DAVI YOSHINOBU KIKUCHI

SISTEMA DE CONTROLE SERVO VISUAL DE UMA CÂMERA PAN-TILT COM RASTREAMENTO DE UMA REGIÃO DE REFERÊNCIA

São Paulo
2007

DAVI YOSHINOBU KIKUCHI

SISTEMA DE CONTROLE SERVO VISUAL DE UMA CÂMERA PAN-TILT COM RASTREAMENTO DE UMA REGIÃO DE REFERÊNCIA

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Engenharia.

São Paulo
2007

DAVI YOSHINOBU KIKUCHI

SISTEMA DE CONTROLE SERVO VISUAL DE UMA CÂMERA PAN-TILT COM RASTREAMENTO DE UMA REGIÃO DE REFERÊNCIA

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Engenharia.

Área de Concentração:
Engenharia de Controle e Automação Mecânica

Orientador:
Prof. Dr. Lucas Antonio Moscato

São Paulo
2007

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 18 de maio de 2007.

Assinatura do autor _____

Assinatura do orientador_____

FICHA CATALOGRÁFICA

Kikuchi, Davi Yoshinobu
Sistema de controle servo visual de uma câmera *pan-tilt* com rastreamento de uma região de referência / D.Y. Kikuchi. -- ed. rev. -- São Paulo, 2007.
106 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1.Visão computacional 2.Controle ótimo 3.Câmeras de video I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos II.t.

À minha família e aos meus amigos, que, não importa o que aconteça, estarão sempre comigo e jamais serão esquecidos.

AGRADECIMENTOS

Ao professor Lucas Antonio Moscato por ser meu orientador desde o terceiro ano de graduação, começando com a Iniciação Científica, passando pelo Trabalho de Formatura e agora pelo curso de Mestrado.

Ao professor Adinan de Souza, que, juntamente com o professor Lucas, apresentou o projeto que deu origem a esta dissertação e incentivou a sua realização.

Ao professor Walter de Britto Vidal Filho, que forneceu o equipamento que permitiu a implementação física dos conceitos desenvolvidos neste trabalho.

Ao colega de laboratório e técnico Rodrigo Guerato Siqueira, pelo total suporte eletrônico fornecido sempre que necessário.

À CTEEP (Companhia de Transmissão de Energia Elétrica Paulista) pelo incentivo, inspiração e suporte durante o período inicial deste projeto.

À CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) pelo apoio financeiro proporcionado.

A todos aqueles que, direta ou indiretamente, contribuíram para a realização deste trabalho.

RESUMO

Uma câmera pan-tilt é capaz de se movimentar em torno de dois eixos de rotação (pan e tilt), permitindo que sua lente possa ser apontada para um ponto qualquer no espaço. Uma aplicação possível dessa câmera é mantê-la apontada para um determinado alvo em movimento, através de posicionamentos angulares pan e tilt adequados. Este trabalho apresenta uma técnica de controle servo visual, em que, inicialmente, as imagens capturadas pela câmera são utilizadas para determinar a posição do alvo. Em seguida, calculam-se as rotações necessárias para manter a projeção do alvo no centro da imagem, em um sistema em tempo real e malha fechada. A técnica de rastreamento visual desenvolvida se baseia em comparação de uma região de referência, utilizando a soma dos quadrados das diferenças (SSD) como critério de correspondência. Sobre essa técnica, é adicionada uma extensão baseada no princípio de estimação incremental e, em seguida, o algoritmo é mais uma vez modificado através do princípio de estimação em multiresolução. Para cada uma das três configurações, são realizados testes para comparar suas performances. O sistema é modelado através do princípio de fluxo óptico e dois controladores são apresentados para realimentar o sistema: um proporcional integral (PI) e um proporcional com estimação de perturbações externas através de um filtro de Kalman (LQG). Ambos são calculados utilizando um critério linear quadrático e os desempenhos deles também são analisados comparativamente.

Palavras-chave: Controle servo visual. Rastreamento Visual. Controle linear quadrático.

ABSTRACT

A pan-tilt camera can move around two rotational axes (pan and tilt), allowing its lens to be pointed to any point in space. A possible application of the camera is to keep it pointed to a certain moving target, through appropriate angular pan-tilt positioning. This work presents a visual servoing technique, which uses first the images captured by the camera to determinate the target position. Then the method calculates the proper rotations to keep the target position in image center, establishing a real-time and closed-loop system. The developed visual tracking technique is based on template region matching, and makes use of the sum of squared differences (SSD) as similarity criterion. An extension based on incremental estimation principle is added to the technique, and then the algorithm is modified again by multiresolution estimation method. Experimental results allow a performance comparison between the three configurations. The system is modeled through optical flow principle and this work presents two controllers to accomplish the system feedback: a proportional integral (PI) and a proportional with external disturbances estimation by a Kalman filter (LQG). Both are determined using a linear quadratic method and their performances are also analyzed comparatively.

Keywords: Visual servoing. Visual tracking. Linear quadratic control.

LISTA DE ILUSTRAÇÕES

Figura 1.1 – Exemplo da estrutura de um sistema de controle servo visual (baseada em uma figura presente no trabalho de Corke (1994)). 18

Figura 1.2 – Esquema de uma câmera pan-tilt e seus eixos de rotação. 19

Figura 2.1 – Exemplo ilustrativo de um algoritmo de rastreamento por segmentação. . . 22

Figura 2.2 – Exemplo ilustrativo de um algoritmo de rastreamento de contornos. 24

Figura 2.3 – Exemplo ilustrativo de um algoritmo de rastreamento de uma região de referência. 25

Figura 3.1 – Esquema de uma estrutura de controle servo visual não-hierárquica baseada em posição (criado a partir de uma figura presente no trabalho de Corke (1994)). 29

Figura 3.2 – Esquema de uma estrutura de controle servo visual hierárquica baseada em imagem (criado a partir de uma figura presente no trabalho de Corke (1994)). 30

Figura 4.1 – Diagrama de blocos do sistema. 33

Figura 4.2 – Esquema dos sistemas de coordenadas da câmera e do plano de formação da imagem. 35

Figura 4.3 – Esquema dos sistemas de coordenadas da câmera e do plano de formação da imagem (vista superior). 37

Figura 5.1 – Exemplo representando uma região-alvo e um padrão de referência genérico. 41

Figura 5.2 – Esquema do algoritmo baseado na minimização do erro quadrático. 47

Figura 5.3 – Esquema do algoritmo estendido através de estimação incremental. 49

Figura 5.4 – Exemplo ilustrativo do princípio de estimação em multiresolução. 50

Figura 5.5 – Esquema do algoritmo estendido por estimação em multiresolução. 52

Figura 6.1 – Diagrama de blocos do sistema com o controlador proporcional integral. . 61

Figura 6.2 – Diagrama de blocos do sistema com o controlador linear quadrático gaussiano. 64

Figura 7.1 – Esquema do equipamento utilizado. 67

Figura 7.2 – Ilustração do software em função manual (I). 71

Figura 7.3 – Ilustração do software em função manual (II). 71
Figura 7.4 – Ilustração do software em função de rastreamento. 72
Figura 8.1 – Imagem contendo a região de referência utilizada no teste do filtro. 74
Figura 8.2 – Conjunto de imagens utilizado no teste do filtro. 75
Figura 8.3 – Valores de deslocamento obtidos no teste do filtro. 76
Figura 8.4 – Valores de deslocamento obtidos no teste do filtro (continuação). 77
Figura 8.5 – Valores de deslocamento obtidos no teste do filtro (continuação). 78
Figura 8.6 – Valores de deslocamento obtidos no teste do filtro (continuação). 79
Figura 8.7 – Imagem contendo uma amostra da região de referência utilizada no teste do sistema. 81
Figura 8.8 – Imagem de amostra indicando o deslocamento imposto ao alvo no teste do sistema. 81
Figura 8.9 – Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo base de rastreamento e o controlador PI. 82
Figura 8.10 – Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo base de rastreamento e o controlador LQG. 83
Figura 8.11 – Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo de estimação incremental e o controlador PI. 84
Figura 8.12 – Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo de estimação incremental e o controlador LQG. 85
Figura 8.13 – Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo de estimação em multiresolução e o controlador PI. 86
Figura 8.14 – Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo de estimação em multiresolução e o controlador LQG. 87
Figura 8.15 – Gráficos de valores de deslocamento obtidos no teste do sistema para um alvo em movimento utilizando o controlador PI. 90
Figura 8.16 – Gráficos de valores de deslocamento obtidos no teste do sistema para um alvo em movimento utilizando o controlador LQG. 90

Figura A.1 – Esquema das distâncias envolvidas no experimento de determinação do valor da distância focal da câmera. 101

Figura B.1 – Gráfico da variação de R_X com x_1 e x_2. 104

Figura B.2 – Gráfico da variação de R_X com x_1 e x_2 (vista lateral). 104

LISTA DE TABELAS

Tabela A.1 – Valores de distância focal obtidos através do experimento. 101

LISTA DE ABREVIATURAS E SIGLAS

LQ	Linear Quadrático
LQG	Linear Quadrático Gaussiano
PI	Proporcional Integral
RGB	“Red”–“Green”–“Blue”
SSD	Soma dos quadrados das diferenças (“Sum of Squared Differences”)

LISTA DE SÍMBOLOS

$\mathbf{x} = [x_1, x_2]^T$	Vetor de variáveis de estado correspondentes às coordenadas (horizontal e vertical) do alvo segundo o sistema de coordenadas sobre o plano de formação da imagem
C	Origem do sistema de coordenadas sobre o plano de formação da imagem
O	Origem do sistema de coordenadas fixo na câmera e ponto focal do modelo de formação de imagem
X, Y, Z	Coordenadas da posição (3D) do objeto rastreado em relação ao sistema de coordenadas fixo na câmera
t_i	Instante de tempo genérico
T	Período de amostragem
$\mathbf{d} = [d_1, d_2]^T$	Vetor de perturbações externas sobre o modelo
$\mathbf{u} = [u_1, u_2]^T$	Vetor de variáveis de controle sobre o modelo
$\mathbf{w} = [w_1, w_2]^T$	Vetor de incertezas do modelo
$\mathbf{I}_i$	Matriz identidade de dimensão i
$\mathbf{y} = [y_1, y_2]^T$	Vetor de medidas
$\mathbf{v} = [v_1, v_2]^T$	Vetor de ruídos nas medidas
$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{E}, \boldsymbol{G}$	Matrizes do modelo do sistema
R_X	Velocidade de rotação tilt
R_Y	Velocidade de rotação pan
f	Distância focal da câmera
$\Re$	Região-alvo de uma imagem
$\mathbf{p}_j$	Posição (horizontal e vertical) de um pixel na imagem
N	Número de pixels da região-alvo
$\mathbf{f}$	Função paramétrica de modelagem de movimento do padrão a ser rastreado
$\mu = [\mu_1, \mu_2, \mu_3, \mu_4]^T$	Vetor de parâmetros (translação horizontal, translação vertical, rotação e escalonamento) da função $\mathbf{f}$
I	Intensidade de um pixel
$\mathbf{I}$	Vetor de intensidades dos pixels da região-alvo deslocados por $\mathbf{f}$

$\mathbf{I}(t_0)$	Vetor de intensidades dos pixels correspondentes ao padrão de referência
$\delta\mu$	Incremento adicionado ao valor de μ a cada iteração
SSD	Função objetivo a ser minimizada no cálculo de um filtro
∇	Função gradiente
$\boldsymbol{M}_0, \boldsymbol{\Lambda}, \boldsymbol{\Sigma}$	Matrizes utilizadas na iteração dos algoritmos de filtragem
$n\_iteracoes$	Número máximo de iterações permitido
$erro_j$	Valor mínimo de erro relativo permitido para μ_j
nts_j	Indica se houve troca de sinal no valor de $\delta\mu_j$
$iteracao$	Número de iterações já realizadas
sin_j	Sinal de $\delta\mu_j$ obtido na primeira iteração
$\mathrm{sinal}(x)$	Sinal (positivo ou negativo) de x
I^2	Valor de I obtido a partir de uma imagem convertida para uma resolução duas vezes menor do que a original (I^1)
J	Função objetivo a ser minimizada para os cálculos dos controladores LQ
t_{n+1}	Tempo final de interesse considerado para os controladores LQ
$\boldsymbol{X_f}, \boldsymbol{X}, \boldsymbol{S}, \boldsymbol{U}$	Matrizes de peso utilizadas no cálculo dos controladores LQ
$\boldsymbol{K_c}$	Matriz de ganhos dos controladores
$\boldsymbol{X_c}$	Solução da equação de Riccati para os controladores LQ
$\mathrm{E}(x)$	Função de média de x
$\boldsymbol{Q}, \boldsymbol{R}$	Matrizes de covariância de $\mathbf{w}$ e $\mathbf{v}$ (ruídos brancos)
t_i^-, t_i^+	Instantes de tempo imediatamente anterior e posterior à incorporação de novas medidas
$\hat{\mathbf{x}}$	Vetor de estados estimado pelo filtro de Kalman
$\boldsymbol{P}$	Matriz de covariâncias de $\hat{\mathbf{x}}$
$\boldsymbol{K_f}$	Matriz de ganhos de Kalman
$\mathbf{q}$	Vetor de estados adicionais do controlador PI
$\mathbf{x_0}$	Valor de equilíbrio de $\mathbf{x}$ para o controlador PI
$\delta\mathbf{x}$	Perturbação em torno do valor de equilíbrio de $\mathbf{x}$ para o controlador PI
$\mathbf{x_a}$	Valor de $\mathbf{x}$ do modelo aumentado
$\mathbf{w_d}$	Vetor de incertezas do modelo das perturbações externas do controlador LQG

$\boldsymbol{Q_d}$	Matriz de covariâncias de $\mathbf{w_d}$ (ruído branco)
$\hat{\mathbf{d}}$	Vetor de perturbações externas estimado pelo filtro de Kalman
$\mathbf{u_n}$	Vetor de controle modificado utilizado no controlador LQG
$pan, tilt$	Posições angulares pan e tilt da câmera
I_R, I_G, I_B	Componentes vermelha, verde e azul de um pixel colorido (RGB)

SUMÁRIO

1 INTRODUÇÃO 17

1.1 Controle Servo Visual 17

1.2 Câmera Pan-Tilt 18

1.3 Objetivos e Estrutura da Dissertação 20

2 RASTREAMENTO VISUAL 21

2.1 Rastreamento por Segmentação 22

2.2 Rastreamento de Contornos e Bordas 23

2.3 Rastreamento de uma Região de Referência 24

2.4 Algoritmo de Rastreamento Visual Escolhido 26

3 CONTROLE SERVO VISUAL 28

3.1 Estrutura de Controle Hierárquica ou Não-hierárquica 28

3.2 Controle Baseado em Posição ou em Imagem 29

3.3 Algoritmos de Controle Servo Visual Baseado em Imagem 30

3.4 Algoritmo de Controle Servo Visual Escolhido 32

4 MODELAGEM DO SISTEMA 33

4.1 Modelo do Sistema 34

5 PROJETO DO FILTRO 40

5.1 Algoritmo Baseado na Minimização do Erro Quadrático 40

5.2 Extensão do Algoritmo Através de Estimação Incremental 48

5.3 Extensão do Algoritmo por Meio de Estimação em Multiresolução 50

6 PROJETO DO CONTROLADOR 53

6.1 Controle Linear Quadrático 53

6.2 Filtro de Kalman 55
6.3 Controlador Proporcional Integral 57
6.4 Controlador Proporcional com Estimação da Perturbação 62

7 EQUIPAMENTO UTILIZADO E IMPLANTAÇÃO DO SOFTWARE 67

7.1 Câmera 67
7.2 Módulo Pan-tilt 68
7.3 Computador e Software 70

8 RESULTADOS EXPERIMENTAIS 73

8.1 Teste do Filtro 73
8.2 Teste do Sistema 80
8.3 Teste do Sistema para um Alvo em Movimento 89

9 CONCLUSÕES 92

Referências Bibliográficas 95

Apêndice A – Cálculo da distância focal da câmera 100

Apêndice B – Limitações de movimento do alvo e influência de aproximações realizadas 102

1 INTRODUÇÃO

O uso de câmeras de vídeo está tradicionalmente relacionado à necessidade de registrar visualmente determinadas situações, tornando possível sua visualização posteriormente. Entretanto, sua aplicação em conjunto com técnicas de visão computacional vem se tornando cada vez mais freqüente e é tema de diversos trabalhos de pesquisa.

Uma imagem pode ser considerada uma representação virtual de um determinado ambiente físico. Assim, a área de visão computacional tem como objetivo analisar e interpretar imagens, visando obter informações sobre o ambiente representado. Uma de suas mais importantes aplicações consiste na utilização de câmeras como sensores de posição em sistemas robóticos. Neste caso, técnicas de visão são utilizadas para determinar a localização de determinados objetos (alvos) a partir das imagens geradas.

A utilização de câmeras como sensores tem algumas vantagens. Uma delas é o fato de permitir a realização de medidas sem necessidade de contato direto com o ambiente, através da análise de sua projeção na imagem. Outra vantagem é que a câmera possibilita uma maior visão do espaço de trabalho e, dessa maneira, uma maior capacidade de adaptação do robô ao ambiente, em relação aos sensores de posição mais comuns. Uma das grandes desvantagens, o alto custo computacional de aplicações em tempo real, vem sendo cada vez mais minimizada com o desenvolvimento de novas tecnologias em processadores e computadores.

Um dos principais segmentos que se utiliza dos benefícios da aplicação de visão computacional em sistemas robóticos é o chamado controle servo visual.

1.1 Controle Servo Visual

De acordo com Corke (1994), nos trabalhos tradicionais que utilizavam visão em robótica, a informação visual recebida por uma câmera era aplicada no controle de um robô na forma de malha aberta. Ou seja, a posição de um determinado alvo era obtida *a priori*, a partir da análise de imagens, para então o robô se movimentar. Dessa maneira, durante uma operação, eventuais erros provenientes do sistema de posicionamento do robô ou do sistema de visão tendem a se acumular, prejudicando a precisão dos seus movimentos. Assim, foi criado em 1979 por Hill e Park (apud CORKE, 1994) o conceito de controle servo visual (“visual servoing”), em que a informação visual obtida pela câmera é utilizada em um laço de realimentação (Figura 1.1).

O controle do sistema passa a ser em malha fechada, com os dados obtidos pelo sensor visual adquiridos e atualizados em tempo real. Com isso, os erros na determinação da posição do alvo e de posicionamento do robô podem ser minimizados pelo controlador do sistema.

