, pode-se implementar um filtro passa baixas contínuo para a solução do problema. Primeiramente, há que se definir a frequência de corte do filtro, deste modo estimou-se a frequência principal que o ruído apresentava, determinando assim, que esta seria a frequência de corte. Deve ser lembrado que este valor é uma aproximação, pois todo o sistema de controle servo visual foi efetuado para processar em uma frequência de 25 Hz, deste modo o ruído também será um sinal discreto amostrado nesta mesma frequência.

Observação: Vale ressaltar que todo desenvolvimento do controle adaptativo servo visual foi construído no domínio do tempo, contudo a sua aplicação ocorre em um sistema parcialmente discreto, haja visto que as imagens são capturadas em períodos de tempo específicos e consequentemente os sinais de controle são produzidos nestes mesmos intervalos de tempo. Desta maneira, quanto maior a frequência de amostragem do sistema, maior será a sua proximidade com um sistema contínuo. Caso a aplicação não permita tal aproximação, seria necessário efetuar a discretização do filtro passa baixas mencionado, assim como do controle servo visual através de algum dos métodos conhecidos, como o método de Euler por exemplo.

Denotando f_r como a frequência de corte do filtro passa baixas contínuo e w_r como a mesma frequência em radianos por segundo, define-se a a função de transferência do filtro contínuo da seguinte maneira:

$$w_r = 2\pi f_r$$

$$H_f(s) = \frac{w_r}{s + w_r} \tag{3.10}$$

Com base na equação (3.10), denotando A_{vf} como a área filtrada e $\dot{A_{vf}}$ a sua derivada, a dinâmica da área da imagem filtrada será:

$$\dot{A_{vf}} = -w_r A_{vf} + w_r A_v \tag{3.11}$$

3.1.2 Componentes do Sistema de Controle Servo Visual

Após o entendimento de alguns conceitos relacionados à Visão Computacional, é necessário definir os componentes básicos que irão compor o sistema servo visual real.

Coforme já mencionado, uma classe específica de sistemas servo visuais apresentam três componentes básicos: Uma câmera colorida como sensor, um manipulador industrial como sistema robótico e um computador para o processamento de imagem, comunicação com o sistema robótico e implementação da estratégia de controle.

Seguindo a configuração acima, neste trabalho será utilizado como sensor uma Câmera Web Digital HD Logitech C270 ©, o sistema robótico Motoman DIA10 © e um computador usuário com o software Matlab ©.

Câmera Web Digital HD Logitech C270

As câmeras web são câmeras digitais simples que podem ser modeladas com câmeras pinhole e que efetuam a captura e digitalização de imagens, transmitindo os dados para computadores através de conexão usb. Desta maneira, através das lentes a câmera efetua a captura da imagem, a digitaliza e representa em uma matriz de pixels que carregam consigo a informação de cor de um ponto específico da imagem. A resolução de (640 x 480) pixels é a mais comum para câmeras web, podendo apresentar resoluções maiores ou menores. Já a representação das cores geralmente é efetuada no sistema RGB.

Em virtude das características apresentadas, foi utilizada na implementação prática a câmera Web Digital HD Logitech C270. O modelo em questão foi escolhido em virtude da elevada qualidade da imagem capturada, superior às outras câmeras de mesma categoria, além de fornecer imagens em resolução HD (1280 x 720) pixels. Ressalta-se ainda que, a taxa de transmissão dos dados utilizada na implementação prática foi de 25 quadros por segundo, pois o

intervalo de tempo definido entre os quadros, será suficiente para efetuar todo o processamento computacional do controle servo visual.

Sistema Robótico Motoman DIA10

Atualmente os manipuladores robóticos são largamente utilizados nos diversos segmentos da indústria ou até mesmo em outros campos. Isso se devendo à facilidade e flexibilidade com que desenvolvem suas tarefas, superando a habilidade humana em termos de força e precisão em muitos casos. Desta maneira, para implementação prática do Controle Servo Servo Visual proposto por [10], será utilizado o sistema robótico Motoman DIA10.

O Motoman DIA10 , como pode ser observado na figura 3.2, é um sistema composto por dois manipuladores robóticos com 7 juntas de revolução cada manipulador e 1 junta de revolução central na estação, totalizando 15 juntas e consequentemente 15 DOF's. A disposição das juntas foi desenvolvida de forma a permitir que o seu movimento se assemelhe ao movimento do braço humano, desta maneira o Motoman DIA10 é apto a realizar as mais variadas tarefas como embalagem, montagem e transferência de objetos, dentre outras. Ações estas que seriam possíveis apenas pela intervenção humana. A sua estrutura de atuação (atuadores, encoders, redução e freios), foi desenvolvida e combinada em um pacote pequeno, leve e poderoso, o que favorece a locomoção do robô e amplia sua capacidade para movimentação de cargas.

Figura 3.2: Ilustração do sistema robótico Motoman DIA10

Para o problema proposto neste trabalho, serão utilizadas as juntas L e U do braço robótico esquerdo e a junta S1 da estação em uma configuração antropomórfica como proposto em [10]. Desta maneira, as juntas restantes permanecerão estáticas. As juntas de interesse, as medidas relevantes e a posição do alvo, utilizadas na implementação prática, estão representadas na figura 3.3.

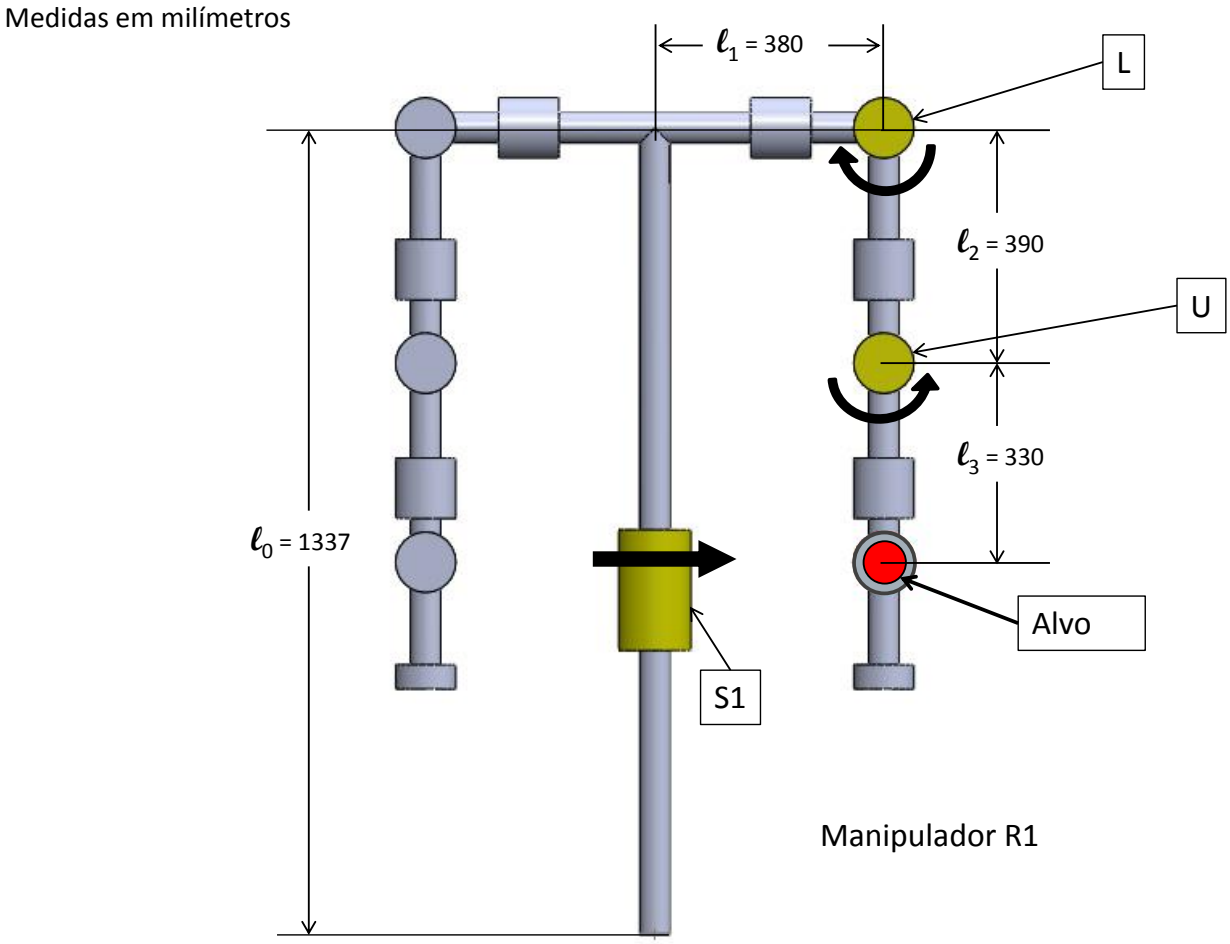

Figura 3.3: Diagrama representativo com as juntas utilizadas na implementação prática

Além dos manipuladores existem ainda três componentes do sistema robótico Motoman de extrema importância e que foram utilizados na implementação prática, sendo eles:

- O controlador NX100
- O controle manual *Programming Pendant*
- High-speed Synchronous Controller (HSC)

A seguir, haverá uma descrição resumida da funcionalidade de cada um desses componentes para que a sua utilização na implementação prática seja compreendida, contudo no apêndice A existem informações mais detalhadas sobre cada um deles, incluindo um resumido manual

de operação da plataforma Motoman DIA10 e a sua utilização no Sistema de Controle Servo Visual.

O controlador NX100 é responsável por executar a malha de controle servo que garante a movimentação das juntas do robô Motoman. Desta maneira, os comandos de posicionamento das juntas são enviados ao NX100 que processa e reenvia os comandos em forma de sinal de controle até o Motoman. Ressalta-se ainda que, é através do NX100 que o sistema robótico é alimentado.

O controle manual *Programming Pendant* é o principal instrumento de interação entre o operador e o robô Motoman. Desta maneira, a partir dele pode-se alterar as configurações, visualizar dados e definir tarefas e comandos de posicionamento junta a junta, dentre outros. As principais funcionalidades do *Programming Pendant* utilizadas na implementação prática foram a movimentação manual das juntas para a posição inicial utilizada no teste experimental e a definição da configuração no modo Playback que permite a execução de tarefas programadas.

O High-speed Synchronous Controller (HSC) consiste em um sistema composto por hardware e software que permite a operação do Motoman através de computadores usuários. Assim, o HSC efetua uma interface de comunicação em alta velocidade entre o computador usuário e o NX100, permitindo o envio e recebimento de comandos entre ambas as partes. Esta funcionalidade foi extremamente importante, visto que todo o controle servo visual foi desenvolvido em um computador usuário.

Computador e Software Matlab ©

As aplicações robóticas em sua maioria necessitam de uma estrutura responsável pelo processamento e recebimento de dados dos sensores, cálculo dos sinais de controle para a atuação e envio dos comandos ao robô, seja ele um manipulador ou qualquer outro sistema robótico. Para tal são utilizados microcontroladores, microcomputadores ou computadores, dentre outros. Em virtude da elevada capacidade de processamento de dados, da sua flexibilidade em termos de conexões para comunicação e da relativa simplicidade com que se constroem aplicações, um computador comum popularmente conhecido como *Desktop*, foi utilizado na implementação prática do Controle Servo Visual.

O computador e o software Matlab farão parte da "Inteligência"do sistema servo visual

implementado. Com o sistema robótico responsável pela movimentação e execução da tarefa, e a câmera responsável pela visualização do ambiente através da captura da imagem, há ainda a necessidade de se integrar ambas funções, para em conjunto com a estratégia de controle definir por completo o Sistema de Controle Servo Visual. Assim, esta função será exercida pelo computador, utilizando-se o software Matlab.

O Matlab é um ambiente com linguagem em alto nível e interativo para computação numérica, visualização e programação. É uma poderosa ferramenta que permite o desenvolvimento de algoritmos, análise de dados, além de criar modelos e aplicações com interface gráfica. Existe ainda um ambiente para simulação conhecido como Simulink que permite simular e testar sistemas dinâmicos, modelos e algoritmos em alto nível.

Uma característica de grande importância apresentada no Matlab é a presença do pacote aplicativo para processamento de imagens e computação visual. Com estas duas ferramentas, ações como captura de imagem a partir de câmeras web, pré-processamento e filtragem, detecção de cor e segmentação, puderam ser executados em conjunto com os cálculos geométricos da imagem necessários ao Controle Servo Visual proposto.

O software Matlab foi utilizado inicialmente para a simulação do modelo Servo Visual, em conjunto com a estratégia de controle proposta e em seguida foi utilizado para a implementação prática do algoritmo de controle e processamento de imagem.

Para a implementação prática, foi programado no Matlab rotinas e funções para executar a captura, processamento de imagem e extração das informações da área e centroide do alvo. Em seguida, as rotinas para a execução da estratégia de controle servo visual, utilizando as informações da imagem foi desenvolvida para a posterior integração com os componentes restantes do sistema servo visual.

Por fim, foi utilizado um aplicativo essencial para a implementação prática do estudo proposto, o MotomanSimulinkHost. Este programa é responsável por efetuar uma interface de comunicação via software entre o Matlab e o HSC. Desta maneira, o aplicativo em questão permitiu que algoritmo de controle servo visual desenvolvido na plataforma Matlab pudesse receber os dados de posição corrente das juntas do robô através do HSC e ao mesmo tempo enviar as posições desejadas, permitindo desta maneira, a execução completa da malha de controle

servo visual.

Ressalta-se que o apêndice A apresenta uma descrição detalhada do funcionamento do aplicativo MotomanSimulinkHost e sua estrutura interna.

Integração dos componentes servo visuais

Após uma definição básica dos componentes que compõem o sistema servo visual, pode-se agora, descrever como ocorre a integração destes componentes para a implementação prática do estudo proposto.

A câmera web, conectada ao computador usuário, fornece as imagens do alvo fixado na extremidade do efetuador. As rotinas desenvolvidas no Matlab que implementam o controle servo visual, processam as imagens e extraem as informações de área e centroide utilizadas na malha de controle adaptativa servo visual. A malha de controle, por sua vez, envia os comandos das posições desejadas das juntas ao MotomanSimulinkHost que efetuará a interface de comunicação via software entre o HSC e a aplicação no Matlab. O HSC receberá os comandos do computador usuário e enviará em alta velocidade ao NX100 para que a movimentação das juntas do robô Motoman se processe. A nova posição do manipulador decorrente movimentação das juntas será detectada novamente através da imagem, fechando assim a malha completa de controle servo visual, incluindo o computador usuário, a câmera (visão) e o sistema robótico. No apêndice A mais detalhes sobre a integração dos sistemas são apresentados.

3.2 Interface Gráfica de Controle

Para executar as etapas de simulação e testes experimentais de maneira mais simplificada e automatizada, foi criada uma interface gráfica utilizando o componente GUI (*Graphical User Interface*) presente no software Matlab.

O GUI consiste em uma ferramenta que permite ao usuário executar tarefas interativamente, através de controles como botões, caixas de edição (*edit boxes*) etc. A integração do GUI com o Matlab permite incorporar, dentro da interface, tarefas como a customização e criação de plotagens, além da criação e execução de funções e arquivos com extensão (.m).

A criação de uma GUI fica a cargo do componente conhecido como GUIDE (*GUI deve-*

lopment environment). O GUIDE é a principal ferramenta para desenvolver e programar uma GUI. Através do GUIDE Layout Editor, pode-se criar e configurar os elementos da interface como os botões, janelas, eixos de plotagem e automaticamente gerar o código em Matlab que define as propriedades dos elementos e as GUI *Callbacks* , que são rotinas executadas todas as vezes que o usuário interage com o elemento da interface.

Deste modo, para a implementação prática, uma interface gráfica GUI foi criada para executar tanto as simulações quanto os testes práticos. Nesta interface foram incorporados os algoritmos de controle servo visuais desenvolvidos, as funções e tarefas para o captura e processamento de imagens, além da exibição dos gráficos e rotinas para salvar e carregar os dados dos parâmetros de um experimento ou simulação.

Deve ser ressaltado que, anteriormente ao desenvolvimento da interface gráfica, todo o código com as rotinas e funções tanto da simulação quanto do teste prático já haviam sido construídos e testados. Para a simulação, um arquivo em extensão (.m) foi desenvolvido, incorporando o modelo servo visual, a estratégia de controle servo visual e as rotinas para a geração dos gráficos referentes aos resultados.

Para a execução do teste prático, foi efetuado o mesmo procedimento. Um arquivo (.m) foi desenvolvido de forma a incorporar em primeira instância as etapas de captura, pré-processamento, extração das características e visualização da imagem capturada. Em seguida, incorporou-se os comandos para conexão com o MotomanSimulinkHost através interface de serviço, além do recebimento das posições das juntas do Motoman e envio das posições desejadas. Por fim, foi adicionado ao arquivo, o código com a estratégia de controle servo visual, as rotinas para salvar e carregar os dados dos parâmetros e as rotinas para exibição dos gráficos referentes aos resultados do teste prático.

Após o desenvolvimento dos códigos, com as etapas de simulação e teste prático, foi dado início ao desenvolvimento da interface gráfica com o objetivo de automatizar as execuções das duas tarefas. Assim, a interface gráfica resultante pode ser observada na figura 3.4.

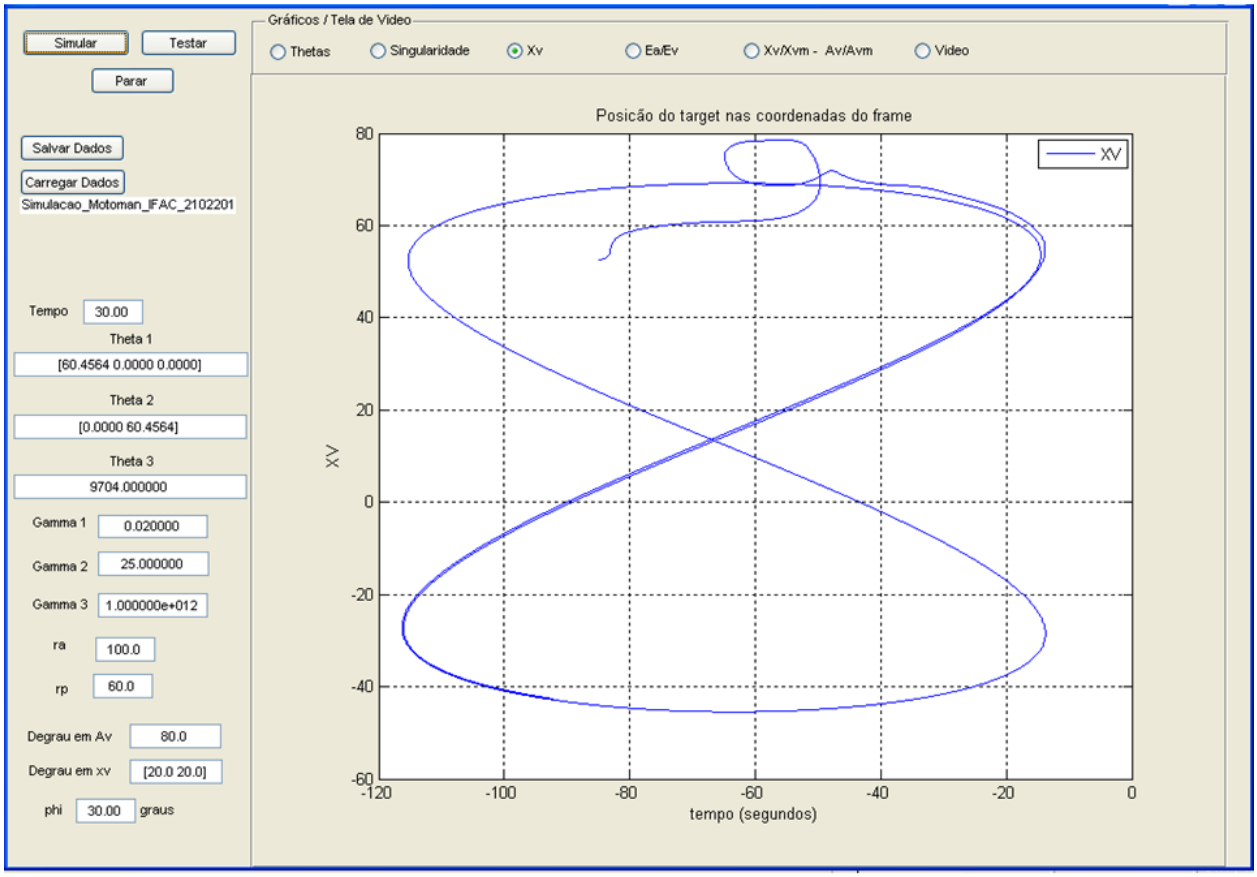

Figura 3.4: Interface Gráfica (GUI) desenvolvida para a simulação e o teste experimental do controle servo visual

Assim, todo o processo de escolha dos parâmetros, na simulação e teste experimental, foi efetuado utilizando a interface gráfica desenvolvida, o que automatizou e otimizou a execução dessas tarefas. Caso o leitor tenha interesse, na próxima subseção há uma descrição básica de operação da interface gráfica desenvolvida, contudo a leitura desta subseção não é necessária para a compreensão da implementação prática. Desta maneira, o leitor pode prosseguir diretamente para a seção 3.3 onde os resultados da implementação prática são demonstrados.

3.2.1 Operação da Interface Gráfica

Os botões Simular e Testar no canto superior esquerdo da tela são utilizados para chamar os arquivos (.m) que executam a simulação e o teste prático respectivamente. Finalizados a simulação ou o teste prático, em ambos os casos foi implementado uma rotina para a exibição dos

gráficos resultantes, diretamente na janela principal da interface. Desta maneira, basta selecionar uma dentre as opções "Thetas", "Singularidade", "Xv", "Ea/Ev"ou "Xv/Xvm - Av/Avm"no canto superior direito da tela principal que serão exibidos os gráficos referentes a cada opção.

Quando se está executando um teste prático, é possível visualizar a imagem capturada da câmera, clicando na opção "Vídeo" no canto superior direito da tela principal. A opção "Vídeo" também é utilizada para verificar se a trajetória de referência definida na imagem se encontra dentro do espaço visível da câmera. Para executar tal verificação, foi desenvolvido uma rotina que efetua a plotagem da trajetória desejada na tela de visualização da imagem capturada, assim o usuário pode observar simultaneamente, trajetória e imagem. Se algum trecho da trajetória não se apresentar visível na imagem observada da interface, significa que o caminho criado está fora do espaço visual da câmera e o usuário deve criar uma nova trajetória, para poder executar o teste.