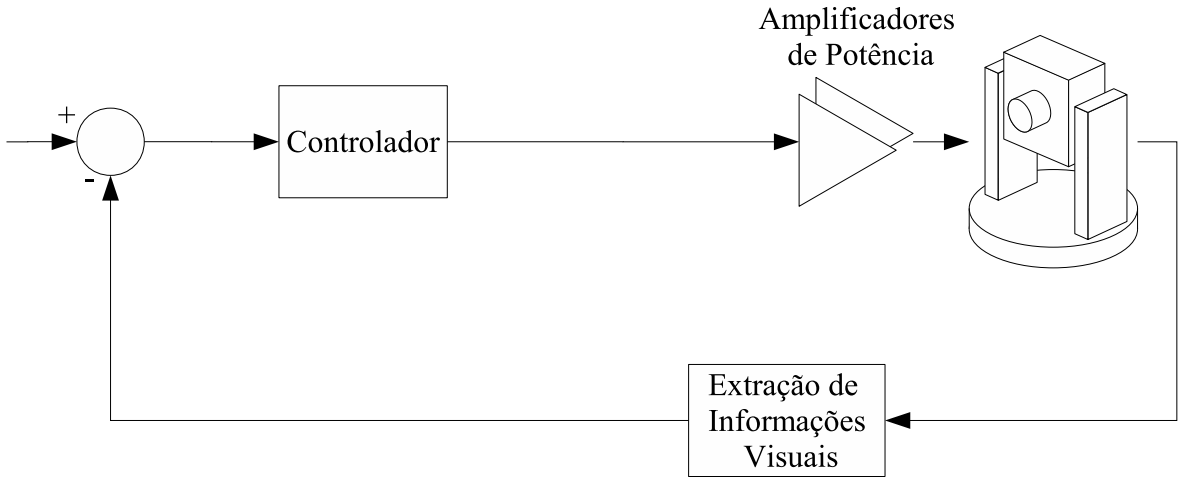

Figura 1.1: Exemplo da estrutura de um sistema de controle servo visual (baseada em uma figura presente no trabalho de Corke (1994)).

Há diversas aplicações que utilizam um sistema de controle servo visual. Jung e Sukhatme (2004) apresentam um robô que utiliza esta metodologia para detectar e se desviar de obstáculos ao ar livre. Papanikolopoulos, Khosla e Kanade (1993) utilizam a informação visual no controle de robôs manipuladores. Corke (1994) cita outras aplicações, como robôs para jogos de ping-pongue e para colheita de frutas.

Um outro aparato bastante utilizado que emprega controle servo visual é a chamada câmera pan-tilt.

1.2 Câmera Pan-Tilt

Uma câmera pan-tilt é composta por um dispositivo motorizado acoplado à câmera em si, possibilitando sua rotação segundo dois eixos distintos, pan e tilt (Figura 1.2). Variando os valores dos ângulos pan e tilt, é permitida à câmera uma visão de qualquer ponto ao seu redor (mas que, na prática, é limitada devido a aspectos construtivos).

Uma câmera móvel permite a realização de diversas aplicações, como videoconferências, ensino a distância e sistemas de vigilância (CHEN; CHENG; TSAI, 2002). Nas duas primeiras, o objetivo é permitir ao interlocutor se movimentar livremente, sem a preocupação de se

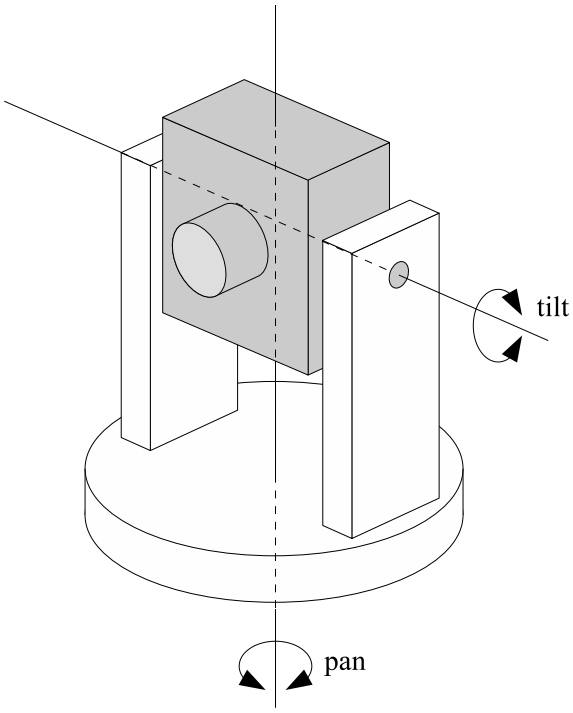

Figura 1.2: Esquema de uma câmera pan-tilt e seus eixos de rotação.

manter dentro do enquadramento da câmera. Assim, o mecanismo pan-tilt é o responsável por acompanhar o movimento da pessoa. Para tanto, é necessário um algoritmo para identificar um indivíduo e deslocar a câmera de modo a mantê-lo na região central da imagem. Já em sistemas de vigilância, a função do aparato é o monitoramento de um ambiente. Uma aplicação possível é a detecção de um intruso, em que a câmera deve identificar sua presença e acompanhar o seu movimento, mantendo-o no centro da imagem. O monitoramento pode ser realizado também em relação a um alvo móvel pré-determinado. Nesse caso, a presença de um alvo já é garantida e o dispositivo deve se movimentar para que ele permaneça dentro do enquadramento da câmera.

Câmeras pan-tilt podem ser utilizadas também em conjunto com outros sistemas robóticos. Crétual e Chaumette (2000) empregam uma câmera móvel em um robô submarino. Dado que o robô sofre interferência em seu posicionamento devido ao movimento da água em sua volta, é necessário estabilizar a imagem recebida, para que esta não se mostre trêmula. Já Clady et al. (2001) utilizam uma câmera pan-tilt-zoom (ou seja, além de possibilitar movimento em relação aos ângulos pan e tilt, possui também controle de zoom) em um sistema de assistência para direção de automóveis. Sua função é detectar e rastrear outros veículos próximos. Outra aplicação é dada por Ghidary et al. (2000), que utilizam uma câmera pan-tilt-zoom em um robô doméstico para rastrear faces humanas.

1.3 Objetivos e Estrutura da Dissertação

O objetivo deste trabalho é desenvolver um algoritmo de controle servo visual para câmeras pan-tilt. A sua premissa é que, dado um alvo móvel qualquer, a câmera se movimente de modo a mantê-lo no centro das imagens recebidas, possibilitando um monitoramento constante.

Segundo Espiau, Chaumette e Rives (1992), a abordagem de sistemas baseados em controle servo visual pode ser dividida em duas etapas. A primeira corresponde à extração de informações visuais. No caso da câmera pan-tilt, o objetivo é determinar a posição do alvo em cada imagem obtida, o que é denominado rastreamento visual. A segunda parte é a implementação do algoritmo de controle em si, de modo a exercer sobre o sistema controlado uma resposta adequada às informações visuais obtidas. Neste caso, corresponde à movimentação da câmera segundo os eixos pan e tilt para manter o alvo no centro da imagem.

A dissertação está organizada da seguinte maneira. O Capítulo 2 contém métodos pesquisados de rastreamento visual. O Capítulo 3 apresenta métodos de controle servo visual para a realimentação do sistema. No Capítulo 4 se encontra o modelo do sistema a ser controlado. Já os Capítulos 5 e 6 descrevem, respectivamente, os métodos de rastreamento (filtragem) e controle utilizados no trabalho. O Capítulo 7 apresenta o equipamento e os outros materiais utilizados nos experimentos. O Capítulo 8, por sua vez, descreve os experimentos realizados e a análise de seus resultados. Por fim, o Capítulo 9 apresenta as conclusões deste trabalho.

2 RASTREAMENTO VISUAL

Ao se utilizar uma câmera como um sensor de posição, a localização de um determinado objeto é realizada a partir da obtenção da posição de sua projeção na imagem. Assim, um procedimento de rastreamento visual consiste em localizar uma ou mais características desejadas em uma seqüência de imagens. Se o alvo que contém essas características ou a câmera se movimentam, é necessário determinar novas posições para cada imagem recebida. No caso de uma câmera pan-tilt, dado um determinado alvo, deseja-se obter a posição do seu centróide (centro de gravidade) nas imagens adquiridas. Assim, é possibilitada à câmera se movimentar convenientemente para manter o alvo no centro da imagem.

Geralmente, em um sistema de controle servo visual, a tarefa de rastreamento é a responsável pelo maior custo computacional durante a operação, o que é determinante em uma aplicação em tempo real. Silveira Filho (2002) cita algumas soluções para essa exigência. Uma delas é definir uma região de interesse. Dada uma seqüência de imagens contendo um elemento móvel, a rapidez de obtenção de cada imagem obedece uma certa freqüência. Se essa taxa é alta o suficiente, a variação de posição do elemento entre duas imagens consecutivas não é muito elevada. Neste caso, não é necessário que o rastreamento seja realizado a partir da imagem inteira, é considerada apenas uma região, contendo o alvo, próxima à última posição determinada. Outra alternativa possível para tornar o rastreamento mais rápido é reamostrar as imagens para resoluções menores. Dessa maneira, com a diminuição do tamanho da imagem, o espaço de busca também se reduz, embora, por outro lado, o nível de detalhamento também diminua. Assim, deve-se analisar o tamanho ideal de imagem a ser utilizado. Uma outra solução ainda é investir em equipamentos mais sofisticados, incluindo, por exemplo, sistemas dedicados, de aplicação específica ou arquiteturas paralelas.

Para desenvolver um método de rastreamento visual de um determinado elemento, é necessário analisar os fatores envolvidos na aplicação. Esses fatores correspondem, por exemplo, a características do alvo (forma, geometria, cor), condições do ambiente (iluminação, plano de fundo da imagem) e equipamento utilizado (restrições relativas à taxa de amostragem de imagens empregada). Grassi Junior (2002) e Silveira Filho (2002) classificam os métodos de rastreamento visual mais utilizados em três categorias, dadas a seguir.

2.1 Rastreamento por Segmentação

A segmentação envolve técnicas de processamento de imagens que permitem destacar determinadas regiões do restante da imagem (região de interesse), utilizando critérios pré-definidos. Esses critérios se baseiam em informações como cor, intensidade e movimento dos pixels que representam o alvo a ser rastreado. Obtida a região da imagem correspondente ao alvo, sua posição pode ser determinada através do seu centro de gravidade.

A Figura 2.1 mostra um exemplo de aplicação desta técnica. Dado que o alvo a ser rastreado é um quadrado cinza, o valor de intensidade correspondente ao tom de cinza do quadrado é utilizado como critério. Assim, é destacada na região de interesse de uma imagem recebida os pixels com valores próximos ao do quadrado.

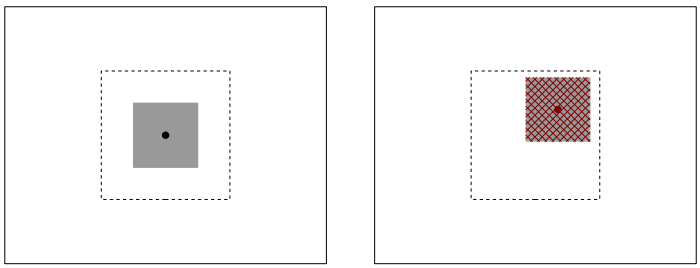

Figura 2.1: Exemplo ilustrativo de um algoritmo de rastreamento por segmentação.

As informações de cor ou intensidade são características utilizadas freqüentemente no rastreamento de faces humanas. Xu e Sugimoto (1998) realizam a detecção de faces a partir de um conhecimento prévio do valor de intensidade relativo à cor de pele. Este valor é utilizado como um limiar que separa os pixels correspondentes à pele dos restantes. Contudo, esta técnica pode apresentar problemas se há variação significativa de iluminação no ambiente. Para este caso, a solução apresentada por Cubber, Berrabah e Sahli (2004) é criar um modelo de ilumina-ção, permitindo retirar o efeito das mudanças provocadas pela variação de luz sobre os valores de intensidade dos pixels. Este modelo é construído e atualizado através de uma abordagem probabilística (redes Bayesianas).

Haritaoglu, Harwood e Davis (1999) e Ye et al. (2000) utilizam uma técnica que consiste em mapear previamente o ambiente, sem a presença do alvo, e armazenar as informações em um banco de imagens. Estas imagens do ambiente representam o plano de fundo da câmera, sendo que o plano principal corresponde somente ao alvo. Assim, durante o rastreamento, é realizada uma subtração entre uma imagem recebida e uma armazenada conveniente, de modo

que o resultado descartaria o fundo da imagem e apontaria somente o alvo. Huang et al. (2002) também utilizam esta técnica, em conjunto com um procedimento de segmentação pela cor de pele.

Murray e Basu (1994) apresentam uma técnica chamada de rastreamento por energia de movimento. Ela é baseada na diferença entre duas imagens obtidas em instantes de tempo de amostragem consecutivos e um algoritmo de detecção de bordas, o que resultaria no contorno do alvo. Entretanto, se a câmera se movimenta no período de tempo entre aquisições, o fundo das imagens também apresenta um movimento relativo, de modo que ele não se mantém constante. Assim, a subtração de imagens apresentaria detalhes relativos ao fundo, e não somente ao alvo, prejudicando sua detecção. Para contornar este problema, é aplicado um procedimento de compensação do deslocamento da câmera. Assim, o efeito do movimento da câmera sobre as imagens pode ser retirado e o fundo mantido invariável. Chen, Cheng e Tsai (2002) também utilizam este princípio, mas com uma técnica baseada em diferenças obtidas a partir de três imagens.

2.2 Rastreamento de Contornos e Bordas

Os contornos e as bordas em uma imagem estão geralmente associados às características geométricas do alvo a ser rastreado. Dessa maneira, os critérios utilizados se baseiam em elementos como retas, cantos, circunferências e outros conjuntos de pontos de interesse. A tarefa de definição dos elementos mais adequados depende das características particulares do alvo. Além disso, devem ser levados em conta outros fatores, como as características de iluminação, os detalhes do plano de fundo da imagem e a possibilidade de oclusão do alvo por outro objeto no ambiente.

A Figura 2.2 ilustra um exemplo de aplicação desta técnica. Considerando o mesmo exemplo anterior, pode ser utilizado um procedimento de detecção de segmentos de retas. Assim, é possível encontrar as bordas que definem os limites do quadrado a ser rastreado.

Uma maneira de identificar formas geométricas é através de funções paramétricas, cujos parâmetros indicam suas propriedades (posições, inclinações, raios). Espiau, Chaumette e Rives (1992) apresentam exemplos dessas funções para algumas formas primitivas. Silveira Filho (2002) utiliza a função relativa a retas para detectar linhas correspondentes a estradas, em uma aplicação de navegação aérea. Yachi, Wada e Matsuyama (2000) aplicam técnicas de segmentação baseadas em cor, intensidade e subtração do fundo para rastreamento de faces humanas,

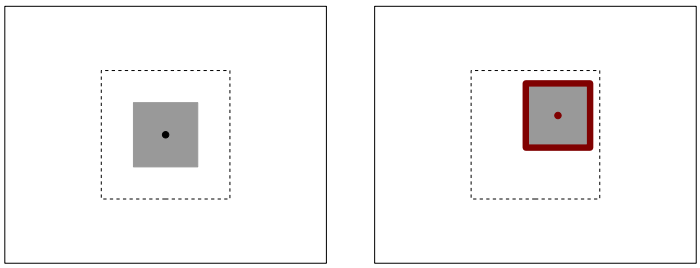

Figura 2.2: Exemplo ilustrativo de um algoritmo de rastreamento de contornos.

em conjunto com uma modelagem paramétrica de um rosto por meio de uma elipse.

Outra forma de detectar retas ou circunferências é através da chamada transformada de Hough, utilizada por Ghidary et al. (2000). Neste caso, a partir do contorno de uma pessoa obtido por segmentação por energia de movimento, a transformada é aplicada para se determinar uma circunferência, que corresponderia ao contorno de uma cabeça. Segmentação baseada em cor também é utilizada em conjunto nesta aplicação.

Um outro tipo de procedimento consiste em algoritmos que utilizam determinados critérios para selecionar de maneira automática pontos de interesse que formam bordas e contornos de um alvo. Definidos os pontos em uma imagem modelo, eles devem ser rastreados em cada imagem recebida para determinar as bordas do alvo. Lowe (1999) utiliza este princípio através de um critério baseado em filtros gaussianos, buscando selecionar pontos de interesse que não variem com efeitos como translação, escalonamento (aproximação ou afastamento do alvo) e mudanças de iluminação nas imagens.

2.3 Rastreamento de uma Região de Referência

Esta técnica consiste em, inicialmente, armazenar os valores dos pixels de uma região de interesse contendo o alvo a ser rastreado. Esta área é denominada região de referência e corresponde ao padrão a ser localizado em cada imagem recebida. Assim, deve ser determinada a posição da janela, dentre todas as possíveis contidas no interior de uma imagem, cujos pixels apresentam o melhor casamento com os da região de referência definida.

Uma possível aplicação desta técnica sobre o mesmo exemplo dos métodos anteriores pode ser encontrada na Figura 2.3. Inicialmente, define-se uma região de referência contendo o alvo. Assim, em cada imagem recebida, é realizada uma varredura de janelas em todas as posições

possíveis, permitindo determinar a posição daquela que é a mais semelhante à referência.

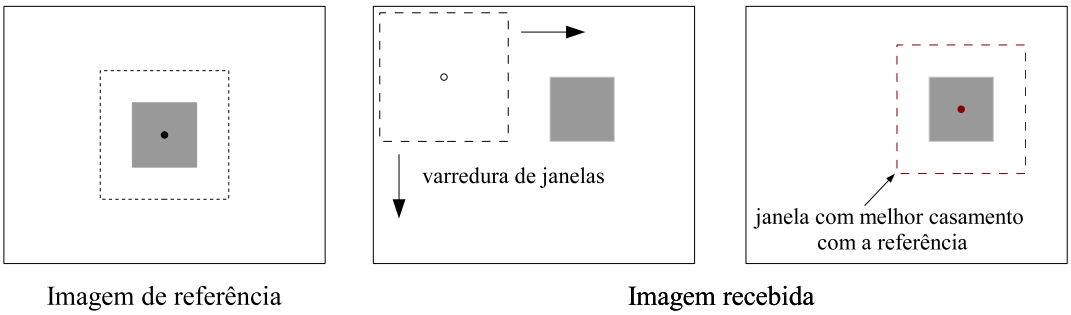

Figura 2.3: Exemplo ilustrativo de um algoritmo de rastreamento de uma região de referência.

Para quantificar o grau de semelhança entre a região de referência e uma janela qualquer, uma das funções de correlação mais utilizadas é a chamada soma dos quadrados das diferenças (“Sum of Squared Differences” – SSD). Neste caso, a janela que fornece o menor valor desta soma corresponde à janela com o melhor casamento com a referência.

O custo computacional envolvido em uma varredura do algoritmo sobre toda a imagem pode ser bastante elevado. Assim, geralmente, o cálculo das somas é realizado apenas para janelas localizadas em uma região próxima à última posição estimada. Esse princípio é utilizado por Suryadarma et al. (1997), aplicado em conjunto com um filtro de Kalman. Buscando ainda mais velocidade, Chen, Cheng e Tsai (2002) empregam uma técnica de busca hierárquica, que minimiza o número de posições candidatas de janelas para que devem ser calculadas as somas.

A técnica descrita anteriormente tem como hipótese que o movimento do alvo na imagem é de translação. Dessa maneira, outras distorções, como rotação, escalonamento e mudanças na iluminação do ambiente não são consideradas. Uma solução para inclui-las, dada por Suryadarma et al. (1997), é a utilização de múltiplas regiões de referência, obtidas para diversas configurações (de posição, rotação e iluminação) distintas do alvo. Outra abordagem, apresentada por Papanikolopoulos, Khosla e Kanade (1993), é a atualização constante da região de referência, ou seja, a janela obtida pelo algoritmo como o melhor casamento, em cada imagem recebida, se torna a nova referência. Assim, pequenas alterações no alvo podem ser incorporadas continuamente.

Hager e Belhumeur (1998) apresentam uma técnica desenvolvida a partir de uma função paramétrica de modelagem de movimento genérica. Essa função pode ser definida de acordo com as características de movimento a que o alvo está sujeito na aplicação (como translações, rotações e escalonamento). O critério de correlação utilizado também é a SSD. Além do modelo de movimento, são apresentados também um modelo para variação de iluminação e estimado-

res robustos para o caso de oclusão do alvo. O procedimento utiliza uma matriz jacobiana, calculada através de gradientes da imagem, que relaciona valores de intensidade dos pixels com os parâmetros da função de modelagem de movimento. Jurie e Dhome (2001), em técnica semelhante à de Hager e Belhumeur (1998), determinam o jacobiano através de uma fase de aprendizado, utilizando valores conhecidos para os parâmetros e as imagens correspondentes. Já Odobez e Bouthemy (1995) apresentam outro algoritmo semelhante, com modelagem de movimento, iluminação e estimadores robustos, mas a partir de um modelo paramétrico utilizando o princípio de fluxo óptico e estimação por mínimos quadrados. Em seguida, são aplicados dois princípios denominados estimação incremental e estimação em multiresolução.

2.4 Algoritmo de Rastreamento Visual Escolhido

Este trabalho tem como premissa possibilitar a realização do rastreamento de qualquer tipo de alvo em um ambiente indefinido. Como as técnicas de segmentação ou de contornos e bordas dependem diretamente de características particulares de um determinado alvo (como cor, intensidade e geometria), sua utilização em uma aplicação com alvos desconhecidos provavelmente não proporcionaria um desempenho satisfatório. Os procedimentos de rastreamento de bordas de seleção automática de pontos de interesse ou de segmentação por energia de movimento poderiam se aplicar a este caso, mas o algoritmo optado é o baseado em região de referência. Como o rastreamento de uma região de referência não depende diretamente do objeto de interesse, ele possui maior flexibilidade quanto ao tipo de alvo a ser rastreado.

Chen, Cheng e Tsai (2002) realizam uma comparação entre os métodos SSD de busca hierárquica (região de referência) e de energia de movimento (segmentação). O primeiro apresenta desempenho pior para grandes deslocamentos (maiores velocidades) do alvo, já que há possibilidade de sua saída da região de busca. Em compensação, o procedimento de energia de movimento é menos preciso. Ou seja, cada tipo de técnica apresenta algumas vantagens e outras desvantagens.

A técnica escolhida para este trabalho é a de Hager e Belhumeur (1998), que apresenta baixo custo computacional em relação aos algoritmos SSD de busca (varredura). Além disso, são tomadas como hipóteses que o ambiente de trabalho é fechado e bem iluminado e que não ocorre oclusão do alvo. Assim, os modelos para variação de iluminação e estimação robusta não precisam ser utilizados. Contudo, como desvantagem, este procedimento necessita de que haja mudanças pequenas entre duas imagens obtidas em instantes consecutivos. Dessa maneira,

o movimento do alvo é limitado para apenas pequenos deslocamentos, como no algoritmo de varredura estudado por Chen, Cheng e Tsai (2002). A maneira utilizada para contornar essa restrição é a incorporação de duas extensões apresentadas por Odobez e Bouthemy (1995): estimação incremental e estimação por multiresolução. No Capítulo 8, a influência de cada procedimento é comparada de maneira progressiva, ou seja, inicialmente, é analisado o algoritmo sem as extensões; em seguida, o ganho de desempenho devido ao passo de estimação incremental; e por último, a performance com a adição da técnica de estimação por multiresolução.

3 CONTROLE SERVO VISUAL

Após a aquisição de uma imagem e a aplicação de um algoritmo para sua interpretação, obtêm-se medidas do sistema observado. O objetivo do controlador servo visual é processar essas medidas e calcular as entradas a serem fornecidas ao sistema, completando a realimentação da malha de controle. No caso de uma câmera pan-tilt, as medidas obtidas se referem à posição do alvo rastreado, determinadas por um algoritmo de rastreamento visual. Assim, a função do controlador é fornecer o posicionamento adequado da câmera segundo os ângulos pan e tilt, de modo a manter o alvo centralizado na imagem.

Sanderson e Weiss (apud CORKE, 1994) introduzem duas classificações distintas para sistemas de controle servo visual. A apresentação realizada a seguir é simplificada e voltada para a aplicação em uma câmera pan-tilt. Detalhes mais completos sobre controle servo visual podem ser encontrados nos trabalhos de Corke (1994) ou Hutchinson, Hager e Corke (1996).

3.1 Estrutura de Controle Hierárquica ou Não-hierárquica

Em uma estrutura hierárquica (denominada “dynamic look-and-move”), a malha de controle é composta por dois laços de realimentação: um externo, que é o controle servo visual propriamente dito, e um interno, responsável pelo controle de posição do robô. Em geral, robôs manipuladores têm como entradas apenas sinais de posicionamento para suas articulações. Um controlador interno, por sua vez, é o responsável por realizar a movimentação adequada (acionar os atuadores para que sejam atingidas as posições ou mantidas as velocidades desejadas para os graus de liberdade). Assim, neste caso, a função do controle servo visual é gerar esses sinais de posicionamento para o controlador do robô.

Caso os sinais de controle sejam aplicados diretamente na movimentação dos atuadores do robô, não há malha de controle interna e a estrutura é não-hierárquica (“direct visual servo”). Esta arquitetura é pouco empregada, já que grande parte dos robôs industriais apresentam controladores internos, favorecendo a utilização de estruturas hierárquicas. Além disso, o custo computacional envolvido em algoritmos de visão faz com que a taxa de amostragem empregada em sistemas servo visuais seja relativamente baixa. Por outro lado, o controle de um manipulador industrial é complexo e não-linear, de forma que altas taxas de amostragem são necessárias para que o robô não se desestabilize. Assim, é preferível manter uma malha de controle interna

com uma freqüência de amostragem elevada para o posicionamento do robô.

A maioria das câmeras pan-tilt disponíveis no mercado apresentam controle interno de posicionamento, assim como o equipamento utilizado neste trabalho (ver o Capítulo 7). Por esse motivo, é empregada neste trabalho a estrutura hierárquica de controle.

3.2 Controle Baseado em Posição ou em Imagem

Em um sistema de controle baseado em posição, as informações visuais obtidas são utilizadas em conjunto com modelos geométricos do alvo e da câmera. A partir desses dados, gera-se a posição tridimensional do alvo no espaço, calculada em relação à posição da câmera. Essa posição do alvo em 3D é, então, fornecida ao controlador. Já no controle baseado em imagem, as informações obtidas da imagem (bidimensional) são utilizadas diretamente no controlador servo visual. Esta abordagem apresenta menor custo computacional (já que não é necessário determinar a posição em 3D do alvo) e elimina os erros provenientes da necessidade de modelagem e calibração da câmera. Por outro lado, há o problema da complexidade e não-linearidade do robô, que pode dificultar bastante a tarefa de posicionamento das juntas sem um modelo tridimensional de movimento do alvo.

A câmera pan-tilt apresenta somente dois graus de liberdade de rotação, de forma que seu posicionamento em relação a um alvo se movimentando no espaço não é complexo como o de um manipulador industrial. Por esse motivo, é utilizado no trabalho um sistema de controle baseado em imagem.

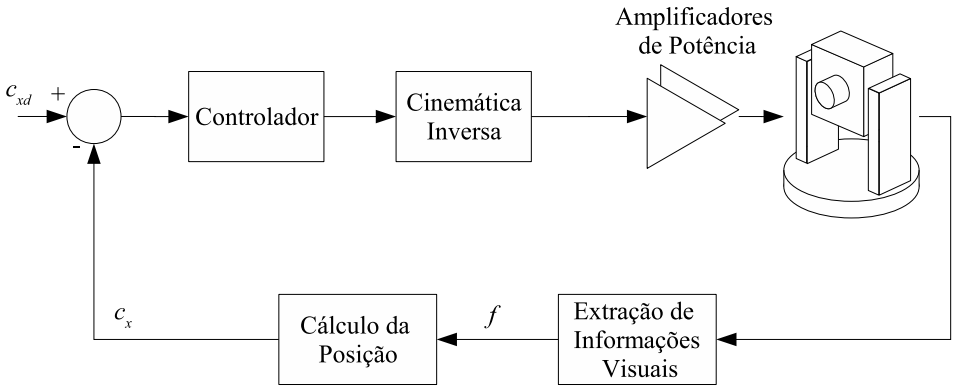

Figura 3.1: Esquema de uma estrutura de controle servo visual não-hierárquica baseada em posição (criado a partir de uma figura presente no trabalho de Corke (1994)).

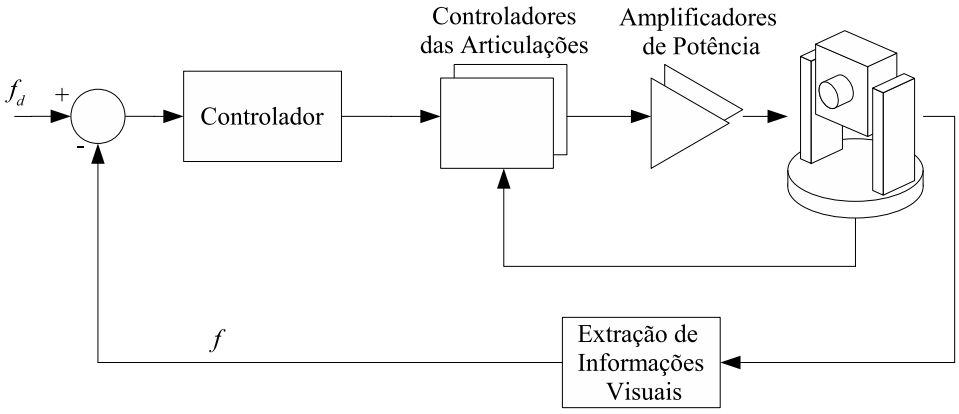

Figura 3.2: Esquema de uma estrutura de controle servo visual hierárquica baseada em imagem (criado a partir de uma figura presente no trabalho de Corke (1994)).

3.3 Algoritmos de Controle Servo Visual Baseado em Imagem

O procedimento mais simples de controle baseado em imagem dos graus de liberdade de um robô é o chamado "look-and-move". Para descrever este procedimento, pode ser considerado, como exemplo, o problema de manter uma câmera apontada para um determinado objeto (como no caso da câmera pan-tilt deste trabalho). Utilizando a imagem adquirida mais recente, obtém-se a última posição conhecida do alvo. Assim, a partir desta posição, a técnica "look-and-move" consiste em determinar os valores dos graus de liberdade do robô que permitam o alinhamento da câmera com o alvo. Dessa maneira, deslocando as juntas para as posições obtidas, o alvo passa a se localizar no centro da imagem (se ele não se movimentar).

Para um posicionamento adequado do robô, é necessário determinar o chamado jacobiano de imagem (HUTCHINSON; HAGER; CORKE, 1996), que corresponde a uma matriz que relaciona o vetor de velocidades dos parâmetros (alvo) da imagem com o vetor de velocidades do efetuador do robô. Para o caso de uma câmera pan-tilt, o jacobiano permite o cálculo das velocidades de rotação pan e tilt (graus/s) a partir da velocidade do alvo na imagem (pixels/s).

Uma maneira de determinar o jacobiano é através dos parâmetros de um modelo de formação de imagem da câmera. Chen, Cheng e Tsai (2002), por exemplo, realizam o cálculo utilizando um modelo de projeção perspectiva e o valor de distância focal da câmera. Outra possibilidade é determinar o jacobiano empiricamente. Impondo deslocamentos conhecidos no efetuador e determinando os deslocamentos correspondentes dos parâmetros do alvo na imagem, é possível calcular o jacobiano através de algum método de estimação. Yoshimi e Allen (1994) utilizam este método e, além disso, o jacobiano é atualizado constantemente durante a execução do algoritmo. Isso lhe permite maior adaptabilidade para diferentes condições, em

relação aos procedimentos que empregam jacobiano de valor fixo. Assim, o espaço de atuação permitido ao robô é maior, em troca de um também maior custo computacional. Outro exemplo é dado por Piepmeier, McMurray e Lipkin (1999), que formulam o cálculo dos valores dos graus de liberdade do robô como um problema de otimização não-linear. A solução apresentada consiste em um método de Quasi-Newton, em que os valores dos graus de liberdade e do jacobiano são atualizados de forma iterativa ao longo do tempo.

A estratégia "look-and-move" tem como desvantagem o fato de considerar exclusivamente a última posição conhecida do alvo como informação para realizar a movimentação da câmera. Se o alvo se encontra em movimento, o rastreamento fica defasado e pode ser prejudicado. Uma solução possível é dada por Xu e Sugimoto (1998), que utilizam um modelo cinemático de movimento do alvo para prever a sua posição futura. É assumido que o alvo se desloca com aceleração constante e o cálculo é realizado com base nas posições obtidas através das três imagens recebidas mais recentes. O jacobiano, neste caso, também é calculado através de projeção perspectiva e distância focal. Já Suryadarma et al. (1997) estimam a posição futura do alvo através de um filtro de Kalman.

Uma abordagem clássica de controle servo visual baseado em imagem pode ser encontrada nos trabalhos de Chaumette, Rives e Espiau (1991) ou Espiau, Chaumette e Rives (1992). A partir de um conjunto definido de parâmetros a serem rastreados nas imagens, é utilizado o conceito de função de tarefa. Essa função define um vetor de erros entre os valores de posição dos parâmetros em uma imagem e as posições desejadas. A minimização desse vetor de erros é realizada através de um modelo cinemático e uma lei de controle adaptativa. Nesse procedimento, o jacobiano de imagem é denominado matriz de interação, sendo que Espiau, Chaumette e Rives (1992) apresentam métodos para o seu cálculo considerando vários tipos de parâmetros (como pontos, retas e circunferências). Silveira Filho (2002) e Clady et al. (2001) também utilizam essa abordagem.

Papanikolopoulos, Khosla e Kanade (1993) utilizam um modelo de movimento projetado sobre a imagem baseado em um princípio denominado fluxo óptico. Ele inclui deslocamentos tanto do alvo como da câmera, além de entradas de ruídos. O modelo é simples, linear e são testados controladores proporcional integral (PI), adaptativos e linear quadrático gaussiano (LQG) para sua estabilização. O cálculo do jacobiano é realizado através de projeção perspectiva e distância focal.

Um outro procedimento de controle servo visual baseado em imagem é dado por Abdel-Hakim e Farag (2005), que aplicam a técnica de forças virtuais no controle ("look-and-move") de uma câmera pan-tilt. Já Cubber, Berrabah e Sahli (2004) apresentam um modelo de movi-

mento mais complexo, que necessita de duas etapas de funcionamento: a primeira utiliza um controlador PI, em que os parâmetros do modelo são identificados. Obtidos os parâmetros, entra em ação a segunda etapa, que utiliza controle adaptativo e um filtro de Kalman. Ghidary et al. (2000), por sua vez, aplicam controle fuzzy, dispensando a formulação de um modelo de movimento do alvo.

3.4 Algoritmo de Controle Servo Visual Escolhido

Em grande parte dos trabalhos que utilizam câmeras pan-tilt, os algoritmos de rastreamento visual recebem um enfoque bem maior em relação ao dado às técnicas de controle. Como a tarefa de posicionamento segundo os eixos pan e tilt não é complexa como, por exemplo, o controle de posição de um manipulador industrial, ela é freqüentemente deixada em segundo plano. Dessa maneira, o uso de um algoritmo formal de controle servo visual muitas vezes não é realizado. Contudo, a aplicação de uma técnica de controle neste trabalho é optada visando garantir um desempenho mais eficiente ao sistema.

A estratégia de controle dada por Espiau, Chaumette e Rives (1992) é bastante abrangente, sendo utilizada em várias aplicações. É uma técnica relativamente complexa, utilizada em sistemas elaborados como robôs manipuladores (CHAUMETTE; RIVES; ESPIAU, 1991) e veículos aéreos (SILVEIRA FILHO, 2002). Mas Clady et al. (2001) e Crétual e Chaumette (2000) também empregam o procedimento no controle de uma câmera pan-tilt, que é uma aplicação mais simples.

No entanto, o algoritmo de Papanikolopoulos, Khosla e Kanade (1993), por se tratar de uma técnica relativamente simples, é o utilizado neste trabalho. A abordagem apresentada envolve a utilização de um modelo simples e linear, de modo que vários tipos usuais de controladores podem ser empregados. Seguindo dois dos procedimentos de controle apresentados pelos autores, são utilizados neste trabalho um controlador proporcional integral (PI) e um proporcional com estimação das perturbações externas (deslocamentos do alvo). Assim, é possível uma análise comparativa de desempenho do sistema com cada uma das estratégias. Neste trabalho em particular, o critério escolhido para a determinação dos controladores é o linear quadrático (LQ). Assim, como a estimação das perturbações externas na segunda estratégia é realizada através de um filtro de Kalman, o controlador é denominado linear quadrático gaussiano – LQG (PAPANIKOLOPOULOS; KHOSLA; KANADE, 1993).

4 MODELAGEM DO SISTEMA

O sistema proposto neste trabalho, composto pela câmera pan-tilt e o controle servo visual, pode ser representado pelo esquema da Figura 4.1.

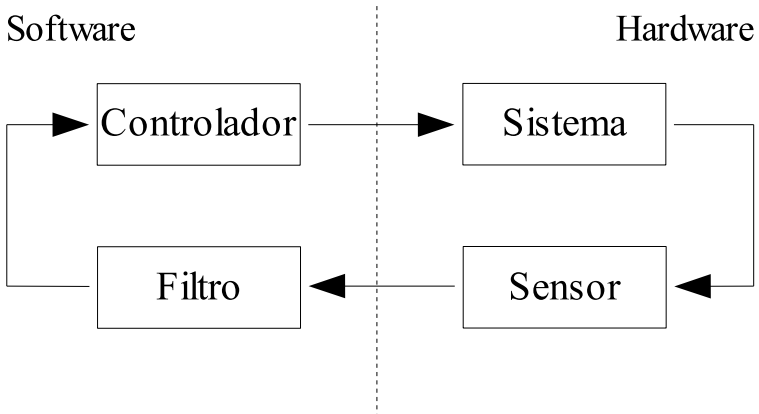

Figura 4.1: Diagrama de blocos do sistema.