Por fim, para modificar os parâmetros utilizados tanto na simulação quanto no teste experimental, basta o usuário inserir os dados nos respectivos espaços presentes no canto esquerdo da tela. O campo com a etiqueta "Tempo"é utilizada para alterar os tempos de simulação ou teste prático, os campos "Theta1", "Theta2"e "Theta3"são utilizados para alterar as condições iniciais dos parâmetros de adaptação. Já as opções "Gamma1", "Gamma2"e "Gamma3" alteram os ganhos de adaptação. Para trajetórias de referência senoidais na imagem, os parâmetros "ra"e "rp" alteram as amplitudes das senoides, sendo o "ra" referente ao rastreamento em profundidade e "rp"ao rastreamento planar. As opções "Degrau em Xv"e "Degrau em Av", por sua vez, alteram a posição inicial das trajetórias de referência em relação aos valores obtidos na primeira vez em que se extrai da imagem, a área e o centroide do alvo. Por fim, o parâmetro "phi" altera o ângulo da câmera em relação ao sistema de coordenadas da base do manipulador quando se está efetuando uma simulação.

Caso o usuário queira parar a execução da simulação ou teste experimental e observar os resultados até aquele instante, basta pressionar o botão Parar.

Existe ainda, a possibilidade do usuário carregar e salvar dados de uma simulação ou teste prático, podendo assim posteriormente repetir um experimento com base nos dados ou simplesmente visualizar novamente os resultados a partir dos gráficos. Desta maneira, para salvar ou

carregar os dados, basta o usuário pressionar os botões Carregar ou Salvar respectivamente e escolher as pastas de destino ou fonte. Ao salvar, deve-se escolher o nome do arquivo. Ressalta-se ainda que, somente arquivos de extensão (.mat) são suportados para as duas ações mencionadas.

3.3 Resultados

Tendo em mãos todas as ferramentas e componentes descritos nas seções anteriores, foi possível então, efetuar o teste experimental de maneira que tanto as etapas de desenvolvimento da teoria, quanto as etapas simulação do sistema servo visual pudessem ser validadas. Assim, a estratégia de controle servo visual proposta foi implementada em uma aplicação real e a análise dos resultados e gráficos foi efetuada, comprovando o rastreamento satisfatório das trajetórias planares.

O teste experimental demonstrado a seguir, apresenta apenas o rastreamento de uma trajetória planar senoidal com apenas uma componente de frequência.

Os parâmetros utilizados foram:

Sinal de referência planar: $x_v^* = r_p\,[\,\sin(w_n\,t) \quad \cos(w_n\,t)\,]^T + [\,-204 \quad -62\,]^T\,pixels$

Tempo de simulação: $60s$

Período de amostragem do sistema completo: $0.04s$

Parâmetros utilizados:

$q(0) = [-\pi \quad -\frac{\pi}{4} \quad \frac{\pi}{2}]$

$w_n = \frac{\pi}{10} \quad rad/s$

$x_{v0} = [-204 \quad -2]$

$x_{vm0} = [-204 \quad -2]$

$\lambda_m = 1$

$\lambda_M = 1$

$\lambda = 10$

$rp = 60$

$\Theta_1(0) = [0.2063 \quad -0.0162 \quad 0.0084]$

$\Theta_2(0) = [0.0 \quad 0.2]$

$\gamma_1 = 0.8$

$\gamma_2 = 0.8$

$\phi \approx 0$

A partir do gráfico da trajetória de rastreamento planar representado na figura 3.5, observa-se a proximidade entre a trajetória de referência definida na imagem e a trajetória real, o que vem a confirmar o rastreamento. Desta maneira, ao analisar o gráfico da figura 3.6, observamos que o erro de rastreamento planar oscila inicialmente, contudo após aproximadamente 25 segundos de teste, ambos apresentam uma variação em módulo em torno de $2\,pixels$, o que para fins práticos caracteriza uma convergência satisfatória do erro de rastreamento para zero. Portanto, a estratégia de controle servo visual aplicada pode ser validada não só pelos resultados teóricos como também pelo resultado experimental apresentado.

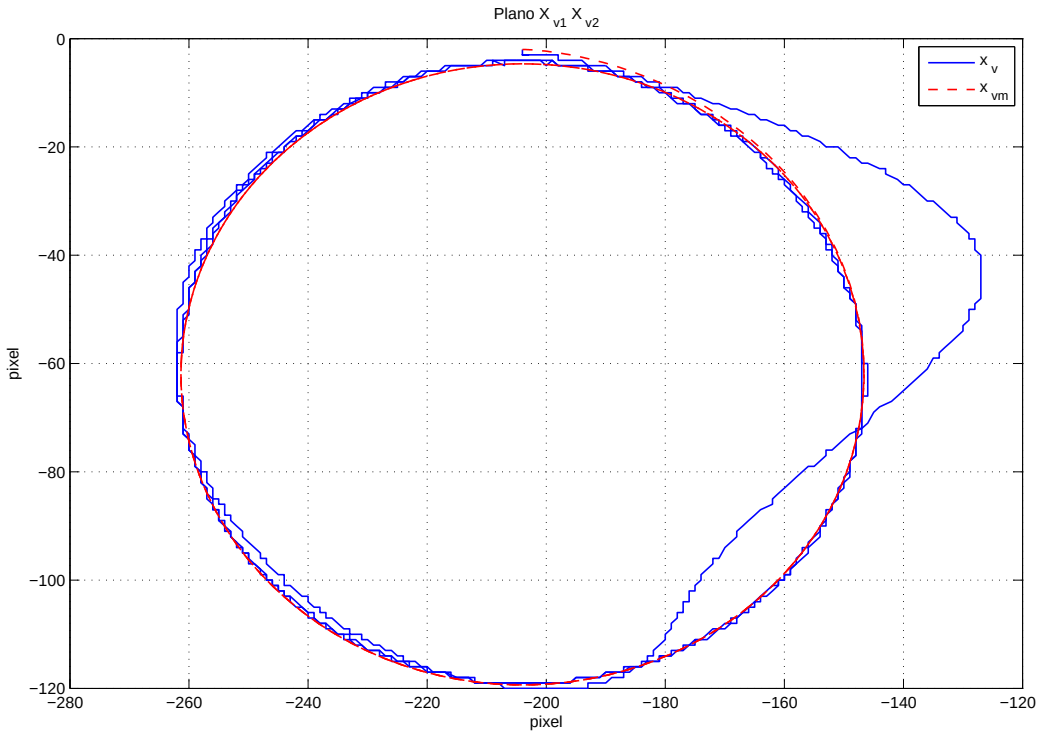

Figura 3.5: Gráfico das trajetórias real e de referência para o rastreamento planar, obtido em teste experimental com ganhos de adaptação $\gamma_1 = 0.8$, $\gamma_2 = 0.8$

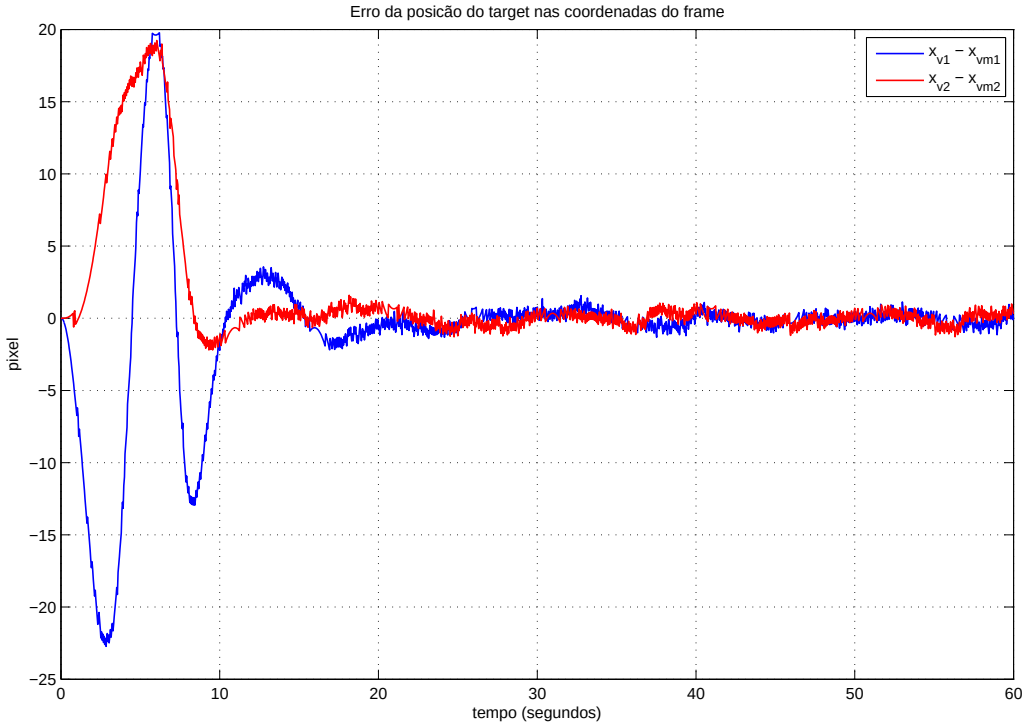

Figura 3.6: Gráfico do erro de rastreamento planar, obtido em teste experimental com ganhos de adaptação $\gamma_1 = 0.8$, $\gamma_2 = 0.8$

Deve-ser ressaltado ainda que, os parâmetros de adaptação (figura 3.7) também oscilam inicialmente e logo após convergem para valores específicos, o que vem a confirmar a eficácia da lei de adaptação implementada e a consequente limitação dos parâmetros de adaptação.

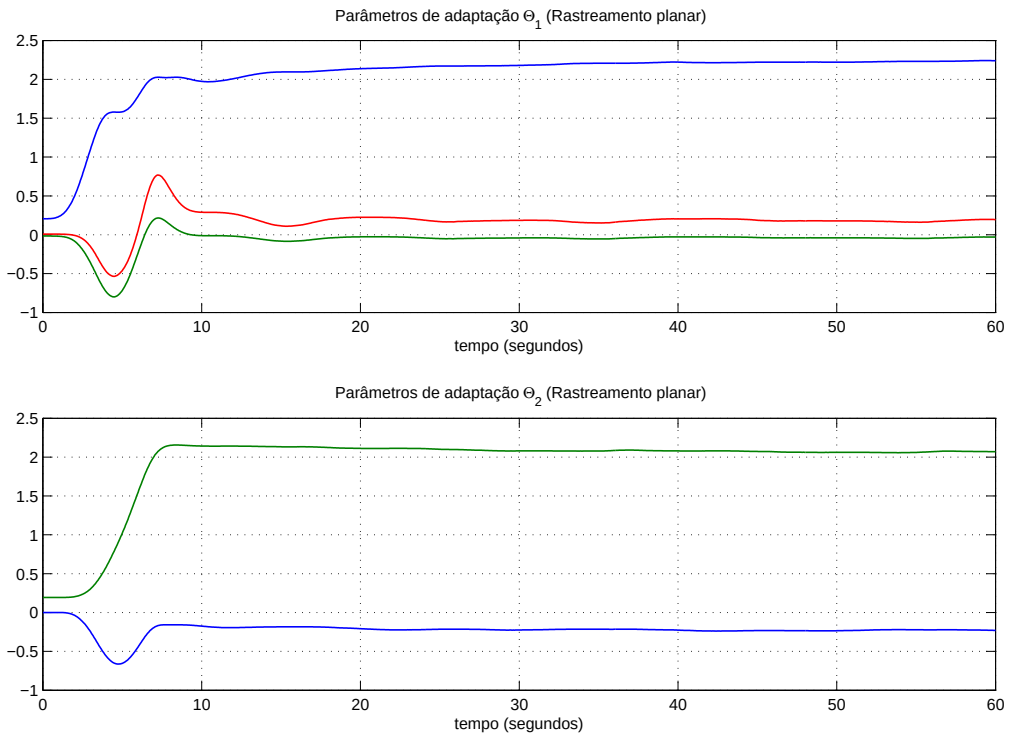

Figura 3.7: Gráfico da estimação dos parâmetros Θ_1 e Θ_2 desconhecidos, obtido em teste experimental com ganhos de adaptação $\gamma_1 = 0.8$, $\gamma_2 = 0.8$

Ressalta-se que , em virtude de algumas restrições no cronograma, não foi possível efetuar testes experimentais para o rastreamento de trajetórias tridimensionais (planar e em profundidade). Assim, o teste completo do estudo proposto será designado para trabalhos futuros.