O esquema é composto por quatro blocos, sendo que os blocos à direita (hardware) correspondem aos componentes físicos do sistema e os à esquerda (software) aos algoritmos (programas) que realizam cálculos matemáticos, possibilitando a realimentação da malha de controle.

Na parte de hardware, o bloco “Sistema” representa o dispositivo físico que deve ser controlado, ou seja, a câmera pan-tilt. Já o bloco “Sensor” corresponde ao instrumento que fornece dados (medidas) relativos ao ambiente em que o sistema se encontra. Neste caso, ele é o sensor da câmera responsável por captar e digitalizar as imagens.

Na subdivisão de software, o bloco “Filtro” tem a função de processar as medidas obtidas pelo sensor, de modo a obter parâmetros adequados (variáveis de estado) para serem utilizados pelo controlador. Dessa maneira, o filtro é responsável por realizar o rastreamento visual, ou seja, analisar as imagens fornecidas pela câmera e calcular a posição do alvo. O bloco “Controlador”, por sua vez, deve utilizar as variáveis de estado determinadas pelo filtro para calcular os sinais de entrada (variáveis de controle) a serem fornecidos aos atuadores do sistema, para que este se comporte da maneira desejada. Assim, a partir da posição do alvo, o controlador servo visual deve determinar os valores segundo as direções pan e tilt a serem fornecidos ao mecanismo de movimentação da câmera.

Portanto, dada a câmera pan-tilt (hardware), o objetivo deste trabalho é desenvolver algoritmos correspondentes aos blocos “Filtro” e “Controlador” (software). Mas, antes disso, é apresentado a seguir o modelo matemático do sistema a ser controlado.

4.1 Modelo do Sistema

A modelagem do sistema é realizada seguindo o procedimento apresentado por Papanikolopoulos, Khosla e Kanade (1993), utilizando um princípio denominado fluxo óptico. Segundo Tsakiris (1988), fluxo óptico pode ser definido como o campo de velocidades gerado sobre o plano da imagem, seja pela projeção de objetos movendo em 3D, ou do movimento do observador (câmera) em relação ao ambiente, ou ainda de um movimento aparente, quando uma série de imagens fornece a ilusão de movimento. Assim, deslocamentos reais equivalem a alterações nos valores dos pixels das imagens, sendo que o fluxo óptico define a velocidade dessas alterações.

Antes de determinar o fluxo óptico correspondente ao alvo a ser rastreado neste trabalho, é necessário definir um sistema de coordenadas. Segundo a Figura 4.2, o sistema de coordenadas sobre o plano de formação da imagem é dado por um eixo x_1 horizontal positivo para a direta, um eixo x_2 vertical positivo para cima e uma origem C localizada no centro da imagem. É em relação a este sistema que é calculada a posição do alvo. Além disso, é definido também um sistema O_{XYZ} fixo à câmera, com eixos X e Y paralelos e com mesmo sentido de x_1 e x_2, respectivamente, e Z sobre o eixo óptico da câmera, mas com sentido apontando para trás. Este sistema fornece a posição 3D do alvo no espaço. A figura mostra também que o modelo de formação de imagem adotado neste trabalho é o de projeção perspectiva e modelo de câmera "pin-hole" (FAYMAN; SUDARSKY; RIVLIN, 1998). Dessa maneira, todos os raios de luz que vêm de um objeto seguem uma trajetória em linha reta e passam por um único ponto (ponto focal, representado por O) antes de formar a imagem.

Considerando somente deslocamentos horizontais e movimentos tanto do alvo como da câmera (devido às rotações pan e tilt), o fluxo óptico de um ponto (centróide do alvo, por exemplo) de coordenada x_1, em um instante de tempo de amostragem t_i, é dado por:

$$\dot{x}_1(t_i) = d_1(t_i) + u_1(t_i) \tag{4.1.1}$$

onde:

$d_1(t_i)$ = componente do fluxo óptico induzido pelo movimento do alvo (pixels/s)

$u_1(t_i)$ = componente do fluxo óptico induzido pelo movimento da câmera (pixels/s)

Assumindo que os valores de velocidade são constantes entre dois instantes de amostragem consecutivos:

$$\dot{x}_1(t_i) = \frac{x_1(t_{i+1}) - x_1(t_i)}{T}$$

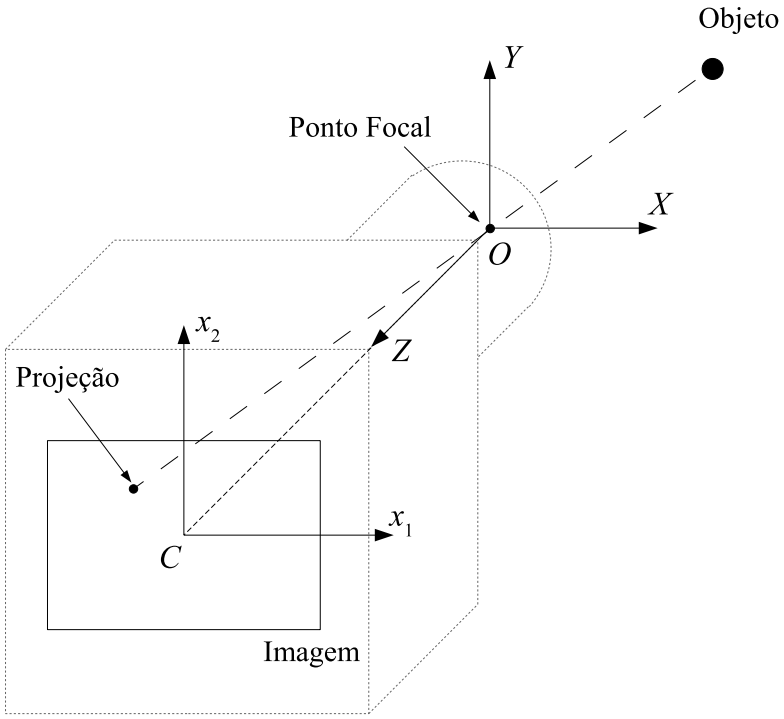

Figura 4.2: Esquema dos sistemas de coordenadas da câmera e do plano de formação da imagem.

onde:

T = período de amostragem (s)

Substituindo em (4.1.1):

$$x_1(t_{i+1}) = x_1(t_i) + T.u_1(t_i) + T.d_1(t_i) + w_1(t_i)$$

onde:

$w_1(t_i)$ = incertezas e ruídos sobre o modelo (pixels)

Percebe-se na equação um termo adicional w_1. Este termo se refere a possíveis incertezas e ruídos a que o modelo pode estar sujeito. Considerando agora as componentes do movimento projetadas tanto no eixo horizontal como no vertical da imagem, a equação é estendida para um sistema de equações matricial, dando origem a um sistema de equações de diferenças.

$$\mathbf{x}(t_{i+1}) = \boldsymbol{A}\mathbf{x}(t_i) + \boldsymbol{B}\mathbf{u}(t_i) + \boldsymbol{E}\mathbf{d}(t_i) + \boldsymbol{G}\mathbf{w}(t_i) \tag{4.1.2}$$

onde:

$\mathbf{x}(t_i) = \begin{bmatrix} x_1(t_i) \\ x_2(t_i) \end{bmatrix}$, $\mathbf{u}(t_i) = \begin{bmatrix} u_1(t_i) \\ u_2(t_i) \end{bmatrix}$, $\boldsymbol{A} = \boldsymbol{G} = \mathbf{I}_2$

$$\mathbf{d}(t_i) = \begin{bmatrix} d_1(t_i) \\ d_2(t_i) \end{bmatrix}, \mathbf{w}(t_i) = \begin{bmatrix} w_1(t_i) \\ w_2(t_i) \end{bmatrix}, \boldsymbol{B} = \boldsymbol{E} = T.\mathbf{I}_2$$

Este sistema corresponde a um modelo em espaço de estados, sendo que $\mathbf{x}$ é o vetor de variáveis de estado e $\mathbf{u}$ é o vetor de variáveis de controle. Além disso, completam o modelo um vetor de perturbações externas, $\mathbf{d}$, e um vetor de incertezas, $\mathbf{w}$. Dessa maneira, para trazer o alvo para o centro da imagem, o objetivo do sistema de controle é anular o valor do vetor de estados $\mathbf{x}$.

Com a definição do modelo do sistema, deve ser definido um vetor de medidas dos sensores ($\mathbf{y}$):

$$\mathbf{y}(t_i) = \boldsymbol{C}\mathbf{x}(t_i) + \mathbf{v}(t_i) \tag{4.1.3}$$

onde:

$\mathbf{y}(t_i) = \begin{bmatrix} y_1(t_i) \\ y_2(t_i) \end{bmatrix}, \boldsymbol{C} = \mathbf{I}_2$

$\mathbf{v}(t_i) = \begin{bmatrix} v_1(t_i) \\ v_2(t_i) \end{bmatrix}$ = ruído nos sensores (pixels)

Considerando que as medidas recebidas correspondem às posições x_1 e x_2, calculadas pelo filtro (rastreamento visual), $\boldsymbol{C}$ é dada pela matriz identidade e o vetor de medidas é igual ao vetor de estados (somado ao vetor de possíveis ruídos sobre as medidas).

Um detalhe importante a ser observado é que, no modelo proposto, as variáveis de controle são dadas por velocidades lineares, enquanto que as entradas no sistema real são as velocidades de rotação nas direções tilt (R_X) e pan (R_Y). Dessa maneira, é necessário determinar a matriz jacobiana de imagem, que relaciona os valores de R_X e R_Y com os de u_1 e u_2.

Pela Figura 4.3, utilizando a distância focal da câmera e semelhança de triângulos, pode-se obter:

$$\frac{x_1}{X} = \frac{f}{-Z} \Rightarrow x_1 = -f\frac{X}{Z} \tag{4.1.4}$$

onde:

f = distância focal da câmera (pixels)

Analogamente:

$$\frac{x_2}{Y} = \frac{f}{-Z} \Rightarrow x_2 = -f\frac{Y}{Z} \tag{4.1.5}$$

OBS.: O sinal negativo aplicado em Z se refere ao fato de o cálculo de semelhança de

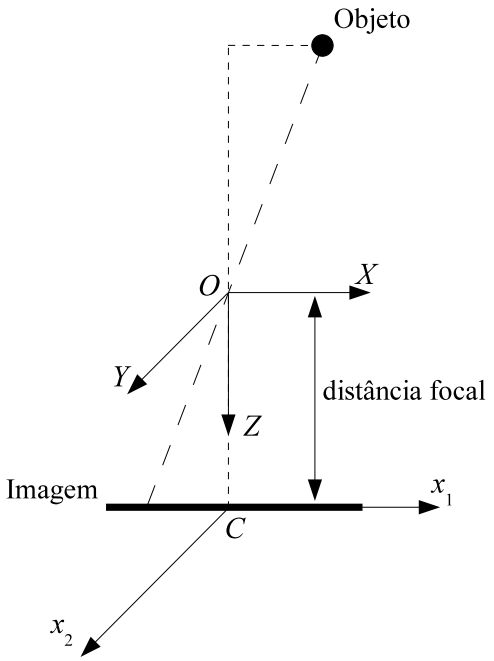

Figura 4.3: Esquema dos sistemas de coordenadas da câmera e do plano de formação da imagem (vista superior).

triângulos ser realizado em relação a **distâncias**, e não coordenadas. Assim, como é considerado que o objeto se localiza à frente da câmera (e, portanto, com coordenada Z negativa), uma correção é necessária. Assim sendo, o mesmo se aplicaria em relação à coordenada x_1 (que é negativa na Figura 4.3). Entretanto, as câmeras já realizam, geralmente, uma correção por padrão antes de fornecer as imagens (caso contrário, as imagens se mostrariam invertidas). Portanto, os valores de x_1 e X (assim como x_2 e Y) podem ser utilizados normalmente.

O jacobiano deve permitir promover velocidades u_1 e u_2 em um ponto de coordenadas x_1 e x_2 na imagem através das rotações R_X e R_Y da câmera. Em outras palavras, ele relaciona o movimento de um ponto na imagem devido somente ao movimento da câmera. Assim, para determiná-lo, pode-se considerar que o objeto-alvo se encontra fixo no espaço e o fluxo óptico é induzido somente pela rotação da câmera. Dessa maneira, diferenciando (4.1.4) e (4.1.5):

$$u_1 = \dot{x}_1 = -f\frac{\dot{X}}{Z} + f\frac{X}{Z^2}\dot{Z} \tag{4.1.6}$$

$$u_2 = \dot{x}_2 = -f\frac{\dot{Y}}{Z} + f\frac{Y}{Z^2}\dot{Z} \tag{4.1.7}$$

Considerando um objeto localizado na posição $\mathbf{r} = [X, Y, Z]^T$, sua velocidade $\dot{\mathbf{r}}$ é dada por:

$$\dot{\mathbf{r}} = (\dot{\mathbf{r}})_t + \boldsymbol{\omega} \times \mathbf{r}$$

onde:

$(\dot{\mathbf{r}})_t$ = velocidade de translação do objeto

$\boldsymbol{\omega}$ = velocidade angular do objeto

Apesar de, neste caso, o alvo ser considerado fixo no espaço, a rotação da câmera ocasiona em uma rotação do sistema de coordenadas O_{XYZ}. Assim, o alvo apresenta movimento em relação a este referencial.

Como o objeto não apresenta translação, $(\dot{\mathbf{r}})_t = \mathbf{0}$. Além disso, a câmera não sofre rotação em torno de Z. Assim, assumindo que os eixos de rotação de R_X e R_Y são, respectivamente, X e Y, a velocidade angular do **objeto em relação ao sistema de coordenadas fixo na câmera** é $\boldsymbol{\omega} = -[R_X, R_Y, 0]^T$. Logo:

$$\dot{\mathbf{r}} = \boldsymbol{\omega} \times \mathbf{r} = -\begin{vmatrix} \hat{X} & \hat{Y} & \hat{Z} \\ R_X & R_Y & 0 \\ X & Y & Z \end{vmatrix}$$

Logo:

$$\dot{X} = -\begin{vmatrix} R_Y & 0 \\ Y & Z \end{vmatrix} = -Z.R_Y$$

$$\dot{Y} = \begin{vmatrix} R_X & 0 \\ X & Z \end{vmatrix} = Z.R_X$$

$$\dot{Z} = -\begin{vmatrix} R_X & R_Y \\ X & Y \end{vmatrix} = X.R_Y - Y.R_X$$

Substituindo em (4.1.6) e (4.1.7):

$$u_1 = f.R_Y + f\left(\frac{X}{Z}\right)^2 R_Y - f\left(\frac{X}{Z}\right)\left(\frac{Y}{Z}\right)R_X$$

$$u_2 = -f.R_X + f\left(\frac{X}{Z}\right)\left(\frac{Y}{Z}\right)R_Y - f\left(\frac{Y}{Z}\right)^2 R_X$$

Substituindo (4.1.4) e (4.1.5):

$$u_1 = f.R_Y + f\left(-\frac{x_1}{f}\right)^2 R_Y - f\left(-\frac{x_1}{f}\right)\left(-\frac{x_2}{f}\right)R_X$$

$$u_2 = -f.R_X + f\left(-\frac{x_1}{f}\right)\left(-\frac{x_2}{f}\right)R_Y - f\left(-\frac{x_2}{f}\right)^2 R_X$$

Rearranjando as equações:

$$\left| \begin{array}{l} u_1 = -\dfrac{x_1 x_2}{f} R_X + \dfrac{x_1^2 + f^2}{f} R_Y \\ u_2 = -\dfrac{x_2^2 + f^2}{f} R_X + \dfrac{x_1 x_2}{f} R_Y \end{array} \right. \Rightarrow$$

$$\Rightarrow \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} -\frac{x_1 x_2}{f} & \frac{x_1^2+f^2}{f} \\ -\frac{x_2^2+f^2}{f} & \frac{x_1 x_2}{f} \end{bmatrix} \begin{bmatrix} R_X \\ R_Y \end{bmatrix}$$

Assim:

$$\begin{bmatrix} R_X \\ R_Y \end{bmatrix} = \begin{bmatrix} -\frac{x_1 x_2}{f} & \frac{x_1^2+f^2}{f} \\ -\frac{x_2^2+f^2}{f} & \frac{x_1 x_2}{f} \end{bmatrix}^{-1} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Portanto:

$$\begin{bmatrix} R_X \\ R_Y \end{bmatrix} = \begin{bmatrix} \frac{x_1 x_2}{(x_1^2+x_2^2+f^2)f} & -\frac{x_1^2+f^2}{(x_1^2+x_2^2+f^2)f} \\ \frac{x_2^2+f^2}{(x_1^2+x_2^2+f^2)f} & -\frac{x_1 x_2}{(x_1^2+x_2^2+f^2)f} \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

ou

$$R_X = \frac{u_1 x_1 x_2 - u_2(x_1^2 + f^2)}{(x_1^2 + x_2^2 + f^2)f} \tag{4.1.8}$$

$$R_Y = \frac{-u_2 x_1 x_2 + u_1(x_2^2 + f^2)}{(x_1^2 + x_2^2 + f^2)f} \tag{4.1.9}$$

Outras maneiras de realizar essa demonstração podem ser encontradas no trabalho de Tsakiris (1988) ou Papanikolopoulos, Khosla e Kanade (1993).

Pode-se observar que o jacobiano de imagem depende das posições x_1 e x_2 do alvo, que, por sua vez, se alteram conforme a câmera se movimenta. Assim, os valores de rotação correspondentes às entradas de controle se modificam com o próprio movimento da câmera. Entretanto, como aproximação, são aplicados valores constantes de rotação entre instantes de amostragem consecutivos. Essas rotações são calculadas utilizando as coordenadas x_1 e x_2 determinadas pelo filtro, que correspondem à última posição conhecida do alvo. No Apêndice B se encontra uma análise do efeito dessa aproximação (que é bastante pequeno) na tarefa de posicionamento. No Apêndice A, são apresentados métodos para determinar o valor da distância focal f da câmera.

5 PROJETO DO FILTRO

O filtro tem como função o processamento e a análise de imagens, de forma a determinar a posição do alvo rastreado no sistema de coordenadas sobre o plano de formação da imagem. Além disso, para um dos controladores projetados (LQG), é necessária também a aplicação de um filtro de Kalman. Entretanto, apesar de ser um filtro, como sua aplicação está diretamente relacionada ao controle do sistema, ele está apresentado juntamente com o controlador (Capítulo 6).

A seguir, são apresentados os três algoritmos de rastreamento visual utilizados neste trabalho. O primeiro é baseado em um critério de minimização do erro quadrático. O segundo procedimento, na realidade, é uma extensão do primeiro, utilizando-o como base em uma técnica denominada estimação incremental. De maneira semelhante, o terceiro algoritmo usa o segundo como base, englobando-o em uma técnica chamada estimação em multiresolução.