Capítulo 4

Conclusões

No desenvolvimento deste trabalho, foi possível compreender e analisar a estratégia de controle servo visual proposta por [10]. As etapas de detalhamento da teoria aplicada, execução de simulações e todas as ações desenvolvidas no processo de implementação prática, foram cruciais para alcançar os objetivos almejados inicialmente.

Para lidar com as incertezas paramétricas da câmera, a impossibilidade de medição de profundidade e velocidades da imagem, foi utilizada uma abordagem de controle adaptativo servo visual que se mostrou eficiente para a resolução do problema proposto. O movimento do robô foi particionado em um movimento 2D e 1D, caracterizados pela movimentação planar e em profundidade do manipulador. Em seguida as leis de controle e adaptação foram desenvolvidas e o problema de definição dos sinais da matriz K_p foi solucionado via parametrização SDU. Após, a estabilidade assintótica dos erros de rastreamento do sistema servo visual foi comprovada através do conceito de funções SPR e da teoria de Lyapunov.

Assim, os mesmos resultados obtidos em [10] foram encontrados e percebeu-se que algumas definições como a utilização das propriedades de sistemas SPR e a parametrização do sinal de controle para o rastreamento planar com base na fatoração SDU, foram extremamente importantes para se alcançar a convergência dos erros de rastreamento e dos parâmetros de adaptação, utilizando a teoria de Lyapunov.

Concluído o estudo dos conceitos, foi iniciado a etapa de simulação. Ao longo das simulações observou-se que a sintonia dos ganhos de adaptação é um trabalho complexo, visto que há que se respeitar o compromisso entre o aumento do ganho de adaptação para a melhoria da performance dos erros de rastreamento e por outro lado evitar que este aumento nos valores dos

ganhos produzam elevadas oscilações tanto na estimação dos parâmetros, quanto nos próprios erros de rastreamento.

Deve-se ressaltar que, o parâmetro λ presente nos erros filtrados interferem em alto nível na eficácia do controle servo visual. O aumento do parâmetro λ reduz os erros de rastreamento e o respectivos tempos de convergência, melhorando assim a performance da estratégia de controle. Além disso, observou-se que ao definir condições iniciais dos parâmetros de adaptação muito discrepantes em relação aos respectivos valores de convergência, é notório o aumento das oscilações nos erros de rastreamento, mesmo elevando os ganhos de adaptação, o que novamente dificulta a sintonia.

Desta maneira, a partir dos gráficos resultantes das simulações, demonstrados na seção 2.4, observou-se que o controle servo visual implementado proporcionou erros de rastreamento planar e em profundidade com valores aceitáveis em torno de zero, o que vem a comprovar a eficácia da teoria de controle servo visual desenvolvida.

A implementação prática além da sua importância para a validação da teoria servo visual desenvolvida, foi relevante também pelo conjunto de resultados indiretos alcançados. O primeiro deles foi o desenvolvimento da interface gráfica de controle GUI através do componente GUIDE do Matlab, o que facilitou e otimizou tanto a execução dos testes experimentais quanto as simulações. Outro resultado importante, foi a descrição detalhada da etapa de implementação prática que abordou questões como a integração dos sistemas, além do desenvolvimento de um manual simplificado da plataforma Motoman DIA10 e a sua operação através do HSC. Este fato permitirá que outras pessoas usufruam das informações contidas neste simplificado manual para o desenvolvimento de novos trabalhos.

Finalmente, os resultados diretos obtidos na implementação prática, validaram a estratégia de controle servo visual proposta por [10]. A partir dos gráficos, observou-se que para o rastreamento planar em um ambiente real o controle servo visual proporcionou erros menores que 1 pixel em módulo, o que para fins práticos comprova a convergência assintótica do erro de rastreamento planar para zero, consolidando assim toda a análise efetuada desde o início deste trabalho. Além disso, os parâmetros de adaptação apresentaram transitórios com pouca oscilação e rapidamente convergiram para valores constantes, denotando a eficácia da lei de

adaptação utilizada.

4.1 Trabalhos Futuros

Com base no estudo efetuado em [10], outra possibilidade seria efetuar a implementação prática, utilizando incertezas tanto na dinâmica quanto na cinemática do manipulador. Além disso, poderia ser considerado no modelo servo visual rotações da câmera em torno dos três eixos coordenados, efetuando assim, uma análise das implicações que ocorreriam sob a estratégia de controle adaptativo servo visual.

Tendo em vista a rápida evolução dos robôs humanoides, outra continuidade deste trabalho poderia considerar a utilização de uma garra acoplada à extremidade de um dos manipuladores do Motoman DIA10 para a execução de tarefas de rastreamento de trajetórias e captura de objetos utilizando a servo visão.

Bibliografia

[1] CHANG, W. C., AND SHAO, C. K. Hybrid eye-to-hand and eye-in-hand visual servoing for autonomous robotic manipulation. *SICE Annual Conference* (2010), 415–422.

[2] CHAUMETTE, F., AND HUTCHINSON, S. Visual servo control part I: Basic approaches. *IEEE Robotics and Automation Magazine* (2006), 82–90.

[3] CHAUMETTE, F., AND HUTCHINSON, S. Visual servo control part II: Advanced approaches. *IEEE Robotics and Automation Magazine* (2007), 109–118.

[4] CHEN, Q., ZHU, S., WANG, X., AND WU, W. Analysis on an uncalibrated image-based visual servoing for 6 DOF industrial welding robots. *IEEE International Conference on Mechatronics and Automation* (2012), 2013–2018.

[5] CHIEN, M. C., AND HUANG, A. C. FAT-based adaptive visual servoing for robots with time varying uncertainties. *IEEE International Conference on Robotics and Automation* (2009), 3700–3705.

[6] COSTA, R. R., HSU, L., IMAI, A. K., AND KOKOTOVIĆ, P. Lyapunov-based adaptive control of MIMO systems. *Automatica* (2003), 1251–1257.

[7] FRANKLIN, G. F., POWELL, J. D., AND EMAMI-NAEINI, A. *Feedback Control of Dynamic Systems*, 6a ed. Pearson, 2009.

[8] HERREJON, R., KAGAMI, S., AND HASHIMOTO, K. Position based visual servoing for catching a 3-D flying object using RLS trajectory estimation from a monocular image sequence. *International Conference on Robotics and Biomimetics* (2009), 665–670.

[9] IOANNOU, P. A., AND SUN, J. *Robust Adaptive Control*. Prentice Hall, 1996.

[10] LEITE, A. C., ZACHI, A. R. L., LIZARRALDE, F., AND HSU, L. Adaptive 3D visual servoing without image velocity measurement for uncertain manipulators. *18th IFAC World Congress* (2011), 14584–14589.

[11] LIU, Y., AND WANG, H. An adaptive controller for image-based visual servoing of robot manipulators. *World Congress on Intelligent Control and Automation* (2010), 988–993.

[12] MA, Y., KOSECKÁ, J., SOATTO, S., AND SASTRY, S. *An Invitation to 3-D Vision from Images to Models*. Springer, 2004.

[13] RUSS, J. C. *The Image Processing handbook*, 2ª ed. IEEE Press, 1995.

[14] SICILIANO, B., SCIAVICCO, L., VILLANI, L., AND ORIOLO, G. *Robotics: Modelling, Planning and Control*, 2ª ed. Springer, 2009.

[15] SLOTINE, J.-J. E., AND LI, W. *Applied Nonlinear Control*. Prentice Hall, 1991.

[16] WANG, H., LIU, Y.-H., CHEN, W., AND WANG, Z. A new approach to dynamic eye-in-hand visual tracking using nonlinear observers. *IEEE/ASME Transactions on Mechatronics* (2011), 387–394.

[17] ZACHI, A. R. L., HSU, L., LIZARRALDE, F., AND LEITE, A. C. Adaptive control of nonlinear visual servoing systems for 3d cartesian traking. *Revista Controle & Automação* (2006), 381–390.

Apêndice A

Manual de Operação da Plataforma Motoman DIA10

Este manual irá descrever todos os componentes presentes na plataforma Motoman DIA10, incluindo sua operação no contexto da aplicação servo visual desenvolvida. Desta maneira, o leitor poderá compreender com mais detalhe o funcionamento destes componentes que permitiram o a execução do experimento baseado no estudo proposto.

A.1 A Plataforma Motoman DIA10

Como já mencionado, a plataforma Motoman DIA10 consiste em uma base com dois manipuladores com sete juntas de revolução cada um e uma junta de revolução na estação base, um gabinete controlador denominado NX100, um controle manual denominado *Programming Pendant* e o *High-speed Synchronous Controller PC* (HSC). Nas próximas subseções haverá uma descrição de cada um desses componentes, contendo informações relevantes que permitam compreender a operação do sistema robótico inserida em uma aplicação servo visual.

A.1.1 Os Manipuladores

Para simplificar a designação de cada manipulador e da estação do Motoman DIA10, será denominado R1 o manipulador esquerdo, R2 o manipulador direito, e S1 a estação. Cada junta apresenta um nome para seu eixo de rotação, fato que será útil nas próximas etapas, pois a partir dessa nomenclatura identificamos qual junta do robô esta sendo controlada. Na figura A.1 é demonstrado o sistema robótico Motoman DIA10 real e em seguida a figura A.2 consiste

em uma representação esquemática da plataforma completa com as juntas representadas pela cor verde, os respectivos eixos de rotação e os nomes de cada um.

Figura A.1: Sistema robótico Motoman DIA10

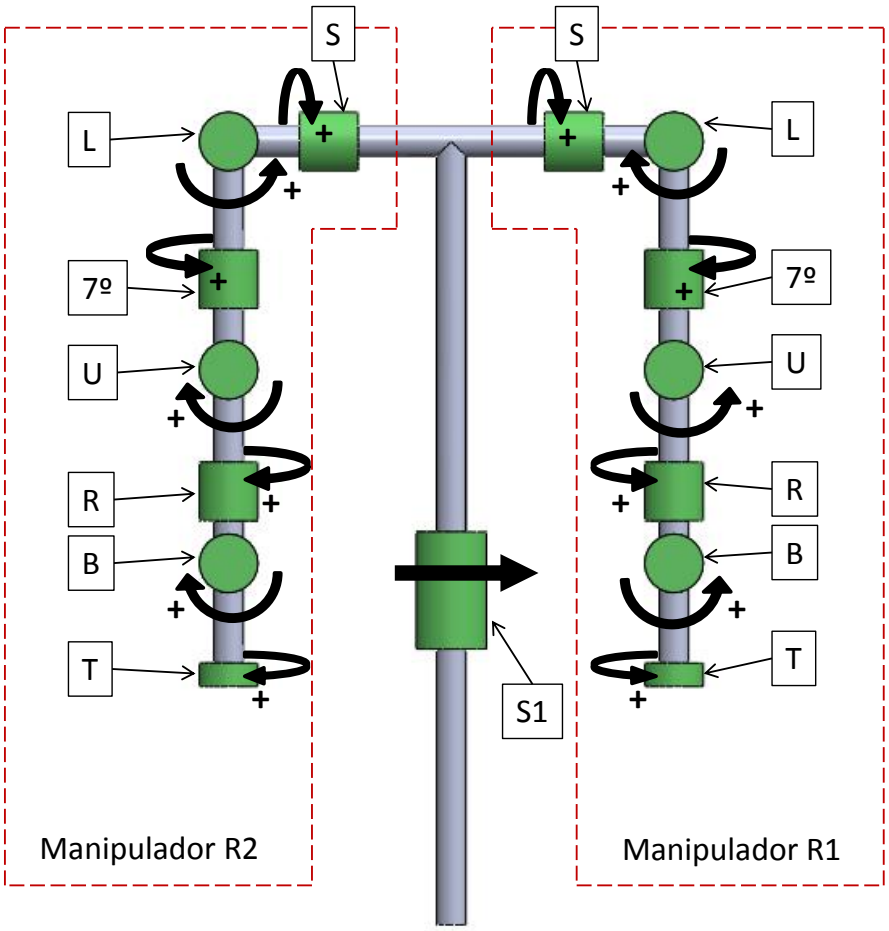

Figura A.2: Diagrama representativo das juntas, nomes e respectivos eixos de rotação do Motoman DIA10

Para a implementação prática utilizou-se apenas um dos manipuladores, o manipulador esquerdo (R1) e a estação (S1). Ressalta-se ainda que, em virtude da estratégia de controle utilizada estar baseada em um manipulador antropomórfico com 3 DOF's ou seja 3 juntas de revolução, foram selecionadas as juntas L e U do manipulador R1 e a junta S1 da estação para reproduzir o experimento de acordo com o estudo proposto (figura A.3).