5.1 Algoritmo Baseado na Minimização do Erro Quadrático

O primeiro algoritmo de rastreamento desenvolvido é baseado em um procedimento apresentado por Hager e Belhumeur (1998). A abordagem utilizada a seguir é simplificada, direcionada para a aplicação deste trabalho (câmera pan-tilt).

Tomando-se como entrada do algoritmo uma imagem em tons de cinza, define-se, inicialmente, uma **região-alvo** $\Re = \{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \ldots, \mathbf{p}_N\}$, onde $\mathbf{p}_j = [x_1, x_2]^T$ é a posição (horizontal e vertical) de um pixel da imagem e N é o número de pixels contidos em $\Re$. A região-alvo consiste em um conjunto de pixels (janela) de posição e tamanho fixos. A aplicação do algoritmo é limitada apenas para janelas com este tamanho, não sendo realizada sobre a imagem inteira. Nesta aplicação em particular, cujo objetivo é manter o alvo no centro da imagem, a região-alvo é definida como uma janela quadrada centralizada na imagem.

Considera-se também um **padrão de referência** como sendo os valores (pré-definidos) de intensidade dos pixels da região-alvo que se deseja rastrear, ou seja, que contêm o alvo a ser seguido. Neste trabalho, considera-se que o alvo está localizado no centro do padrão de referência e que é obtido a partir da imagem recebida no instante de tempo t_0 (ver a Figura 5.1).

Ainda em relação ao alvo, assume-se que as possíveis transformações que ele pode sofrer são translação na direção dos eixos x_1 e x_2 (correspondente a um deslocamento no espaço do

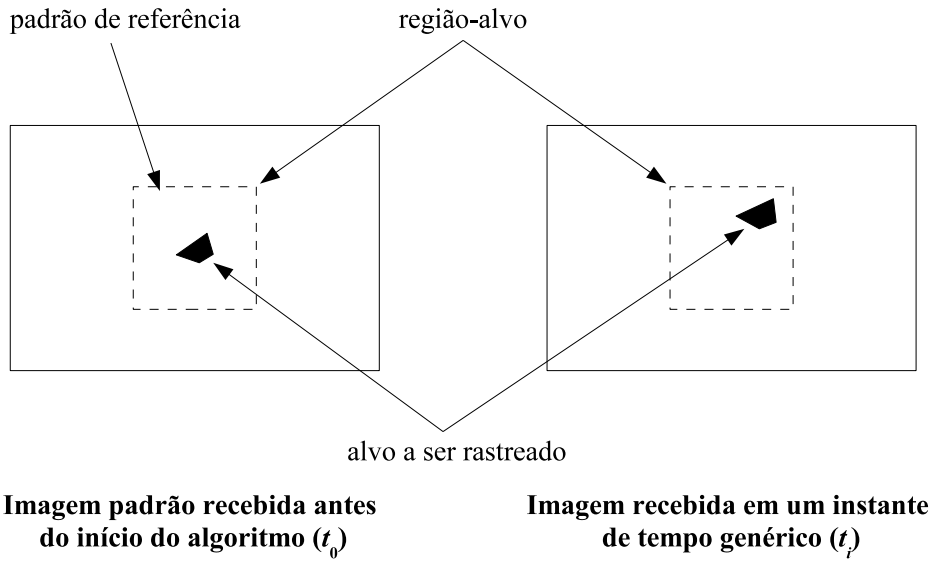

Figura 5.1: Exemplo representando uma região-alvo e um padrão de referência genérico.

objeto rastreado segundo as direções X e Y), rotação (rotação do objeto em torno do eixo Z) e escalonamento (deslocamento do objeto na direção Z, ou seja, aproximação ou afastamento da câmera). Então, define-se uma função paramétrica de modelagem de movimento $\mathbf{f}$, dada por:

$$\begin{aligned}\mathbf{f}\big(\mathbf{p}_j;\boldsymbol{\mu}(t_i)\big) &= \mu_4 \,.\, \boldsymbol{Rot}(\mu_3) \,.\, \mathbf{p}_j + \boldsymbol{\mu_t} = \\ &= \mu_4 \cdot \begin{bmatrix}\cos\mu_3 & -\operatorname{sen}\mu_3 \\ \operatorname{sen}\mu_3 & \cos\mu_3\end{bmatrix} \cdot \begin{bmatrix}x_1 \\ x_2\end{bmatrix} + \begin{bmatrix}\mu_1 \\ \mu_2\end{bmatrix}\end{aligned}$$

onde:

$\boldsymbol{\mu_t}(t_i) = [\mu_1(t_i), \mu_2(t_i)]^T$ = parâmetros correspondentes à translação (horizontal e vertical) do alvo (pixels)

$\mu_3(t_i)$ = parâmetro correspondente à rotação do alvo (radianos)

$\mu_4(t_i)$ = parâmetro correspondente ao escalonamento do alvo (adimensional)

$\boldsymbol{Rot}\big(\mu_3(t_i)\big)$ = matriz de rotação segundo um ângulo μ_3

$\boldsymbol{\mu}(t_i) = [\mu_1, \mu_2, \mu_3, \mu_4]^T$ = vetor de parâmetros da função de movimento

Assim, dado um pixel localizado em $\mathbf{p}_j$, a função $\mathbf{f}$ é responsável por determinar sua nova posição ao ser transladado horizontalmente e verticalmente por deslocamentos μ_1 e μ_2, respectivamente, rotacionado segundo um ângulo μ_3 (em relação ao centro da imagem), e escalonado por um fator μ_4.

Portanto, para cada imagem recebida contendo o alvo deslocado, o algoritmo de rastreamento visual deve determinar o valor do vetor de parâmetros $\boldsymbol{\mu}(t_i)$, que define uma determinada

função de modelagem de movimento $\mathbf{f}$. Esta função, ao ser aplicada sobre as posições dos pixels que formam a região-alvo da imagem adquirida, deve resultar em uma janela que seja semelhante ao padrão de referência. Em outras palavras, é necessário obter os parâmetros que definem a transformação sofrida pelo padrão de referência em cada imagem recebida.

Dentre os parâmetros a serem obtidos, podem-se destacar os de translação (μ_1 e μ_2), que devem ser fornecidos ao controlador para que possam ser compensados pelo mecanismo pan-tilt. Se a câmera apresentasse um controle de zoom e um grau de liberdade de rotação adicional em torno do eixo Z, os parâmetros μ_3 e μ_4 poderiam também ser minimizados.

Definindo $I(\mathbf{p}_j, t_i)$ como o valor (intensidade) do pixel na posição $\mathbf{p}_j$ no instante de tempo t_i, estabelece-se o vetor de intensidades $\mathbf{I}$:

$$\mathbf{I}(\boldsymbol{\mu}, t_i) = \begin{bmatrix} I\big(\mathbf{f}(\mathbf{p}_1, \boldsymbol{\mu}), t_i\big) \\ I\big(\mathbf{f}(\mathbf{p}_2, \boldsymbol{\mu}), t_i\big) \\ \vdots \\ I\big(\mathbf{f}(\mathbf{p}_N, \boldsymbol{\mu}), t_i\big) \end{bmatrix}$$

O vetor $\mathbf{I}$ contém os valores dos pixels, obtidos a partir da imagem recebida no instante t_i, que compõem a região-alvo deslocados segundo os parâmetros $\boldsymbol{\mu}$. Como o padrão de referência é obtido a partir da imagem adquirida no instante de tempo t_0, ele pode ser representado pelo vetor $\mathbf{I}\big(\boldsymbol{\mu}(t_0), t_0\big)$, dado que $\boldsymbol{\mu}(t_0) = [\mu_1(t_0), \mu_2(t_0), \mu_3(t_0), \mu_4(t_0)]^T = [0, 0, 0, 1]^T$ $\big($de forma que $\mathbf{f}\big(\mathbf{p}_j; \boldsymbol{\mu}(t_0)\big) = \mathbf{p}_j$, correspondendo ao padrão de referência sem sofrer transformações$\big)$. Para simplificar a notação, este vetor de intensidades é representado apenas como $\mathbf{I}(t_0)$.

Considera-se agora uma abordagem incremental.

$$\boldsymbol{\mu}(t_i) = \boldsymbol{\mu}(t_{i-1}) + \boldsymbol{\delta\mu}(t_i)$$

Assim, como $\boldsymbol{\mu}(t_0)$ é conhecido, $\boldsymbol{\mu}(t_i)$ é determinado através do cálculo do valor de $\boldsymbol{\delta\mu}(t_i) = [\delta\mu_1, \delta\mu_2, \delta\mu_3, \delta\mu_4]^T$ em cada instante de tempo.

Para determinar o valor de $\boldsymbol{\delta\mu}(t_i)$ que desloca a região-alvo e resulta na janela mais semelhante ao padrão de referência, o critério utilizado é o da soma dos quadrados das diferenças (“Sum of Squared Differences” – SSD). Dadas duas janelas de mesmo tamanho, contendo o mesmo número N de pixels, a SSD consiste, inicialmente, em realizar N subtrações pixel a pixel entre os valores de dois pixels situados na mesma posição, um em cada janela. Em seguida, cada uma das N diferenças é elevada ao quadrado, e a soma dos resultados fornece a SSD. A função das diferenças entre janelas é obter valores de erro pixel a pixel, a influência de seus

sinais (positivos ou negativos) é desprezada quando elevadas ao quadrado (os erros se tornam absolutos) e a soma final consiste no valor de erro total.

Se cada janela é representada por um vetor $\mathbf{I}$, a SSD pode ser dada pelo quadrado da norma da diferença entre os vetores. Assim, a SSD entre a janela que se deseja obter em uma imagem (região-alvo deslocada por $\mu(t_i)$) e o padrão de referência é dada por:

$$\begin{aligned} SSD(\delta\mu) &= \left\|\mathbf{I}\big(\mu(t_i), t_i\big) - \mathbf{I}(t_0)\right\|^2 \Rightarrow \\ \Rightarrow SSD(\delta\mu) &= \left\|\mathbf{I}\big(\mu(t_{i-1}) + \delta\mu(t_i), t_{i-1} + T\big) - \mathbf{I}(t_0)\right\|^2 \end{aligned} \tag{5.1.1}$$

onde:

SSD = função objetivo a ser minimizada no cálculo de um filtro

Portanto, deve ser encontrado o vetor $\delta\mu(t_i)$ que minimize a função SSD, em cada instante de tempo.

Expandindo em série de Taylor:

$$\mathbf{I}(\mu+\delta\mu, t_{i-1}+T) = \mathbf{I}(\mu, t_{i-1}) + \boldsymbol{M}(\mu, t_{i-1})\delta\mu + T.\mathbf{I_t}(\mu, t_{i-1}) + \text{t.o.s.}$$

onde:

$$\underset{(N\times 4)}{\boldsymbol{M}} = \nabla_{\boldsymbol{\mu}}\mathbf{I} = \left[\frac{\partial \mathbf{I}}{\partial \mu_1}, \frac{\partial \mathbf{I}}{\partial \mu_2}, \frac{\partial \mathbf{I}}{\partial \mu_3}, \frac{\partial \mathbf{I}}{\partial \mu_4}\right]$$

$$\underset{(N\times 1)}{\mathbf{I_t}} = \frac{\partial \mathbf{I}}{\partial t_i}$$

t.o.s. = termos de ordem superior

Desprezando os termos de ordem superior e substituindo em (5.1.1):

$$SSD(\delta\mu) = \|\mathbf{I}(\mu, t_{i-1}) + \boldsymbol{M}(\mu, t_{i-1})\delta\mu + T.\mathbf{I_t}(\mu, t_{i-1}) - \mathbf{I}(t_0)\|^2 \tag{5.1.2}$$

Segundo o método de aproximação discreta de derivada como diferença progressiva:

$$\mathbf{I_t}\big(\mu(t_{i-1}), t_{i-1}\big) = \frac{\mathbf{I}\big(\mu(t_{i-1}), t_i\big) - \mathbf{I}\big(\mu(t_{i-1}), t_{i-1}\big)}{T}$$

Substituindo em (5.1.2):

$$\begin{aligned} SSD(\delta\mu) &= \|\boldsymbol{M}(\mu, t_{i-1})\delta\mu + \mathbf{I}\big(\mu(t_{i-1}), t_i\big) - \mathbf{I}(t_0)\|^2 = \\ &= [\boldsymbol{M}\delta\mu + \mathbf{I}\big(\mu(t_{i-1}), t_i\big) - \mathbf{I}(t_0)]^T.[\boldsymbol{M}\delta\mu + \mathbf{I}\big(\mu(t_{i-1}), t_i\big) - \mathbf{I}(t_0)] \end{aligned}$$

O mínimo da função objetivo é obtido quando sua derivada é nula. De acordo com Brookes (2005), a derivada da função é dada por:

$$\frac{\partial SSD(\delta\mu)}{\partial(\delta\mu)} = 2\boldsymbol{M}^T[\boldsymbol{M}\delta\mu + \mathbf{I}\big(\mu(t_{i-1}), t_i\big) - \mathbf{I}(t_0)] = 0$$

A solução de mínimos quadrados é dada por:

$$\underset{(4\times1)}{\delta\mu} = -(\boldsymbol{M}^T\boldsymbol{M})^{-1}\boldsymbol{M}^T\big[\mathbf{I}\big(\mu(t_{i-1}), t_i\big) - \mathbf{I}(t_0)\big] \tag{5.1.3}$$

Em relação ao termo $\mathbf{I}\big(\mu(t_{i-1}), t_i\big)$, o instante em que μ é computado está explícito para frisar que a região-alvo utilizada é obtida a partir da imagem adquirida no instante t_i, mas deslocada segundo o vetor de parâmetros calculado no instante anterior (t_{i-1}).

A matriz $\boldsymbol{M}$ pode ser calculada, pela regra da cadeia, por:

$$\boldsymbol{M} = \nabla_{\boldsymbol{\mu}}\mathbf{I} = \underset{(N\times2)}{\nabla_{\mathbf{f}}\mathbf{I}} \,.\, \underset{(2\times4)}{\nabla_{\boldsymbol{\mu}}\mathbf{f}} \tag{5.1.4}$$

Assumindo que a estimativa obtida de μ é exata para todo instante de tempo t_i, o padrão de referência é sempre encontrado na posição dada por μ. Assim, pode-se considerar $\mathbf{I}(t_0) = \mathbf{I}\big(\mu(t_i), t_i\big)$. Derivando em relação a $\mathbf{p}$ e utilizando novamente a regra da cadeia:

$$\nabla_{\mathbf{p}}\mathbf{I}(t_0) = \nabla_{\mathbf{p}}\mathbf{I}\big(\mu(t_i), t_i\big) = \underset{(N\times2)}{\nabla_{\mathbf{f}}\mathbf{I}} \,.\, \underset{(2\times2)}{\nabla_{\mathbf{p}}\mathbf{f}} \Rightarrow$$

$$\Rightarrow \nabla_{\mathbf{f}}\mathbf{I} = \nabla_{\mathbf{p}}\mathbf{I}(t_0).(\nabla_{\mathbf{p}}\mathbf{f})^{-1}$$

Substituindo em (5.1.4):

$$\boldsymbol{M} = \nabla_{\mathbf{p}}\mathbf{I}(t_0).(\nabla_{\mathbf{p}}\mathbf{f})^{-1}.\nabla_{\boldsymbol{\mu}}\mathbf{f}$$

onde:

$$\nabla_{\mathbf{p}}\mathbf{I}(t_0) = \left[\frac{\partial\mathbf{I}(t_0)}{\partial x_1}, \frac{\partial\mathbf{I}(t_0)}{\partial x_2}\right]$$

$$\nabla_{\mathbf{p}}\mathbf{f} = \left[\frac{\partial\mathbf{f}}{\partial x_1}, \frac{\partial\mathbf{f}}{\partial x_2}\right] = \mu_4 \cdot \begin{bmatrix} \cos\mu_3 & -\operatorname{sen}\mu_3 \\ \operatorname{sen}\mu_3 & \cos\mu_3 \end{bmatrix} \Rightarrow$$

$$\Rightarrow (\nabla_{\mathbf{p}}\mathbf{f})^{-1} = \frac{1}{\mu_4} \cdot \begin{bmatrix} \cos\mu_3 & -\operatorname{sen}\mu_3 \\ \operatorname{sen}\mu_3 & \cos\mu_3 \end{bmatrix}^{-1}$$

$$\nabla_{\boldsymbol{\mu}}\mathbf{f} = \left[\frac{\partial \mathbf{f}}{\partial \mu_1}, \frac{\partial \mathbf{f}}{\partial \mu_2}, \frac{\partial \mathbf{f}}{\partial \mu_3}, \frac{\partial \mathbf{f}}{\partial \mu_4}\right] =$$

$$= \begin{bmatrix} 1 & 0 & -x_1\mu_4 \operatorname{sen}\mu_3 - x_2\mu_4 \cos\mu_3 & x_1 \cos\mu_3 - x_2 \operatorname{sen}\mu_3 \\ 0 & 1 & x_1\mu_4 \cos\mu_3 - x_2\mu_4 \operatorname{sen}\mu_3 & x_1 \operatorname{sen}\mu_3 + x_2 \cos\mu_3 \end{bmatrix} =$$

$$= \begin{bmatrix} & & & \\ \mathbf{I}_2 & & \mu_4 \cdot \begin{bmatrix} \cos\mu_3 & -\operatorname{sen}\mu_3 \\ \operatorname{sen}\mu_3 & \cos\mu_3 \end{bmatrix} \cdot \begin{bmatrix} -x_2 \\ x_1 \end{bmatrix} & \begin{bmatrix} \cos\mu_3 & -\operatorname{sen}\mu_3 \\ \operatorname{sen}\mu_3 & \cos\mu_3 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ & & & \end{bmatrix}$$

Substituindo:

$$\boldsymbol{M} = \begin{bmatrix} \dfrac{\partial \mathbf{I}(t_0)}{\partial x_1} & \dfrac{\partial \mathbf{I}(t_0)}{\partial x_2} \end{bmatrix} \cdot \frac{1}{\mu_4} \cdot \begin{bmatrix} \cos\mu_3 & \operatorname{sen}\mu_3 & -\mu_4 x_2 & x_1 \\ -\operatorname{sen}\mu_3 & \cos\mu_3 & \mu_4 x_1 & x_2 \end{bmatrix} \Rightarrow$$

$$\Rightarrow \boldsymbol{M} = \begin{bmatrix} \dfrac{\partial \mathbf{I}(t_0)}{\partial x_1} & \dfrac{\partial \mathbf{I}(t_0)}{\partial x_2} \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_2 & x_1 \\ 0 & 1 & x_1 & x_2 \end{bmatrix} \cdot \frac{1}{\mu_4} \cdot \begin{bmatrix} \cos\mu_3 & \operatorname{sen}\mu_3 & 0 & 0 \\ -\operatorname{sen}\mu_3 & \cos\mu_3 & 0 & 0 \\ 0 & 0 & \mu_4 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Considerando:

$$\boldsymbol{M_0} = \begin{bmatrix} \dfrac{\partial \mathbf{I}(t_0)}{\partial x_1} & \dfrac{\partial \mathbf{I}(t_0)}{\partial x_2} \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_2 & x_1 \\ 0 & 1 & x_1 & x_2 \end{bmatrix} \Rightarrow$$

$$\Rightarrow \boldsymbol{M_0} = \begin{bmatrix} \dfrac{\partial \mathbf{I}(t_0)}{\partial x_1} & \dfrac{\partial \mathbf{I}(t_0)}{\partial x_2} & -\dfrac{\partial \mathbf{I}(t_0)}{\partial x_1} x_2 + \dfrac{\partial \mathbf{I}(t_0)}{\partial x_2} x_1 & \dfrac{\partial \mathbf{I}(t_0)}{\partial x_1} x_1 + \dfrac{\partial \mathbf{I}(t_0)}{\partial x_2} x_2 \end{bmatrix} \tag{5.1.5}$$

e

$$\boldsymbol{\Sigma} = \frac{1}{\mu_4} \cdot \begin{bmatrix} \cos\mu_3 & \operatorname{sen}\mu_3 & 0 & 0 \\ -\operatorname{sen}\mu_3 & \cos\mu_3 & 0 & 0 \\ 0 & 0 & \mu_4 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Substituindo:

$$\boldsymbol{M} = \boldsymbol{M_0}.\boldsymbol{\Sigma} \tag{5.1.6}$$

O intuito da realização do desenvolvimento anterior foi obter a matriz $\boldsymbol{M_0}$, que, dado um padrão de referência, possui valor constante e pode ser calculada antes do início efetivo do algoritmo. Os valores das derivadas parciais em relação a x_1 e x_2 podem ser obtidos utilizando as máscaras de convolução de Sobel (YOUNG; GERBRANDS; VLIET, 1997), que funcionam

como o método de diferença central de aproximação discreta de derivadas.

$$\frac{\partial I(\mathbf{p})}{\partial x_1} = \frac{\partial I(x_1, x_2)}{\partial x_1} = \frac{1}{8}[I(x_1{+}1, x_2{+}1) - I(x_1{-}1, x_2{+}1) + 2I(x_1{+}1, x_2){-} \\ - 2I(x_1{-}1, x_2) + I(x_1{+}1, x_2{-}1) - I(x_1{-}1, x_2{-}1)] \quad (5.1.7)$$

$$\frac{\partial I(\mathbf{p})}{\partial x_2} = \frac{\partial I(x_1, x_2)}{\partial x_2} = \frac{1}{8}[I(x_1{+}1, x_2{+}1) - I(x_1{+}1, x_2{-}1) + 2I(x_1, x_2{+}1){-} \\ - 2I(x_1, x_2{-}1) + I(x_1{-}1, x_2{+}1) - I(x_1{-}1, x_2{-}1)] \quad (5.1.8)$$

Considerando as seguintes propriedades de matrizes:

$(\boldsymbol{M_1 M_2})^{-1} = \boldsymbol{M_2}^{-1}\boldsymbol{M_1}^{-1}$ (se $\boldsymbol{M_1}$ e $\boldsymbol{M_2}$ são quadradas e de mesma ordem)

$(\boldsymbol{M_1 M_2})^{T} = \boldsymbol{M_2}^{T}\boldsymbol{M_1}^{T}$

Utilizando (5.1.6), calcula-se o termo $(\boldsymbol{M}^T\boldsymbol{M})^{-1}\boldsymbol{M}^T$:

$$(\boldsymbol{M}^T\boldsymbol{M})^{-1}\boldsymbol{M}^T = \big((\boldsymbol{M_0\Sigma})^T(\boldsymbol{M_0\Sigma})\big)^{-1}\big(\boldsymbol{M_0\Sigma}\big)^T = \\ = \boldsymbol{\Sigma}^{-1}\big((\boldsymbol{M_0\Sigma})^T\boldsymbol{M_0}\big)^{-1}(\boldsymbol{M_0\Sigma})^T = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\Sigma}^T\boldsymbol{M_0}^T\boldsymbol{M_0})^{-1}\boldsymbol{\Sigma}^T\boldsymbol{M_0}^T = \\ = \boldsymbol{\Sigma}^{-1}(\boldsymbol{M_0}^T\boldsymbol{M_0})^{-1}(\boldsymbol{\Sigma}^T)^{-1}\boldsymbol{\Sigma}^T\boldsymbol{M_0}^T = \boldsymbol{\Sigma}^{-1}(\boldsymbol{M_0}^T\boldsymbol{M_0})^{-1}\boldsymbol{M_0}^T$$

Substituindo em (5.1.3):

$$\mu(t_i) = \mu(t_{i-1}) - \boldsymbol{\Sigma}^{-1}(\boldsymbol{M_0}^T\boldsymbol{M_0})^{-1}\boldsymbol{M_0}^T\big[\mathbf{I}\big(\mu(t_{i-1}), t_i\big) - \mathbf{I}(t_0)\big]$$

sendo:

$$\boldsymbol{\Sigma}^{-1} = \begin{bmatrix} \mu_4\cos\mu_3 & -\mu_4\,\mathrm{sen}\,\mu_3 & 0 & 0 \\ \mu_4\,\mathrm{sen}\,\mu_3 & \mu_4\cos\mu_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mu_4 \end{bmatrix}$$

Assim, os cálculos realizados durante cada instante de tempo podem ser simplificados, através do cálculo "offline" do termo $(\boldsymbol{M_0}^T\boldsymbol{M_0})^{-1}\boldsymbol{M_0}^T$.

Portanto, o vetor de parâmetros pode ser determinado para todos os instantes de tempo. Além disso, o cálculo de $\mu(t_i)$ é realizado com base em seu valor no instante anterior $\big(\mu(t_{i-1})\big)$, devido à abordagem incremental do algoritmo. Assim, os parâmetros são calculados a partir do movimento do alvo em relação ao seu posicionamento na imagem anterior, e não em relação à sua configuração inicial. Dessa maneira, os deslocamentos considerados do alvo tendem a ser

menores, o que garante um rastreamento mais eficiente.

O movimento do alvo na imagem, além do deslocamento do objeto rastreado em si, é dado também pela movimentação da câmera, definida pelo controlador do sistema. Esta última parcela (que afeta a posição do alvo segundo os eixos x_1 e x_2), por sua vez, é conhecida $\big($dada por $\boldsymbol{B}\mathbf{u}(t_i) = [T.u_1(t_i), T.u_2(t_i)]^T\big)$. Assim, o deslocamento do alvo devido à rotação da câmera pode ser compensado, ficando a cargo do algoritmo somente o rastreamento do movimento do objeto rastreado. A Figura 5.2 mostra um esquema com os passos deste algoritmo.

"Offline":

- Armazenar $\mathbf{I}(t_0)$
- Calcular $\nabla_{\mathbf{p}}\mathbf{I}(t_0)$ utilizando as equações (5.1.7) e (5.1.8)
- Calcular $\boldsymbol{M_0}$ utilizando a equação (5.1.5)
- Armazenar $\boldsymbol{\Lambda} = (\boldsymbol{M_0}^T\boldsymbol{M_0})^{-1}\boldsymbol{M_0}^T$
- Inicializar $\mu(t_0) = [\mu_1(t_0), \mu_2(t_0), \mu_3(t_0), \mu_4(t_0)]^T = [0, 0, 0, 1]^T$
- Inicializar $u_1(t_0) = u_2(t_0) = 0$

"Online":

PARA $i = 1, 2, 3 \ldots$

$$\mu(t_{i-1}) = [\ \mu_1(t_{i-1}) + T.u_1(t_{i-1})\ ,\ \mu_2(t_{i-1}) + T.u_2(t_{i-1})\ ,\ \mu_3(t_{i-1})\ ,\ \mu_4(t_{i-1})\ ]^T$$

$$\boldsymbol{\Sigma}^{-1} = \begin{bmatrix} \mu_4(t_{i-1})\cos[\mu_3(t_{i-1})] & -\mu_4(t_{i-1})\,\mathrm{sen}[\mu_3(t_{i-1})] & 0 & 0 \\ \mu_4(t_{i-1})\,\mathrm{sen}[\mu_3(t_{i-1})] & \mu_4(t_{i-1})\cos[\mu_3(t_{i-1})] & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mu_4(t_{i-1}) \end{bmatrix}$$

$$\mu(t_i) = \mu(t_{i-1}) - \boldsymbol{\Sigma}^{-1}\boldsymbol{\Lambda}[\mathbf{I}(\mu(t_{i-1}), t_i) - \mathbf{I}(t_0)]$$

$\star\star\star\star\star$

Algoritmo de controle $\big($cálculo de $u_1(t_i)$ e $u_2(t_i)\big)$

$\star\star\star\star\star$

FIM

Figura 5.2: Esquema do algoritmo baseado na minimização do erro quadrático.

Segundo os próprios autores (HAGER; BELHUMEUR, 1998), uma grande limitação do algoritmo é o fato de que seu funcionamento requer que haja apenas pequenas mudanças entre cada imagem recebida e a anterior. O truncamento da expansão em série de Taylor de $\mathbf{I}$ até os termos de primeira ordem, a hipótese realizada de que a estimativa obtida de μ é sempre exata e outras aproximações adotadas fazem com que a velocidade com que alvo pode se deslocar seja bastante limitada (ver o Capítulo 8). Assim, os próximos passos são agregar a este algoritmo base outras técnicas que permitam um ganho de desempenho nesse quesito.

5.2 Extensão do Algoritmo Através de Estimação Incremental

O algoritmo anterior pode ser estendido através da aplicação de um passo de estimação incremental, como o utilizado por Odobez e Bouthemy (1995). É sabido que o algoritmo base produz resultados precisos somente para pequenos deslocamentos do alvo. Por outro lado, se for considerado que o valor de $\delta\mu$, apesar de não fornecer como solução o valor exato de μ, produz uma estimativa relativamente **mais próxima** da exata, é razoável admitir que, se o algoritmo for aplicado novamente, a estimativa obtida é ainda mais acurada. Assim, com soluções cada vez mais próximas da exata, a posição do alvo em relação às estimativas se aproxima de zero. Em outras palavras, o deslocamento relativo do alvo se torna cada vez menor, o que favorece o desempenho do algoritmo.

Esta extensão do algoritmo de rastreamento consiste, então, em aplicar iterativamente o algoritmo anterior (atualizando a estimativa do vetor de parâmetros μ a cada iteração) sobre uma mesma imagem adquirida até que o valor de μ se estabilize em torno de um valor constante.

Para garantir que uma solução seja atingida e que o algoritmo não seja aplicado indefinidamente sobre uma mesma imagem, é necessário estabelecer alguns critérios de parada. O primeiro deles consiste na definição de um valor mínimo para $\delta\mu$. Se o valor calculado (em módulo) se torna inferior a este, significa que as alterações sobre o vetor de parâmetros são desprezíveis e correspondem a pequenas oscilações em torno da posição de equilíbrio. Entretanto, segundo testes realizados, o estabelecimento de um valor mínimo constante a partir do qual o vetor começa a oscilar não é possível com grande segurança, pois depende consideravelmente do padrão de referência adotado. Isso pode ser justificado pelo fato de o cálculo de $\delta\mu$ depender dos valores de gradiente da imagem, que variam de acordo com o padrão considerado.

Assim, a solução encontrada é utilizar em conjunto um critério que consiste na análise do sinal de $\delta\mu$. Através de testes, é percebido que as componentes do vetor de parâmetros atualizadas a cada iteração apresentam um comportamento monotônico. Dessa maneira, seus valores somente crescem ou decrescem antes de atingirem seus valores de equilíbrio (e passarem a oscilar). Assim, as componentes de $\delta\mu$ apresentam sinais constantes, ou seja, cada componente é estritamente positiva ou negativa. Após a estabilização das estimativas, μ passa a oscilar em torno do seu valor de equilíbrio, o que é refletido através da mudança de sinal de $\delta\mu$. Portanto, este critério consiste na iteração do algoritmo até que ocorra pelo menos uma mudança de sinal em cada componente de $\delta\mu$, caracterizando a estabilização das estimativas.

Por segurança, se nenhum dos critérios anteriores forem satisfeitos, é definido ainda um

número máximo de iterações a que o algoritmo pode estar sujeito, para que não seja aplicado infinitamente. Um esquema de funcionamento deste algoritmo pode ser encontrado na Figura 5.3.

Aplicar as inicializações "offline" do primeiro algoritmo
Definir $n\_iteracoes^{*}$, $erro_1$, $erro_2$, $erro_3$ e $erro_4{}^{\dagger}$

PARA $i = 1, 2, 3 \ldots$

$\mu(t_i) = [\ \mu_1(t_{i-1}) + T.u_1(t_{i-1})\ ,\ \mu_2(t_{i-1}) + T.u_2(t_{i-1})\ ,\ \mu_3(t_{i-1})\ ,\ \mu_4(t_{i-1})\ ]^T$

$ts_1 = ts_2 = ts_3 = ts_4{}^{\ddagger} = iteracao^{**} = 0$

FAZER

$$\boldsymbol{\Sigma}^{-1} = \begin{bmatrix} \mu_4(t_i)\cos[\mu_3(t_i)] & -\mu_4(t_i)\,\mathrm{sen}[\mu_3(t_i)] & 0 & 0 \\ \mu_4(t_i)\,\mathrm{sen}[\mu_3(t_i)] & \mu_4(t_i)\cos[\mu_3(t_i)] & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mu_4(t_i) \end{bmatrix}$$

$\delta\mu = -\boldsymbol{\Sigma}^{-1}\boldsymbol{\Lambda}\big[\mathbf{I}\big(\mu(t_i), t_i\big) - \mathbf{I}(t_0)\big]$

$\mu(t_i) = \mu(t_i) + \delta\mu$

SE ($iteracao = 0$) ENTÃO
$sin_1{}^{\dagger\dagger}$ = sinal$^{\ddagger\ddagger}(\delta\mu_1)$ sin_2 = sinal$(\delta\mu_2)$
sin_3 = sinal$(\delta\mu_3)$ sin_4 = sinal$(\delta\mu_4)$

FIM

$iteracao = iteracao + 1$

SE $\big(sin_1 \neq$ sinal$(\delta\mu_1)\big)$ ENTÃO $ts_1 = 1$ FIM
SE $\big(sin_2 \neq$ sinal$(\delta\mu_2)\big)$ ENTÃO $ts_2 = 1$ FIM
SE $\big(sin_3 \neq$ sinal$(\delta\mu_3)\big)$ ENTÃO $ts_3 = 1$ FIM
SE $\big(sin_4 \neq$ sinal$(\delta\mu_4)\big)$ ENTÃO $ts_4 = 1$ FIM

ENQUANTO

$$\left|\begin{array}{l} \big(iteracao < n\_iteracoes\big) \quad \mathrm{E} \\ \big(ts_1 = 0\ \ \mathrm{OU}\ \ ts_2 = 0\ \ \mathrm{OU}\ \ ts_3 = 0\ \ \mathrm{OU}\ \ ts_4 = 0\big) \quad \mathrm{E} \\ \big(|\delta\mu_1| \geq erro_1\ \ \mathrm{OU}\ \ |\delta\mu_2| \geq erro_2\ \ \mathrm{OU}\ \ |\delta\mu_3| \geq erro_3\ \ \mathrm{OU}\ \ |\delta\mu_4| \geq erro_4\big) \end{array}\right.$$

$\star\star\star\star\star$
Algoritmo de controle $\big($cálculo de $u_1(t_i)$ e $u_2(t_i)\big)$
$\star\star\star\star\star$

FIM

$^{*}n\_iteracoes$ = número máximo de iterações permitido
$^{\dagger}erro_j$ = valor mínimo de erro relativo permitido para μ_j
$^{\ddagger}ts_j$ = indica se houve troca de sinal no valor de $\delta\mu_j$
$^{**}iteracao$ = número de iterações já realizadas
$^{\dagger\dagger}sin_j$ = sinal de $\delta\mu_j$ obtido na primeira iteração
$^{\ddagger\ddagger}\mathrm{sinal}(x)$ = sinal (positivo ou negativo) de x

Figura 5.3: Esquema do algoritmo estendido através de estimação incremental.

5.3 Extensão do Algoritmo por Meio de Estimação em Multiresolução

Este procedimento também foi extraído do trabalho de Odobez e Bouthemy (1995). A incorporação da técnica de estimação incremental sobre o algoritmo base permite o rastreamento de um alvo localizado a maiores distâncias em relação à última posição conhecida. Entretanto, se o padrão se deslocar para uma posição externa à janela considerada, o algoritmo se torna impraticável. Assim, a técnica de estimação em multiresolução tem como objetivo aumentar ainda mais o alcance do algoritmo em relação aos deslocamentos de translação do alvo. Inclusive para alguns valores que ultrapassam o tamanho correspondente da região-alvo.

Os algoritmos anteriores funcionam a partir de comparações entre uma imagem recebida e uma imagem armazenada (padrão de referência). O princípio de multiresolução, por sua vez, consiste em utilizar várias imagens a cada iteração, obtidas a partir da conversão de uma imagem recebida para outras de menores resoluções. A vantagem disso é que, se o tamanho considerado da região-alvo em todas as imagens (de diferentes resoluções) é invariável, quanto menor a resolução de uma imagem, maior é o tamanho **relativo** da região-alvo em seu interior (ver a Figura 5.4).

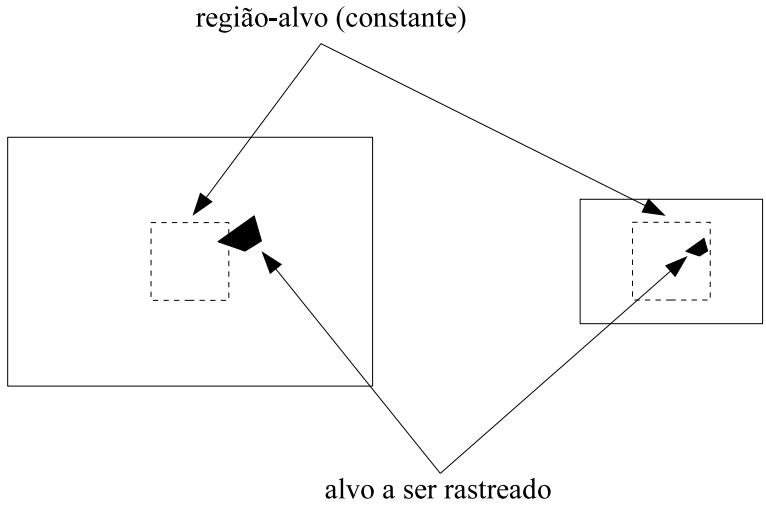

Figura 5.4: Exemplo ilustrativo do princípio de estimação em multiresolução.

Assim, se, por exemplo, a resolução de uma imagem for reduzida à metade, uma posição x do alvo relativa à imagem original se torna $\frac{x}{2}$ na reduzida. Isso significa que, a grosso modo, o algoritmo passa a ser capaz de rastrear um alvo situado até duas vezes mais distante

que originalmente. A desvantagem desse procedimento é que uma redução da resolução da imagem implica em um número menor de pixels representando o alvo. Dessa maneira, ocorre uma diminuição da quantidade de informação presente, o que pode comprometer a precisão do algoritmo.