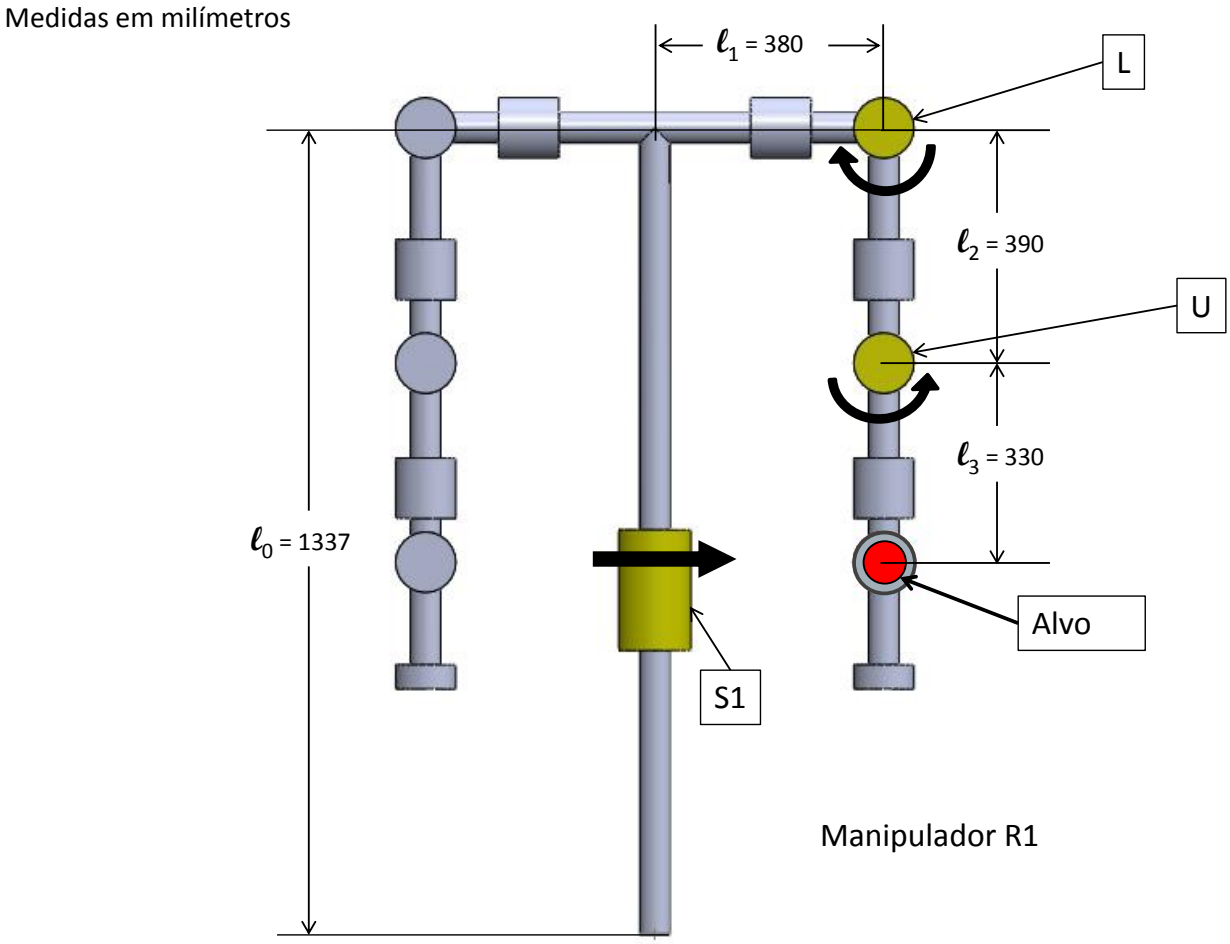

Figura A.3: Diagrama representativo com as juntas e medidas de interesse

Para o desenvolvimento da cinemática diferencial é necessário o conhecimento das medidas do manipulador em questão, desta maneira as medidas de interesse também estão representadas na figura A.3. Assim,

$$l_0 = 1337mm$$

$$l_1 = 380mm$$

$$l_2 = 390mm$$

$$l_3 = 330mm$$

A.1.2 Controlador NX100

O controlador NX100 consiste basicamente em um controlador responsável por receber os comandos do controle manual *Programming Pendant* ou do HSC, processá-los e enviá-los aos manipuladores e estação em forma de sinais elétricos amplificados, para que os motores das juntas sejam movimentados. Desta maneira, além de fornecer energia ao sistema robótico, também efetua a interface entre a operação e o robô.

A figura A.4 exemplifica as conexões básicas efetuadas para uma operação através do *Programming Pendant*:

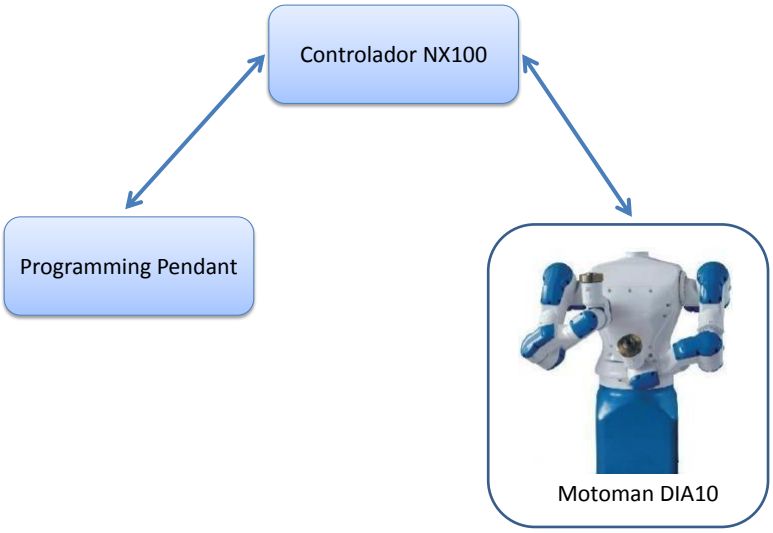

Figura A.4: Diagrama básico das conexões em uma operação através do *Programming Pendant*

A.1.3 Programming Pendant

O *Programming Pendant* é o principal meio de operação do Motoman DIA10. Este controle manual permite ao usuário efetuar configurações, enviar comandos de operação, visualizar dados dos manipuladores e estação, visualizar possíveis erros, programar tarefas autônomas, dentre outros. Ou seja, sua função é permitir que o operador tenha total domínio sob o robô e suas configurações.

O *Pendant* é equipado com diversos botões e chaves que permitem tanto a operação do robô para execução de tarefas quanto para configurações que sejam relevantes.

Abaixo são descritos os botões mais importantes utilizados na implementação prática deste trabalho:

Botão E.STOP: Desliga o Servo Power, ou seja, desliga a energia dos motores acionando todos o freios. Após desligar, os leds do SERVO ON também se apagam.

ENABLE SWITCH: Para religar o Servo Power, ou seja religar os motores e liberar os freios, este botão deve ser acionado até a metade de sua excursão total, contudo a operação funcionará apenas se as luzes do Servo ON estiverem acesas. Se o Enable Switch for liberado completamente ou pressionado até o final, os motores são freados novamente. Esta opção é utilizada principalmente no modo "Teach".

CURSOR: O Cursor é usado para mover o foco de seleção entre as opções.

Botão SELECT: Usado para selecionar as opções no menu.

Botão SHIFT: Dentre outras funções é utilizado para mover a junta 7º dos manipuladores. Assim, basta segurar o botão [SHIFT] e pressionar simultaneamente [S+] ou [S-] para mover a junta.

Botão COORD: Seleciona o sistema de coordenadas de operação, quando se está efetuando uma operação manual.

Botões FAST e SLOW: Para operações manuais, cada vez que os botões são pressionados, eleva ou reduz a velocidade de locomoção das juntas dos manipuladores ou estação.

Botão ROBOT: Na operação manual, seleciona o conjunto que será operado: o manipulador direito (R1), manipulador esquerdo (R2) ou a estação (S1).

Axis Keys: Para mover o robô em modo manual, algum botão da Axis Keys deve estar pressionado. Assim , é responsável por selecionar o eixo do movimento.

Botão CANCEL: Cancela dados de entrada e reseta possíveis erros.

Botão DELETE: Deleta instruções registradas ou dados inseridos.

Botão ENTER: Registra instruções, dados, posições correntes do manipulador etc.

Numeric Keys : Inserção de caracteres na forma de números.

Botão START: Inicia o movimento do robô no modo Playback de operação. Ao pressioná-lo, acende-se uma luz verde em seu interior que se apaga em caso de erro ou o botão E.STOP for pressionado.

Mode Switch: Seleciona os Modos Playback, Teach ou Remote. Na posição PLAY habilita tarefas em Playback, na posição TEACH habilita operação nos eixos e configurações pelo *Programming Pendant* e na posição REMOTE habilita a operação por sinais remotos. Cada modo de operação será melhor detalhado na próxima subseção.

Botão SERVO ON READY: Habilita o Servo Power ser ligado, isto é, permite que os freios dos motores sejam liberados para movimentação caso o botão ENABLE SWITCH seja ativado.

Os botões e a sua posição no *Programming Pendant* podem ser observados nas figuras A.5 e A.6.

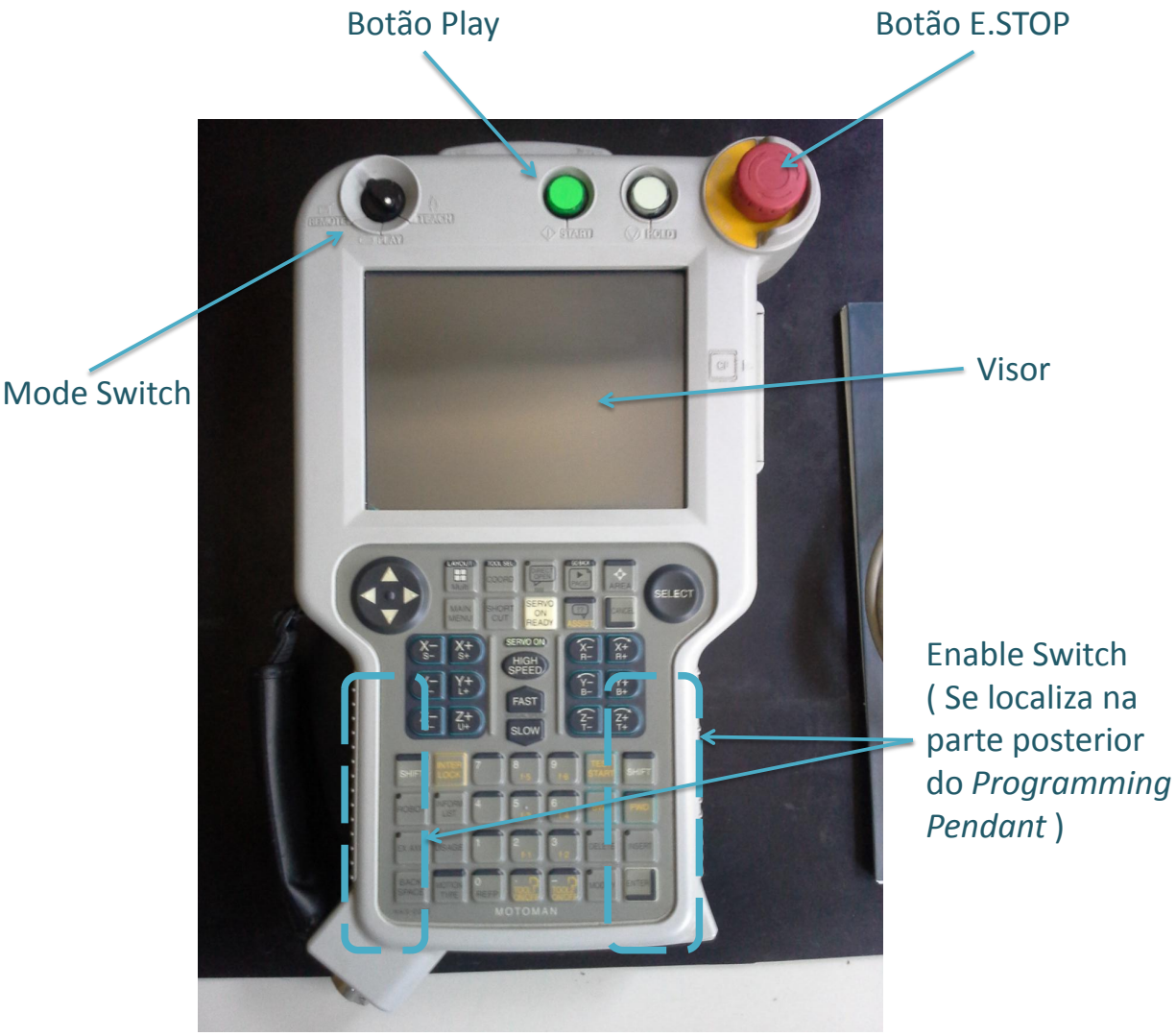

Figura A.5: Controlador Manual *Programming Pendant* - Visão Geral

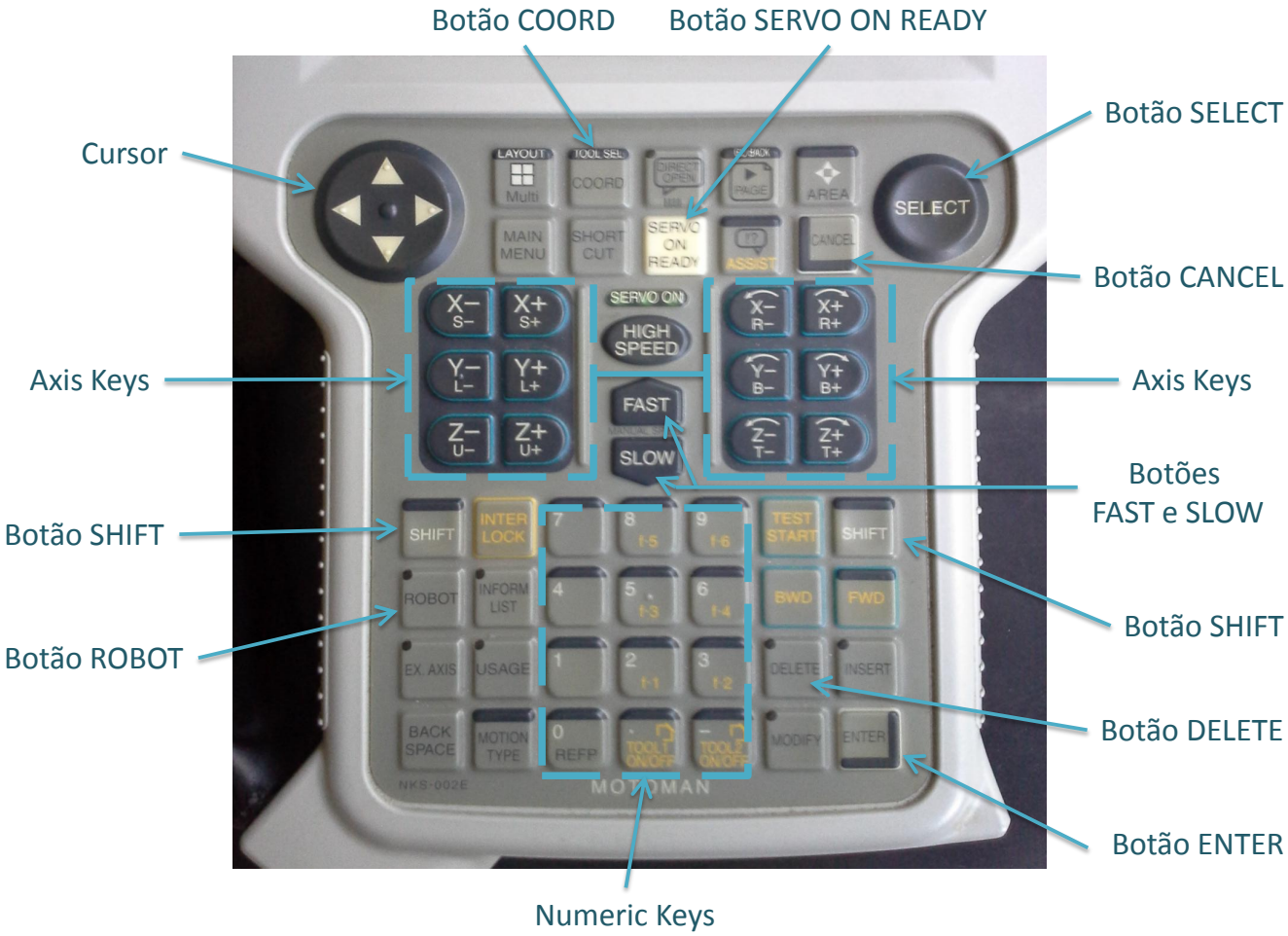

Figura A.6: Controlador Manual *Programming Pendant* - Imagem ampliada dos botões relevantes

A.1.4 Modos de Operação

Existem 3 modos de operação do Motoman DIA10: O Modo Teach, o modo Playback e o modo Remote.

O modo Teach permite que o operador efetue a movimentação das juntas através do *Pendant* , prepare e execute uma tarefa que para o sistema é denominada *job* ou configure arquivos e parâmetros. Assim, na implementação prática deste trabalho, utilizou-se o modo Teach principalmente para mover o manipulador manualmente até uma posição de interesse, para em seguida iniciar a operação servo visual. Ressalta-se ainda que, a operação manual do Motoman pode ser efetuada com base em 5 sistemas de coordenadas:

- Coordenadas das Juntas
- Coordenadas Cartesianas
- Coordenadas Cilíndricas
- Coordenadas da Ferramenta (*Tool Coordinates*)
- Coordenadas do Usuário (*User Coordinates*)

O sistema de coordenadas mais importante na implementação prática foi o sistema de Coordenadas das Juntas que permite a movimentação de cada junta independentemente, assim basta selecionar a junta de acordo com o nome de seu respectivo eixo de rotação e movê-la no sentido positivo ou negativo, de acordo com o sinal definido na figura A.2. Como os outros sistemas de coordenadas não foram utilizados a sua descrição será omitida.

Na subseção (A.1.8), será descrito, passo a passo, uma operação simples no modo Teach para mover o manipulador até uma posição desejada.

O modo Playback permite que o operador crie uma sequência de movimentos para as juntas e as execute como uma tarefa (*job*) ensinada, além de permitir a criação, modificação e exclusão de arquivos de condições. Contudo, a função mais importante do modo Playback, utilizada na implementação prática, consiste em permitir que o robô seja controlado por um computador usuário utilizando o High-speed Synchronous Controller (HSC) como interface entre o computador e o NX100. Desta maneira, para habilitar o controle através do computador é necessário selecionar a *Job Computer* no controle *Programming Pendant* e efetuar a conexão com o HSC. A descrição do funcionamento do HSC, da integração dos sistemas para permitir o controle através do computador e o passo a passo de uma operação através de um computador usuário utilizando o modo Playback, serão efetuados nas próximas seções.

Por fim, existe o modo Remote que permite efetuar operações como *Servo ON Ready*, *Star*, *Cycle Change*, *Call Master Job*, comandadas por sinais de entrada remotos. Contudo, este modo de operação não apresentou utilidade durante o experimento.

A.1.5 High-speed Synchronous Controller (HSC)

O High-speed Synchronous Controller consiste basicamente em um controlador de alta velocidade que permite operar o Motoman DIA10, enviando comandos e recebendo dados a partir de computadores usuários. O diagrama representativo das conexões, envolvendo a operação através do HSC são mostradas na figura A.7. Em termos físicos, o HSC apresenta basicamente um processador com o sistema operacional Linux, executando o software HSC em tempo real, além placa de rede StarFabric e uma saída CRT para a conexão com um monitor.

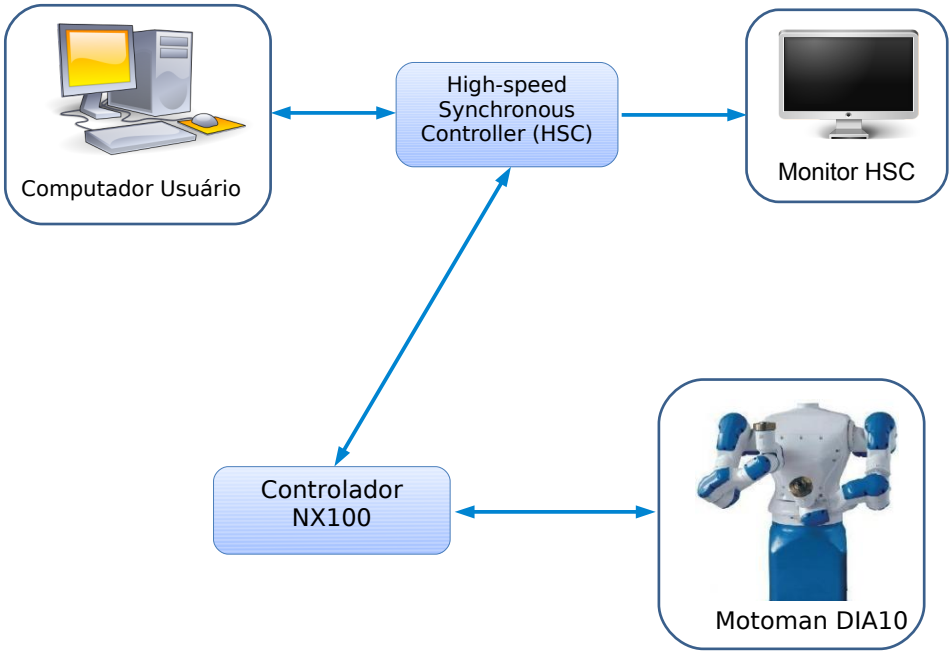

Figura A.7: Diagrama básico das conexões em uma operação com HSC

O Software HSC é encarregado de configurar uma memória compartilhada entre o computador usuário e o Controlador NX100, permitindo assim que informações sejam armazenadas para efetuar o controle síncrono do Motoman DIA10. Desta maneira, o usuário do computador troca as informações com os manipuladores através do HSC. A função principal do Computador é processar dados externos como respostas de sensores ou imagens e gerar as posições desejadas dos manipuladores que devem ser enviadas ao Motoman, consequentemente a tarefa do HSC consistirá em repassar os valores de posições desejadas ao NX100, para que a movimentação do robô se processe. A seguir, a figura A.8 apresenta um diagrama de como se

processa internamente, o controle do Motoman através do HSC.