Portanto, dado um certo número de imagens com diferentes resoluções, a estimação em multiresolução consiste em aplicar o algoritmo anterior (incremental), inicialmente, sobre a imagem de menor resolução, obtendo-se o vetor de parâmetros μ. Em seguida, a partir destes parâmetros, o algoritmo é novamente aplicado, mas sobre a imagem de segunda menor resolução, e assim sucessivamente, até a imagem original. Dessa maneira, a estimação em menores resoluções (menos detalhes) permite ao algoritmo determinar valores do vetor de parâmetros para um alvo relativamente distante. Conforme a resolução da imagem analisada (e o nível de detalhes) aumenta e os deslocamentos relativos do alvo diminuem (valores de μ mais próximos da solução exata), o algoritmo tende a produzir resultados mais próximos dos valores reais dos parâmetros.

Neste trabalho, são utilizados dois níveis de resolução: original (320×240) e 1:2 (160×120). Deve-se ressaltar o fato de que os valores de μ_1 e μ_2 do vetor de parâmetros, ao se passar da resolução inferior para a original, devem ser multiplicados por 2. Analogamente, ao se passar da resolução original para a inferior, devem ser divididos por 2. Já os parâmetros de rotação (μ_3) e escala (μ_4) não sofrem influência direta da variação de resolução em uma imagem.

Para reduzir a resolução de uma imagem à metade, o procedimento adotado consiste em uma simples média entre blocos de 4 pixels (2×2).

$$I^k(\mathbf{p}) = I^k(x_1, x_2) = \frac{1}{4}[I^{k+1}(2x_1-1, 2x_2-1) + I^{k+1}(2x_1-1, 2x_2)+ \\ + I^{k+1}(2x_1, 2x_2-1) + I^{k+1}(2x_1, 2x_2)] \quad (5.3.1)$$

onde I^k corresponde ao valor de um pixel da imagem com resolução duas vezes menor que a de uma imagem representada por pixels de valores dados por I^{k+1}.

Um esquema deste algoritmo pode ser encontrado na Figura 5.5.

Portanto, a partir de um dos algoritmos anteriores, obtêm-se os valores de μ_1 e μ_2, que correspondem, respectivamente, às posições x_1 e x_2 do alvo rastreado em uma imagem recebida. Esses valores compõem o vetor de medidas $\mathbf{y}$, que deve ser fornecido ao controlador para determinar o posicionamento pan-tilt da câmera e completar a realimentação do sistema.

Inicializações “offline” do algoritmo anterior para as duas resoluções (os sobrescritos “1” e “2” referem-se, respectivamente, às resoluções 1:2 e original):

- Armazenar $\mathbf{I}^1(t_0)$ e $\mathbf{I}^2(t_0)$, utilizando a equação (5.3.1)
- Calcular e armazenar as matrizes $\boldsymbol{\Lambda}^k$, a partir de $\mathbf{I}^k(t_0)$, $k = 1, 2$
- Inicializar $\boldsymbol{\mu}(t_0) = [0, 0, 0, 1]^T$ e $u_1(t_0) = u_2(t_0) = 0$
- Definir $n\_iteracoes$, $erro_1$, $erro_2$, $erro_3$ e $erro_4$

PARA $i = 1, 2, 3 \ldots$

$$\boldsymbol{\mu}(t_i) = \left[\frac{\mu_1(t_{i-1}) + T.u_1(t_{i-1})}{2} \ , \ \frac{\mu_2(t_{i-1}) + T.u_2(t_{i-1})}{2} \ , \ \mu_3(t_{i-1}) \ , \ \mu_4(t_{i-1}) \right]^T$$

PARA $k = 1$ e 2

$ts_1 = ts_2 = ts_3 = ts_4 = iteracao = 0$

FAZER

$$\boldsymbol{\Sigma}^{-1} = \begin{bmatrix} \mu_4(t_i)\cos[\mu_3(t_i)] & -\mu_4(t_i)\,\mathrm{sen}[\mu_3(t_i)] & 0 & 0 \\ \mu_4(t_i)\,\mathrm{sen}[\mu_3(t_i)] & \mu_4(t_i)\cos[\mu_3(t_i)] & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mu_4(t_i) \end{bmatrix}$$

$\delta\boldsymbol{\mu} = -\boldsymbol{\Sigma}^{-1}\boldsymbol{\Lambda}^k\left[\mathbf{I}^k\big(\boldsymbol{\mu}(t_i), t_i\big) - \mathbf{I}^k(t_0)\right]$

$\boldsymbol{\mu}(t_i) = \boldsymbol{\mu}(t_i) + \delta\boldsymbol{\mu}$

SE ($iteracao = 0$) ENTÃO

sin_1 = sinal($\delta\mu_1$) sin_2 = sinal($\delta\mu_2$)

sin_3 = sinal($\delta\mu_3$) sin_4 = sinal($\delta\mu_4$)

FIM

$iteracao = iteracao + 1$

SE $\big(sin_1 \neq$ sinal($\delta\mu_1$)$\big)$ ENTÃO $ts_1 = 1$ FIM

SE $\big(sin_2 \neq$ sinal($\delta\mu_2$)$\big)$ ENTÃO $ts_2 = 1$ FIM

SE $\big(sin_3 \neq$ sinal($\delta\mu_3$)$\big)$ ENTÃO $ts_3 = 1$ FIM

SE $\big(sin_4 \neq$ sinal($\delta\mu_4$)$\big)$ ENTÃO $ts_4 = 1$ FIM

ENQUANTO

$$\left| \begin{array}{l} \big(iteracao < n\_iteracoes\big) \quad \text{E} \\ \big(ts_1 = 0 \ \text{ OU } \ ts_2 = 0 \ \text{ OU } \ ts_3 = 0 \ \text{ OU } \ ts_4 = 0\big) \quad \text{E} \\ \big(|\delta\mu_1| \geq erro_1 \ \text{ OU } \ |\delta\mu_2| \geq erro_2 \ \text{ OU } \ |\delta\mu_3| \geq erro_3 \ \text{ OU } \ |\delta\mu_4| \geq erro_4\big) \end{array} \right.$$

SE $k = 1$ ENTÃO

$\mu_1(t_i) = 2\mu_1(t_i)$

$\mu_2(t_i) = 2\mu_2(t_i)$

FIM

FIM

$\star\star\star\star\star$

Algoritmo de controle $\big($cálculo de $u_1(t_i)$ e $u_2(t_i)\big)$

$\star\star\star\star\star$

FIM

Figura 5.5: Esquema do algoritmo estendido por estimação em multiresolução.

6 PROJETO DO CONTROLADOR

Dado o modelo do sistema $\big($equações (4.1.2) e (4.1.3)$\big)$, a função do controlador é calcular os sinais de controle (vetor de controle $\mathbf{u}$) a partir das posições x_1 e x_2 do alvo obtidas pelo filtro (vetor de medidas $\mathbf{y}$). As variáveis de controle devem possibilitar a anulação do vetor de estados $\mathbf{x}$, ou seja, tornar as posições x_1 e x_2 do alvo iguais a zero, trazendo-o para o centro da imagem. Além disso, antes de aplicar os sinais de controle no sistema, é necessário converter o valor de $\mathbf{u}$ para velocidades de rotação pan e tilt, através do jacobiano de imagem dado pelas equações (4.1.9) e (4.1.8).

A seguir, é mostrada uma breve descrição de regulador linear quadrático (LQ). Ele consiste na base para os dois controladores desenvolvidos neste trabalho para estabilizar o sistema, um proporcional integral (PI) e um proporcional com estimação das perturbações externas (LQG). Neste último, as perturbações são estimadas através de um filtro de Kalman, de modo que uma breve descrição desta técnica é dada após a do regulador LQ. Os controladores PI e LQG são apresentados logo após.

6.1 Controle Linear Quadrático

O cálculo dos controladores para o sistema é baseado em um critério ótimo no sentido de minimizar uma função quadrática de custo, aplicado a sistemas lineares. Esse tipo de controlador é chamado de linear quadrático (LQ) (MAYBECK, 1979a).

Considera-se inicialmente um modelo linear discreto dado pelas seguintes equações:

$$\mathbf{x}(t_{i+1}) = \boldsymbol{A}\mathbf{x}(t_i) + \boldsymbol{B}\mathbf{u}(t_i) + \boldsymbol{G}\mathbf{w}(t_i)$$
$$\mathbf{y}(t_i) = \boldsymbol{C}\mathbf{x}(t_i) + \mathbf{v}(t_i)$$

onde:

$\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{G}$ = matrizes conhecidas

$\mathbf{x}(t_i)$ = vetor de estados no instante t_i

$\mathbf{y}(t_i)$ = vetor de medidas no instante t_i

$\mathbf{u}(t_i)$ = vetor das variáveis de controle no instante t_i

$\mathbf{w}(t_i)$ = vetor de incertezas sobre o modelo no instante t_i

$\mathbf{v}(t_i)$ = vetor de ruídos nos sensores no instante t_i

Ignoram-se por enquanto os termos de incertezas ($\boldsymbol{G}$, $\mathbf{w}$ e $\mathbf{v}$). Um controlador **regulador LQ** tem como objetivo fazer com que as variáveis de estado $\mathbf{x}(t_i)$ assumam valores o mais próximo possível de zero. Assim, se $\mathbf{x}(t_0)$ possui um valor não nulo, é necessário determinar convenientemente os valores das variáveis de controle $\mathbf{u}(t_i)$, de modo que, após um certo período de tempo, $\mathbf{x}(t_i)$ e $\mathbf{u}(t_i)$ convirjam para valores próximos de zero.

O critério linear quadrático para o cálculo do vetor de controle consiste na minimização da seguinte função de custo J:

$$J = \frac{1}{2}\mathbf{x}^T(t_{n+1})\boldsymbol{X}_f\mathbf{x}(t_{n+1}) + \sum_{i=0}^{n}\frac{1}{2}\left(\begin{bmatrix}\mathbf{x}(t_i)\\ \mathbf{u}(t_i)\end{bmatrix}^T\begin{bmatrix}\boldsymbol{X} & \boldsymbol{S}\\ \boldsymbol{S}^T & \boldsymbol{U}\end{bmatrix}\begin{bmatrix}\mathbf{x}(t_i)\\ \mathbf{u}(t_i)\end{bmatrix}\right)$$

onde:

$\mathbf{x}(t_{n+1})$ = valor (conhecido) de $\mathbf{x}$ no instante de tempo final de interesse

$\boldsymbol{X_f}, \boldsymbol{X}, \boldsymbol{S} \geq 0$ e $\boldsymbol{U} > 0$ = matrizes de peso de cada termo da função de custo

Considerando-se o caso particular de apenas uma variável de estado, a função se torna:

$$J = \frac{1}{2}\boldsymbol{X}_f.x^2(t_{n+1}) + \sum_{i=0}^{n}\frac{1}{2}(\boldsymbol{X}.x^2(t_i) + \boldsymbol{U}.u^2(t_i) + 2\boldsymbol{S}.x(t_i)u(t_i))$$

Pode-se observar que a função de custo é a soma dos valores quadráticos de x e u, sendo que os valores de peso determinam a taxa com que cada termo é minimizado em relação aos outros.

De acordo com Maybeck (1979a), a função de controle que minimiza a função de custo é dada por:

$$\mathbf{u}(t_i) = -\boldsymbol{K_c}(t_i)\mathbf{x}(t_i) \tag{6.1.1}$$

onde a matriz de ganhos $\boldsymbol{K_c}$ é dada por:

$$\boldsymbol{K_c}(t_i) = [\boldsymbol{U} + \boldsymbol{B}^T\boldsymbol{X_c}(t_{i+1})\boldsymbol{B}]^{-1}[\boldsymbol{B}^T\boldsymbol{X_c}(t_{i+1})\boldsymbol{A} + \boldsymbol{S}^T] \tag{6.1.2}$$

sendo que $\boldsymbol{X_c}$ é solução da seguinte equação recursiva de Riccati:

$$\boldsymbol{X_c}(t_i) = \boldsymbol{X} + \boldsymbol{A}^T\boldsymbol{X_c}(t_{i+1})\boldsymbol{A} - [\boldsymbol{B}^T\boldsymbol{X_c}(t_{i+1})\boldsymbol{A} + \boldsymbol{S}^T]^T\boldsymbol{K_c}(t_i) \tag{6.1.3}$$

resolvida **regressivamente** (ou seja, do instante final de interesse até o instante inicial) a partir da condição **final** $\boldsymbol{X_c}(t_{n+1}) = \boldsymbol{X_f}$.

Com as equações acima, podem ser calculados os valores de $\boldsymbol{K_c}$ para todos os n instantes de tempo, permitindo ao modelo se estabilizar em torno de zero após o intervalo considerado.

Uma aproximação empregada neste trabalho é a utilização de um valor fixo para a matriz $\boldsymbol{K_c}$. A equação de Riccati, ao ser calculada para instantes regressivos de tempo, tende a se estabilizar em torno de um valor constante. Utilizar o valor de $\boldsymbol{K_c}$ correspondente ao valor $\boldsymbol{X_c}$ de estabilização corresponde à técnica chamada de "problema de horizonte infinito", em que não há um tempo definido para a estabilização do modelo ($n \rightarrow \infty$). Uma outra maneira de enxergar esta solução é como a determinação do valor constante de $\boldsymbol{X_c}$ que satisfaz a equação de Riccati (6.1.3). Assim, se a equação não é iterada, ocorre também que a matriz de peso $\boldsymbol{X_f}$ não possui mais significado.

O controlador regulador LQ não pode ser aplicado diretamente no modelo da câmera pan-tilt, já que não leva em consideração perturbações externas (movimento do alvo) como entrada. Em compensação, ele serve de base para outros algoritmos de controle mais elaborados, como os desenvolvidos neste trabalho.

6.2 Filtro de Kalman

O filtro de Kalman é uma técnica utilizada para a estimação das variáveis de estado ($\mathbf{x}$) de um sistema a partir das medidas ($\mathbf{y}$) adquiridas por meio de seus sensores. Diferentemente do observador de estados usual (FLEURY, 1981/1982) utilizado em sistemas de controle, o filtro de Kalman busca estimar parâmetros de um sistema através de uma abordagem estocástica. Quando o sistema apresenta ruídos nos sensores e/ou incertezas na modelagem, o filtro busca minimizar os erros decorrentes através do conhecimento da estatística dos ruídos e das incertezas. No controlador LQG deste trabalho, ele é utilizado como forma de estimar os valores das perturbações externas (movimento do alvo).

A apresentação do filtro de Kalman a seguir é realizada através de uma abordagem bastante simplificada. Detalhes mais completos e descritos de maneira mais rigorosa, além de definições formais de termos utilizados, são dados por Maybeck (1979b) ou Fleury (1981/1982). Inicialmente, considera-se novamente o seguinte sistema de equações de diferenças:

$$\mathbf{x}(t_{i+1}) = \boldsymbol{A}\mathbf{x}(t_i) + \boldsymbol{B}\mathbf{u}(t_i) + \boldsymbol{G}\mathbf{w}(t_i)$$
$$\mathbf{y}(t_i) = \boldsymbol{C}\mathbf{x}(t_i) + \mathbf{v}(t_i)$$

Assume-se que o valor inicial do vetor de estados $\big(\mathbf{x}(t_0)\big)$ seja uma variável aleatória gaussiana $\Big($totalmente caracterizada através de sua média $\mathrm{E}[\mathbf{x}(t_0)]$ e sua matriz de covariâncias

$\mathrm{E}\big[\big(\mathbf{x}(t_0)-\mathrm{E}[\mathbf{x}(t_0)]\big)\big(\mathbf{x}(t_0)-\mathrm{E}[\mathbf{x}(t_0)]\big)^T\big]\big)$ e os vetores $\mathbf{w}(t_i)$ e $\mathbf{v}(t_i)$ ruídos brancos, todos com estatística conhecida.

$$\mathrm{E}[\mathbf{x}(t_0)] = \hat{\mathbf{x}}_\mathbf{0}$$
$$\mathrm{E}\big[\big(\mathbf{x}(t_0) - \hat{\mathbf{x}}_\mathbf{0}\big)\big(\mathbf{x}(t_0) - \hat{\mathbf{x}}_\mathbf{0}\big)^T\big] = \boldsymbol{P_0}$$

$$\mathrm{E}[\mathbf{w}(t_i)] = \mathbf{0} \qquad\qquad \mathrm{E}[\mathbf{v}(t_i)] = \mathbf{0}$$
$$\mathrm{E}[\mathbf{w}(t_i)\mathbf{w}^T(t_j)] = \begin{cases} \boldsymbol{Q}, & t_i = t_j \\ \mathbf{0}, & t_i \neq t_j \end{cases} \qquad \mathrm{E}[\mathbf{v}(t_i)\mathbf{v}^T(t_j)] = \begin{cases} \boldsymbol{R}, & t_i = t_j \\ \mathbf{0}, & t_i \neq t_j \end{cases}$$

onde:

$\hat{\mathbf{x}}_\mathbf{0}$, $\boldsymbol{P_0}$, $\boldsymbol{Q}$, $\boldsymbol{R}$ = matrizes com valores conhecidos

Assumindo também que $\mathbf{x}(t_0)$, $\mathbf{w}(t_i)$ e $\mathbf{v}(t_i)$ são mutuamente independentes e a dinâmica do modelo é dada por um processo de Gauss-Markov, a dinâmica do filtro pode ser caracterizada por duas funções recursivas, calculadas a partir de seus valores no instante de tempo anterior. Essas duas funções correspondem à média e à matriz de covariâncias de uma função densidade de probabilidades, sendo que a média corresponde ao valor do vetor de estados estimado e as covariâncias à sua incerteza.

A dinâmica do filtro pode ser dividida em duas partes: um ciclo de propagação e um ciclo de atualização. Considerando que uma nova medida dos sensores está disponível em t_i, o ciclo de propagação consiste em estimar os valores das variáveis de estado no instante t_i a partir dos seus valores no instante anterior (t_{i-1}), utilizando a dinâmica do modelo. Já o ciclo de atualização é responsável por incorporar as novas medidas $\big(\mathbf{y}(t_i)\big)$ de modo a melhorar a estimativa do vetor de estados, já que traz informações do sistema físico (real).