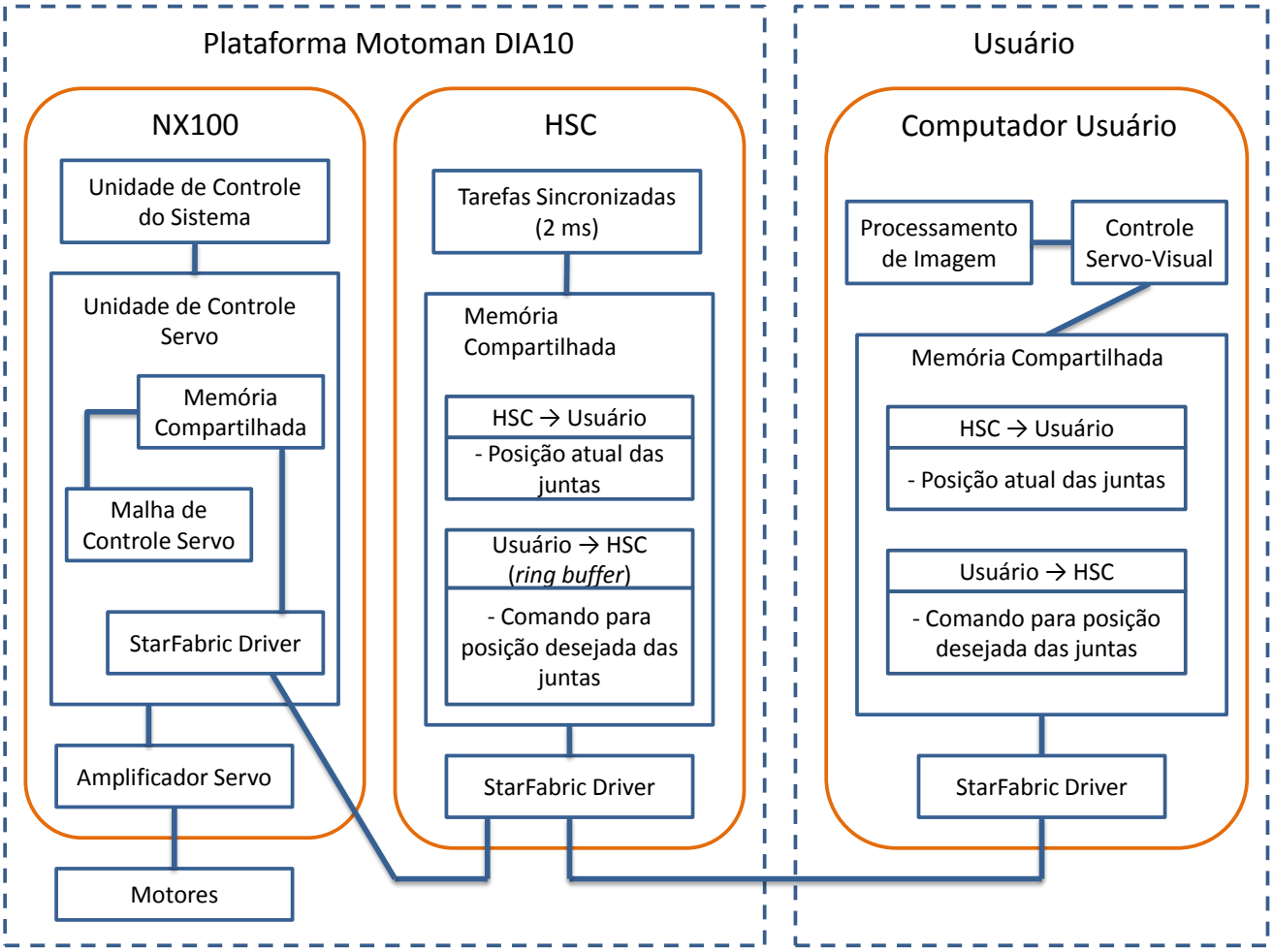

Figura A.8: Diagrama do processo interno de uma operação com HSC

Como o HSC trabalha em tempo real para o envio dos comandos ao NX100 e o sistema operacional dos computadores usuários, usualmente Windows©, não processam em tempo real, a área de memória compartilhada no HSC que recebe os comandos de posição é formada por um *Ring Buffer* ou *Buffer* em anel, para garantir que os dados não sejam perdidos pelo envio assíncrono a partir do computador.

Além do software executado no HSC PC ainda existem dois softwares que devem estar presentes no computador usuário, o HSC TesteMonitor e o HSC TesteMotion. O TesteMonitor é uma aplicação que irá mostrar no monitor os dados das juntas dos manipuladores e estação a cada 0.5 segundos, desta maneira pode-se confirmar se a comunicação entre computador e HSC esta se processando da maneira correta. Já o TesteMotion é uma aplicação de teste para

executar simples *jobs* através do HSC. Contudo, este último não foi utilizado na implementação prática.

Assim, ao definir o Modo Playback no *Programming Pendant* e selecionar a *job computer* é possível controlar o Motoman através de um computador usuário utilizando como interface entre computador e Controlador NX100 o High Speed Syncronous Controller.

A.1.6 MotomanSimulinkHost

Após a conexão entre o computador usuário e o controlador NX100 através do HSC, torna-se então possível a troca de informações entre o computador e Motoman DIA10. Contudo, como a comunicação com o HSC se processa através da rede starfabric é necessário criar uma aplicação para o computador usuário que integre o protocolo de comunicação e permita, assim, o recebimento dos dados e o envio dos comandos para o controle do robô. Para tal tarefa, foi utilizado o aplicativo MotomanSimulinkHost.

O MotomanSimulinkHost proporciona três funcionalidades básicas: Uma interface gráfica, uma malha de controle para posições das juntas e um serviço de comunicação via Robot Raconteur. Cada funcionalidade será descrita a seguir.

A interface gráfica apresenta botões que executam as tarefas de conexão e controle. Os botões start e shutdown são responsáveis por iniciar e desligar respectivamente a comunicação com o HSC. O botão load model tem por objetivo carregar o modelo que implementa a malha de controle para posicionamento das juntas e os botões start model e stop model são responsáveis por iniciar e parar a execução das malhar de controle.

Para a implementação prática foi utilizado um modelo pré-definido, conhecido como MotomanSlave. Entretanto é possível criar outros modelos da malha de controle para posição através do componente Simulink do software Matlab. Este componente permite, dentre outras tarefas, criar malhas de controle e exportar os modelos criados, encapsulando-os em arquivos no formato dll (dynamic link library), que podem ser executados em aplicações C#. Desta maneira, a malha de realimentação utilizada terá a função unicamente de receber a posição desejada do usuário e garantir que as posições das juntas do robô estejam de acordo com o valor desejado. Para aumentar a eficiência, o aplicativo MotomanSimulinkHost foi desenvolvido para executar

a malha de realimentação em um thread de alta prioridade que irá processar os dados recebidos e enviar os dados de controle para o HSC.

Para descrever a última funcionalidade do aplicativo, a criação de um serviço de comunicação com o Matlab, é necessário fazer um esclarecimento básico sobre o que é o Robot Raconteur. O Robot Raconteur é uma ferramenta que permite a criação de interfaces para o controle de componentes através de linguagens de alto nível. Desta maneira, o RobotRaconteur cria "Serviços"de comunicação TCP/IP entre uma linguagem de alto nível e uma aplicação em C++ ou C# e assim, pode-se efetuar o controle de sensores ou atuadores que estejam conectados a computadores, diretamente do Matlab sem que ele apresente nenhum conhecimento a priori do componente que está sendo controlado.

Portanto, o MotomanSimulinkHost utiliza o RobotRaconteur para criar uma interligação via software com o Matlab. Consequentemente, quando o aplicativo estiver executando a malha de controle e se comunicando com o HSC, será possível receber os dados das juntas e enviar posições desejadas, diretamente do Matlab que estará executando a malha de controle servo visual.

A.1.7 Integração dos sistemas

Após o entendimento de todos os componentes utilizados na implementação prática e as suas respectivas funcionalidades, agora será descrita a integração dos componentes que permitiu a execução do experimento.

Na execução prática do Controle Servo Visual, a "inteligência"é efetuada no computador usuário pelo algoritmo de controle servo visual desenvolvido no software Matlab, desta maneira a imagem é capturada, processada e com base nas informações extraídas, a malha de controle servo visual fornece as posições desejadas. Os comandos de posição são enviados ao aplicativo MotomanSimulinkHost através da interface de serviço RobotRaconteur. Paralelamente, o aplicativo estará executando o modelo da malha de controle para posição das juntas e ao receber os comandos do Matlab inicia o controle, enviando os comandos de correção das posições ao HSC através da memória compartilhada composta pelo *buffer* em anel. O HSC, operando em tempo real, enviará os comandos recebidos ao NX100 a cada 2 milissegundos, novamente

através de memória compartilhada. O NX100 configurado pelo *Programming Pendant* , estará operando no modo playbak com a *job computer* selecionada, podendo assim receber os dados do HSC e efetuar a correção das juntas para a posição desejada, conforme foi definido na malha de controle servo visual em execução no computador usuário.

A.1.8 Operação em modo *Teach* - Passo a passo

Nesta seção serão descritos os passos básicos para efetuar uma operação manual em modo *Teach*. Os comandos executados simultaneamente apresentarão o sinal + entre eles. As ações seguem abaixo:

- Girar a chave Power do NX100 para a posição Powe On e aguardar a inicialização do *Programming Pendant* .

- Girar o Mode Switch no canto superior esquerdo do *Programming Pendant* para a posição *Teach*.

- Na tela do menu principal do *Pendant* , selecionar Robot e após Current Position, assim poderá ser visualizado diretamente da tela do *Programming Pendant* as variações de posição de cada junta, ao movê-la.

- Para selecionar qual dos manipuladores ou a estação será movida, basta pressionar o botão ROBOT. A cada vez que é pressionado, seleciona uma das opções R1, R2 ou S1. As alterações podem ser vistas na parte superior direita da tela do *Pendant* . Assim, escolheremos a opção R1 que se refere ao manipulador esquerdo.

- Verificar se o botão E.STOP está pressionado, observando se existe um círculo vermelho no canto superior direito do visor do *Pendant* . Se estiver aceso, girar para direita o botão E.STOP para liberar a operação do robô.

- Pressionar o botão SERVO ON READY. Uma vez pressionado, os led's esverdeados do Servo On acenderão intermitentemente.

- Verificar, no canto superior direito do visor do *Programming Pendant* , se o sistema de coordenadas está selecionado como o sistema de coordenadas das juntas. Caso outro sistema de coordenadas esteja selecionado, pressionar o botão COORD até chegar à opção coordenadas das juntas.

Neste momento, o MotomanDIA10 está apto a ser operado junta a junta, manualmente. Para mover alguma junta, deve-se efetuar duas ações simultâneas:

- Primeiro pressionar o ENABLE SWITCH, na parte posterior do *Programming Pendant* , até a metade da excursão que ele é capaz de efetuar. Simultaneamente, deve-se pressionar um dos botões do *Axis Key*. Desta maneira, ao efetuar, por exemplo, as ações: Pressionar [ENABLE SWITCH] + [BOTÃO L+], estaremos movendo a junta L do manipulador R1 no sentido positivo.

- Para aumentar ou reduzir a velocidade de movimento das juntas basta pressionar no *Pendant* os botões FAST ou SLOW respectivamente. O nível de velocidade pode ser visualizado no canto superior direito da tela presente no *Pendant* .

- Caso haja algum movimento inesperado ou corra o risco de algum dos manipuladores se chocar com outros objetos, liberar ou pressionar até o final da excursão o ENABLE SWITCH para frear todos os motores ou pressionar o botão E.STOP para bloquear a operação.

Assim, ao finalizar a operação manual o Motoman é desligado, girando-se a chave Power do NX100 para a posição Power Off.

A.1.9 Operação por Computador Usuário - Passo a passo

Nesta seção serão descritos os passos básicos para efetuar uma operação em modo Playbak utilizando um computador usuário, assim:

- Girar a chave Power para a posição Powe On e aguardar a inicialização do *Programming Pendant* .

- Girar o Mode Switch no canto superior esquerdo do *Programming Pendant* para a posição *PLAY*.

- Pressionar o botão Play Mode Enable no gabinete NX100.

- Selecionar a opção *Job* no menu principal do *Programming Pendant* com o cursor e o botão SELECT. Nesse momento, algumas opções de *job* aparecerão no visor. Então, selecione a opção *Computer*.

- Pressionar o botão power do HSC. Aguardar o HSC estabelecer conexão com o NX100 e com o computador usuário. Existem duas opções para verificar se a conexão foi estabelecida, a

primeira consiste em aguardar aparecer no monitor do HSC a tela demonstrada na figura A.9, já a segunda consiste em ir até o computador usuário e executar o aplicativo TesteMonitor que abrirá um console com informações de conexão entre HSC e NX100. Quando a conexão for estabelecida os dados das juntas dos manipuladores irão aparecer a cada 0.5 segundos no console, então deve-se parar a execução do aplicativo, pressionando a tecla Q do teclado.

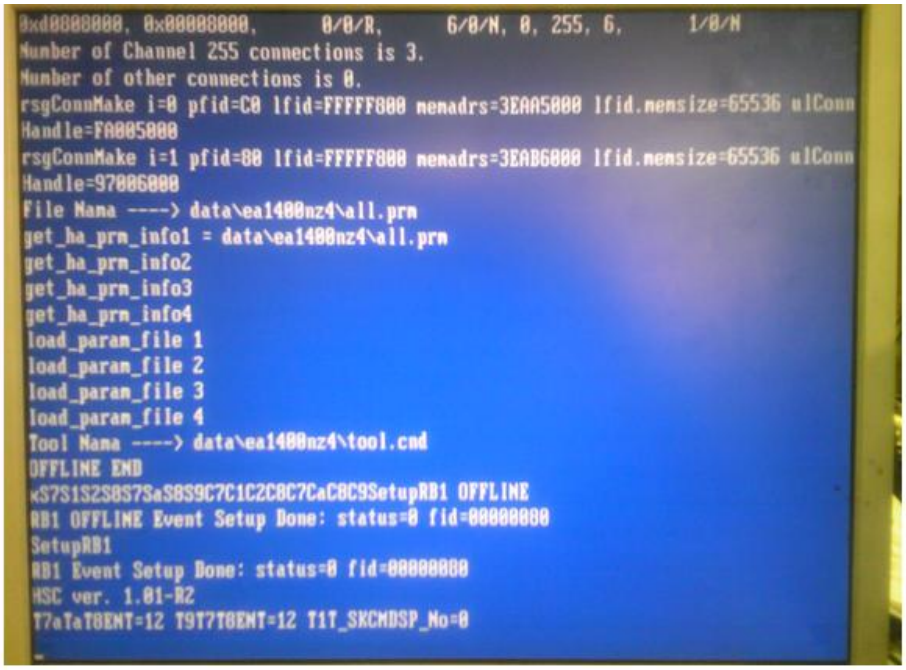

Figura A.9: Imagem mostrada no monitor do HSC, indicando a comunicação bem sucedida com o computador usuário

- Estabelecidas as conexões com o HSC, deve-se executar o programa MotomanSimulinkHost.

- Clicar no botão Start da interface gráfica do MotomanSimulinkHost para iniciar a conexão entre o Computador usuário e o HSC. No momento em que a conexão for estabelecida, os valores das posições das juntas aparecerão na janela principal da interface.

- Clicar no botão load model e carregar o modelo MotomanSlave.

- Clicar no botão start model para executar a malha de controle de posição do modelo MotomanSlave.

- Executar o software Matlab e inserir 2 vezes o seguinte comando em seu prompt:

motoman = RobotRaconteur.Connect('tcp://27.0.0.1:5203/MotomanSimulinkHost/MotomanDIA10');

É necessário executar duas vezes, pois a primeira execução gera um erro, já a segunda é bem sucedida.

Após conectar o Matlab ao aplicativo, para obter as posições das juntas ou enviar o comando de uma posição desejada, deve-se executar respectivamente os seguintes comandos no prompt do Matlab:

var1 = motoman.ActualRobotJointAngles;

motoman.DesiredRobotJointAngles = var2;

O primeiro comando irá inserir na variável var1, um vetor de 15 posições com os valores das posições das juntas em radianos por segundo. Assim, a primeira posição será referente à junta S da estação S1, as 7 posições seguintes serão referentes às juntas da manipulador R2, começando pela junta S até a junta T, seguindo a ordem em que aparecem no manipulador. Deste modo, as 7 últimas posições do vetor serão referentes às juntas do manipulador R1 com o mesmo ordenamento que acontece em R2.

Para executar o segundo comando, deve-se criar uma variável que neste exemplo foi denominada var2 e que também será um vetor com 15 posições, contudo a variável var2 deverá conter os valores das posições desejadas das juntas, também em radianos por segundo, no mesmo ordenamento que ocorreu para a variável var1.

Antes de efetuar a próxima ação o usuário deve executar os dois comandos anteriores por segurança, afim de evitar que os manipuladores ou a estação efetuem movimentos inesperados, ao ativar o botão Start do *Pendant* .

- Para permitir que os comandos de posicionamento enviados do Matlab sejam executados no MotomanDIA, a última ação consiste em pressionar o botão START no *Programming Pendant* . O robô executará os comandos enviados do computador usuário, somente após o botão START ser habilitado.

Deve-se ressaltar novamente que, em caso do robô efetuar algum movimento inesperado e/ou houver risco de choque em alguma de suas partes constituintes, o operador deve pressionar imediatamente o botão E.STOP para bloquear a operação e frear os motores.

Portanto, após executar todos os passos mencionados o usuário poderá implementar seu algoritmo de controle na ferramenta Matlab, para então conduzir seu experimento com Motoman DIA10, enviando e recebendo os dados do robô.

Ao final do experimento, para desligar todo o sistema, deve-se executar as seguintes ações:
- Pressionar o botão stop model no aplicativo MotomanSimulinkHost.
- Logo após, pressionar o botão Shutdown.
- Pressionar o botão Power do HSC PC para desligá-lo. Possivelmente algum alarme de erro será acionado no *Programming Pendant* , contudo este fato é considerado normal.
- Girar a chave Power do NX100 para a posição Power Off.

Apêndice B

Prova da positividade real estrita (SPR) para a função de transferência do erro aumentado

Para comprovar se uma dada função de transferência é SPR, é necessário definir alguns conceitos e teoremas.

Funções estritamente estáveis

A partir de [15], obtêm-se que um sistema linear e invariante no tempo do tipo

$$\begin{aligned} \dot{x} &= Ax + bu \\ y &= cx \end{aligned} \tag{B1}$$

é estritamente estável quando todos os autovalores da matriz A estiverem estritamente no semi-plano esquerdo do plano complexo. A partir de [7] temos que os autovalores da matriz A são justamente os pólos da função de transferência do sistema representado pela equação (B1).

Agora, será descrito o teorema para a definição de funções SPR.

Seja $h(s)$ uma função de transferência racional própria de n-ésima ordem de um sistema linear SISO.

Teorema B.1 *A função de transferência $h(s)$ é estritamente real positiva (SPR) se e somente se*

i) $h(s)$ é uma função de transferência estritamente estável

ii) a parte real da função $h(s)$ é estritamente positiva ao longo do eixo jw, i.e.,

$$\forall w \geq 0 \quad \mathbb{R}\text{e}[h(s)] > 0$$

Nosso objetivo é provar que a função de transferência do erro de rastreamento em profundidade aumentado (equação 2.43) é SPR. Retomando a função, temos:

$$H(s) = \frac{\beta^{-1}(s + \lambda)}{s^2 + \lambda s + \lambda^2} \quad , \quad \lambda > 0$$

Rearranjando denominador da função de transferência em questão, podemos representá-la da seguinte maneira:

$$H(s) = \frac{\beta^{-1}(s + \lambda)}{(s + \frac{\lambda - \lambda\sqrt{3}j}{2})(s + \frac{\lambda + \lambda\sqrt{3}j}{2})} \tag{B2}$$

Da equação (B2), obtêm-se que os pólos s_1 e s_2 da função de transferência são:

$$s_1 = \frac{-\lambda + \lambda\sqrt{3}j}{2} \quad , \quad s_2 = \frac{-\lambda - \lambda\sqrt{3}j}{2} \tag{B3}$$

Como pode ser observado, os pólos apresentam parte real estritamente negativa, visto que $\lambda > 0$. Assim, conclui-se que os pólos se localizam no semiplano esquerdo do plano complexo e consequentemente a função de transferência $H(s)$ é estritamente estável. Desta maneira atendemos a condição (i) do teorema B.1.

Para avaliar a segunda condição do teorema, os seguintes cálculos são efetuados:

Substituindo s por jw na equação (B2) teremos:

$$H(jw) = \frac{\beta^{-1}(jw + \lambda)}{(jw + \frac{\lambda - \lambda\sqrt{3}j}{2})(jw + \frac{\lambda + \lambda\sqrt{3}j}{2})}$$

$$= \frac{\beta^{-1}(jw + \lambda)}{(\lambda^2 - w^2) + jw\lambda}$$

multiplicando o numerador e o denominador pelo termo $(\lambda^2 - w^2) - jw\lambda$ teremos

$$H(jw) = \frac{\beta^{-1}(jw + \lambda)[(\lambda^2 - w^2) - jw\lambda]}{[(\lambda^2 - w^2) + jw\lambda][(\lambda^2 - w^2) - jw\lambda]}$$

$$= \frac{\beta^{-1}[jw\lambda^2 - jw^3 + w^2\lambda + \lambda^3 - w^2\lambda - jw\lambda^2]}{(\lambda^2 - w^2)^2 + w^2\lambda^2}$$

$$= \frac{\beta^{-1}[\lambda^3 - jw^3]}{(\lambda^2 - w^2)^2 + w^2\lambda^2}$$

$$H(jw) = \underbrace{\frac{\beta^{-1}\lambda^3}{(\lambda^2 - w^2)^2 + w^2\lambda^2}}_{\mathbb{R}\mathrm{e}(H(jw))} - j\frac{\beta^{-1}w^3}{(\lambda^2 - w^2)^2 + w^2\lambda^2} \tag{B4}$$

Como pode ser observado, a primeira parcela da equação (B4) representa a parte real da função $H(jw)$. Assim, com $\lambda, \beta^{-1} > 0$, concluímos que $\forall\, w \geq 0$ a parte real da função $H(jw)$ será sempre maior que zero. Desta maneira, a exigência (ii) do teorema B.1 é satisfeita.

Portanto, a função de transferência $H(s)$ do sistema que representa o erro aumentado de rastreamento em profundidade é comprovadamente SPR.