1. Ciclo de propagação:

$$\hat{\mathbf{x}}(t_i^-) = \boldsymbol{A}\hat{\mathbf{x}}(t_{i-1}^+) + \boldsymbol{B}\mathbf{u}(t_{i-1}^+) \tag{6.2.1}$$
$$\boldsymbol{P}(t_i^-) = \boldsymbol{A}\boldsymbol{P}(t_{i-1}^+)\boldsymbol{A}^T + \boldsymbol{G}\boldsymbol{Q}\boldsymbol{G}^T \tag{6.2.2}$$

2. Ciclo de atualização:

$$\boldsymbol{K_f}(t_i) = \boldsymbol{P}(t_i^-)\boldsymbol{C}^T[\boldsymbol{C}\boldsymbol{P}(t_i^-)\boldsymbol{C}^T + \boldsymbol{R}]^{-1} \tag{6.2.3}$$
$$\hat{\mathbf{x}}(t_i^+) = \hat{\mathbf{x}}(t_i^-) + \boldsymbol{K_f}(t_i)[\mathbf{y}(t_i) - \boldsymbol{C}\hat{\mathbf{x}}(t_i^-)] \tag{6.2.4}$$
$$\boldsymbol{P}(t_i^+) = \boldsymbol{P}(t_i^-) - \boldsymbol{K_f}(t_i)\boldsymbol{C}\boldsymbol{P}(t_i^-) \tag{6.2.5}$$

As variáveis $\hat{\mathbf{x}}(t_i)$ e $\boldsymbol{P}(t_i)$ correspondem aos valores de média e matriz de covariâncias de $\mathbf{x}(t_i)$ no instante de tempo t_i, respectivamente. Os símbolos "$-$" e "$+$" sobrescritos em t_i referem-se, respectivamente, aos instantes imediatamente anterior e posterior à chegada de novas medidas. Em relação à matriz $\boldsymbol{K_f}(t_i)$, ela é denominada matriz de ganhos de Kalman. Em (6.2.4) observa-se que ela é responsável por ponderar a influência do vetor de medidas na estimação do vetor de estados. Além disso, como a matriz $\boldsymbol{K_f}$ é calculada a partir de $\boldsymbol{P}$, essa influência das medidas está totalmente relacionada ao nível de conhecimento (incerteza) que se tem sobre o sistema.

Pode-se observar uma semelhança entre a forma do filtro de Kalman e a do regulador LQ, já que ambas apresentam um cálculo recursivo de variáveis ($\boldsymbol{P}$ e $\boldsymbol{X_c}$) e determinação de matrizes de ganhos ($\boldsymbol{K_f}$ e $\boldsymbol{K_c}$). Em relação à matriz de ganhos de Kalman, neste trabalho, ela também é considerada constante e igual ao valor obtido após a estabilização das equações de $\boldsymbol{P}(t_i)$ $\big($(6.2.2) e (6.2.5)$\big)$. Dessa maneira, embora haja um prejuízo na estimação das variáveis de estado, não há necessidade de se calcular e se conhecer o valor de $\boldsymbol{K_f}$ em todos os instantes de tempo. Assim, as equações a serem iteradas são somente as de média:

$$\hat{\mathbf{x}}(t_i^-) = \boldsymbol{A}\hat{\mathbf{x}}(t_{i-1}^+) + \boldsymbol{B}\mathbf{u}(t_{i-1}) \tag{6.2.6}$$

$$\hat{\mathbf{x}}(t_i^+) = \hat{\mathbf{x}}(t_i^-) + \boldsymbol{K_f}[\mathbf{y}(t_i) - \boldsymbol{C}\hat{\mathbf{x}}(t_i^-)] \tag{6.2.7}$$

6.3 Controlador Proporcional Integral

O primeiro controlador projetado para o sistema consiste em um proporcional integral (PI) calculado segundo um critério linear quadrático (LQ). A vantagem de um controlador PI é que ele possui a propriedade de promover o modelo para tipo 1 (MAYBECK, 1979a), ou seja, entradas de perturbações **constantes** não modeladas (de valor desconhecido) adicionadas ao modelo são suprimidas, possibilitando-lhe se estabilizar em torno de seu valor de regime. No caso do modelo deste trabalho, desconsiderando-se as entradas de ruído ($\mathbf{w}$ e $\mathbf{v}$), o valor de regime se anula com esse controlador se o vetor de perturbações externas é constante, ou seja, se o alvo se movimenta com velocidade constante. O controlador PI permite também a estabilização da saída do modelo em torno de um valor não-nulo, mas esta característica não é necessária nesta aplicação.

Maybeck (1979a) propõe dois tipos de algoritmo de controladores PI: um baseado na pseudointegral do erro do regulador e outro baseado na diferença de controle. No modelo deste trabalho, pode ser verificado que o resultado obtido é idêntico nos dois casos. A seguir, encontra-se

retratado somente o controlador baseado na pseudointegral do erro do regulador.

Em controle em espaço de estados, uma ação integral pode ser fornecida criando-se um modelo aumentado através de estados adicionais $\mathbf{q}(t_i)$.

$$\mathbf{q}(t_{i+1}) = \mathbf{q}(t_i) + [\mathbf{y}(t_i) - \mathbf{y_d}]$$

onde:

$\mathbf{y_d}$ = valor em torno do qual deseja-se que o vetor de medidas se estabilize

No caso do modelo deste trabalho, $\mathbf{y_d} = [0, 0]^T$, já que se deseja um equilíbrio em torno do centro da imagem.

$$\mathbf{q}(t_{i+1}) = \mathbf{q}(t_i) + \mathbf{y}(t_i) \tag{6.3.1}$$

De (4.1.2) e (4.1.3), retirando-se os termos relativos às incertezas ($\boldsymbol{G}$, $\mathbf{w}$ e $\mathbf{v}$):

$$\begin{cases} \mathbf{x}(t_{i+1}) = \boldsymbol{A}\mathbf{x}(t_i) + \boldsymbol{B}\mathbf{u}(t_i) + \boldsymbol{E}\mathbf{d}(t_i) \\ \mathbf{y}(t_i) = \boldsymbol{C}\mathbf{x}(t_i) \end{cases}$$

Comparando este modelo com o modelo controlado pelo regulador LQ na Seção 6.1, há um termo adicional ($\boldsymbol{E}\mathbf{d}$), relativo às perturbações externas. Assim, o controlador regulador não pode ser aplicado diretamente, sendo necessário suprimir essa entrada de perturbações.

Utilizando a equação (6.3.1), o modelo aumentado, considerando as perturbações $\mathbf{d}(t_i)$ constantes, torna-se:

$$\begin{bmatrix} \mathbf{x}(t_{i+1}) \\ \mathbf{q}(t_{i+1}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \mathbf{0} \\ \boldsymbol{C} & \mathbf{I_2} \end{bmatrix} \begin{bmatrix} \mathbf{x}(t_i) \\ \mathbf{q}(t_i) \end{bmatrix} + \begin{bmatrix} \boldsymbol{B} \\ \mathbf{0} \end{bmatrix} \mathbf{u}(t_i) + \begin{bmatrix} \boldsymbol{E}\mathbf{d} \\ \mathbf{0} \end{bmatrix} \tag{6.3.2}$$

Admitindo que a solução de equilíbrio (após atingida a estabilização) do modelo é $\mathbf{x}(t_i) = \mathbf{x_0}$, $\mathbf{q}(t_i) = \mathbf{q_0}$ e $\mathbf{u}(t_i) = \mathbf{u_0}$, definem-se as seguintes variáveis de perturbação:

$$\delta\mathbf{x}(t_i) = \mathbf{x}(t_i) - \mathbf{x_0} \tag{6.3.3}$$

$$\delta\mathbf{q}(t_i) = \mathbf{q}(t_i) - \mathbf{q_0} \tag{6.3.4}$$

$$\delta\mathbf{u}(t_i) = \mathbf{u}(t_i) - \mathbf{u_0} \tag{6.3.5}$$

Em regime permanente, o modelo fornecido em (6.3.2) é dado por:

$$\begin{bmatrix} \mathbf{x_0} \\ \mathbf{q_0} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \mathbf{0} \\ \boldsymbol{C} & \mathbf{I_2} \end{bmatrix} \begin{bmatrix} \mathbf{x_0} \\ \mathbf{q_0} \end{bmatrix} + \begin{bmatrix} \boldsymbol{B} \\ \mathbf{0} \end{bmatrix} \mathbf{u_0} + \begin{bmatrix} \boldsymbol{E}\mathbf{d} \\ \mathbf{0} \end{bmatrix}$$

Subtraindo de (6.3.2):

$$\begin{bmatrix} \delta\mathbf{x}(t_{i+1}) \\ \delta\mathbf{q}(t_{i+1}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \mathbf{0} \\ \boldsymbol{C} & \mathbf{I_2} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x}(t_i) \\ \delta\mathbf{q}(t_i) \end{bmatrix} + \begin{bmatrix} \boldsymbol{B} \\ \mathbf{0} \end{bmatrix} \delta\mathbf{u}(t_i) \Rightarrow$$

$$\Rightarrow \mathbf{x_a}(t_{i+1}) = \boldsymbol{A_a}\mathbf{x_a}(t_i) + \boldsymbol{B_a}\mathbf{u_a}(t_i) \tag{6.3.6}$$

Percebe-se que a entrada de perturbações constantes não está mais presente em (6.3.6). Além disso, em regime permanente, deseja-se que o modelo se estabilize na solução de equilíbrio, ou seja, que as variáveis de perturbação $\delta\mathbf{x}(t_i)$, $\delta\mathbf{q}(t_i)$ e $\delta\mathbf{u}(t_i)$ se anulem. Dessa maneira, o controlador regulador pode ser aplicado e, utilizando como vetor de estados o vetor $\mathbf{x_a}$ do modelo aumentado, temos um regulador LQ de perturbações, cuja função quadrática de custo a ser minimizada é dada por:

$$J = \sum_{i=0}^{\infty} \left\{ \frac{1}{2} \left[\begin{array}{c} \delta\mathbf{x}(t_i) \\ \delta\mathbf{q}(t_i) \\ \hline \delta\mathbf{u}(t_i) \end{array}\right]^T \left[\begin{array}{cc|c} \boldsymbol{X_{xx}} & \boldsymbol{X_{xq}} & \boldsymbol{S_{xu}} \\ \boldsymbol{X_{xq}}^T & \boldsymbol{X_{qq}} & \boldsymbol{S_{qu}} \\ \hline \boldsymbol{S_{xu}}^T & \boldsymbol{S_{qu}}^T & \boldsymbol{U} \end{array}\right] \left[\begin{array}{c} \delta\mathbf{x}(t_i) \\ \delta\mathbf{q}(t_i) \\ \hline \delta\mathbf{u}(t_i) \end{array}\right] \right\} \Rightarrow$$

$$\Rightarrow J = \sum_{i=0}^{\infty} \frac{1}{2} \left(\begin{bmatrix} \mathbf{x_a}(t_i) \\ \mathbf{u_a}(t_i) \end{bmatrix}^T \begin{bmatrix} \boldsymbol{X_a} & \boldsymbol{S_a} \\ \boldsymbol{S_a}^T & \boldsymbol{U} \end{bmatrix} \begin{bmatrix} \mathbf{x_a}(t_i) \\ \mathbf{u_a}(t_i) \end{bmatrix} \right)$$

A solução, obtida através das equações (6.1.1), (6.1.2) e (6.1.3), é da forma:

$$\mathbf{u_a}(t_i) = -\boldsymbol{K_c}\mathbf{x_a}(t_i) \Rightarrow$$

$$\Rightarrow \delta\mathbf{u}(t_i) = -[\boldsymbol{K_{c1}} \mid \boldsymbol{K_{c2}}] \left[\begin{array}{c} \delta\mathbf{x}(t_i) \\ \hline \delta\mathbf{q}(t_i) \end{array}\right] = -\boldsymbol{K_{c1}}\delta\mathbf{x}(t_i) - \boldsymbol{K_{c2}}\delta\mathbf{q}(t_i)$$

Substituindo (6.3.3), (6.3.4) e (6.3.5):

$$[\mathbf{u}(t_i) - \mathbf{u_0}] = -\boldsymbol{K_{c1}}[\mathbf{x}(t_i) - \mathbf{x_0}] - \boldsymbol{K_{c2}}[\mathbf{q}(t_i) - \mathbf{q_0}]$$

Rearranjando os termos e substituindo $\mathbf{x_0} = [0, 0]^T$ (posição de equilíbrio em torno da origem):

$$\mathbf{u}(t_i) = -\boldsymbol{K_{c1}}\mathbf{x}(t_i) - \boldsymbol{K_{c2}}\mathbf{q}(t_i) + [\mathbf{u_0} + \boldsymbol{K_{c2}}\mathbf{q_0}] \tag{6.3.7}$$

Os valores das sub-matrizes de ganhos $\boldsymbol{K_{c1}}$ e $\boldsymbol{K_{c2}}$ são obtidos através da solução da equação de Riccati descrita na Seção 6.1. Os valores de $\mathbf{u_0}$ e $\mathbf{q_0}$ ainda necessitam ser determinados.

A primeira equação de (6.3.2) consiste em:

$$\mathbf{x}(t_{i+1}) = \boldsymbol{A}\mathbf{x}(t_i) + \boldsymbol{B}\mathbf{u}(t_i) + \boldsymbol{E}\mathbf{d}$$

A solução em regime é: $\mathbf{x}(t_{i+1}) = \mathbf{x}(t_i) = \mathbf{x_0} = \mathbf{0}$. Substituindo:

$$\mathbf{0} = \boldsymbol{B}\mathbf{u_0} + \boldsymbol{E}\mathbf{d} \Rightarrow \mathbf{u_0} = -\boldsymbol{B}^{-1}\boldsymbol{E}\mathbf{d}$$

Como o modelo apresenta uma entrada de perturbações constantes, o valor de regime de $\mathbf{u}$ é não nulo e também constante, para que essa entrada seja compensada. Contudo, como $\mathbf{d}$ é desconhecido, o valor de $\mathbf{u_0}$ também é desconhecido. O valor de $\mathbf{q_0}$, por sua vez, não está sujeito a nenhuma condição, podendo ser escolhido de forma totalmente livre. Dessa maneira, adota-se:

$$\mathbf{q_0} = {\boldsymbol{K_{c2}}}^{-1}\boldsymbol{B}^{-1}\boldsymbol{E}\mathbf{d}$$

Substituindo em (6.3.7):

$$\mathbf{u}(t_i) = -\boldsymbol{K_{c1}}\mathbf{x}(t_i) - \boldsymbol{K_{c2}}\mathbf{q}(t_i) + [-\boldsymbol{B}^{-1}\boldsymbol{E}\mathbf{d} + \boldsymbol{K_{c2}}{\boldsymbol{K_{c2}}}^{-1}\boldsymbol{B}^{-1}\boldsymbol{E}\mathbf{d}] \Rightarrow$$

$$\Rightarrow \mathbf{u}(t_i) = -\boldsymbol{K_{c1}}\mathbf{x}(t_i) - \boldsymbol{K_{c2}}\mathbf{q}(t_i) \tag{6.3.8}$$

Portanto, apesar de os valores de $\mathbf{u_0}$ e $\mathbf{q_0}$ serem dependentes da perturbação desconhecida $\mathbf{d}$, a lei de controle pode ser completamente determinada, já que $\mathbf{u_0} + \boldsymbol{K_{c2}}\mathbf{q_0}$ é igual a zero. Naturalmente:

$$\mathbf{u_0} = -\boldsymbol{K_{c1}}\mathbf{x_0} - \boldsymbol{K_{c2}}\mathbf{q_0} = -\boldsymbol{K_{c2}}[{\boldsymbol{K_{c2}}}^{-1}\boldsymbol{B}^{-1}\boldsymbol{E}\mathbf{d}] \Rightarrow \mathbf{u_0} = -\boldsymbol{B}^{-1}\boldsymbol{E}\mathbf{d}$$

que é o valor de regime necessário.

A partir das equações (6.3.1) e (6.3.8), e como $\mathbf{y}(t_i) = \mathbf{x}(t_i)$ (desprezando o vetor de ruídos), o diagrama da Figura 6.1 pode ser construído.

Para o cálculo da matriz de ganhos $\boldsymbol{K_c}$, assume-se um período de amostragem $T = 0,5$ s e as matrizes de peso dadas por:

$$\begin{bmatrix} \boldsymbol{X_a} & \boldsymbol{S_a} \\ {\boldsymbol{S_a}}^T & \boldsymbol{U} \end{bmatrix} = \begin{bmatrix} 10.\mathbf{I_4} & \mathbf{0} \\ \mathbf{0} & \mathbf{I_2} \end{bmatrix}$$

Dessa maneira, são considerados somente os termos quadráticos principais (os termos cruzados são ignorados) das matrizes de peso referentes a $\mathbf{x}$, $\mathbf{q}$ e $\mathbf{u}$. O peso referente a $\mathbf{x}$ está

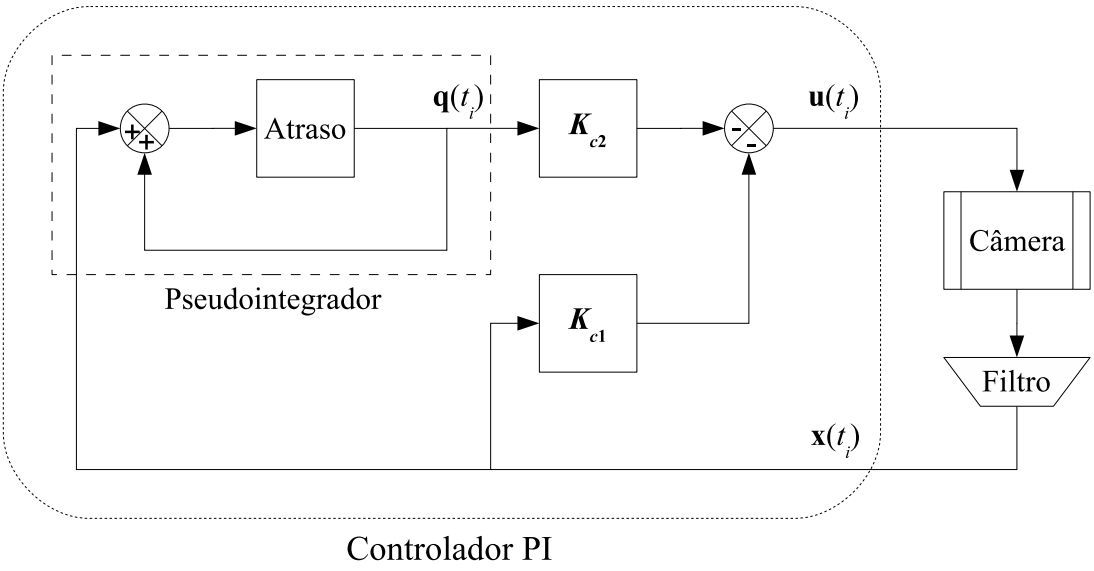

Figura 6.1: Diagrama de blocos do sistema com o controlador proporcional integral.

relacionado com a anulação dos valores das variáveis de estado, enquanto que o referente a $\mathbf{q}$ com a compensação dos valores das perturbações (constantes). Assumindo que os deslocamentos da câmera (e, portanto, $\mathbf{u}$) são pequenos (ver o Apêndice B), a minimização dos valores das variáveis de controle não é considerada prioritária. Assim, a matriz de peso $\boldsymbol{U}$ é admitida com valor relativo menor.

Substituindo as matrizes do modelo aumentado (6.3.6) em (6.1.2) e (6.1.3):

$$\boldsymbol{K_c}(t_i) = [\boldsymbol{U} + \boldsymbol{B_a}^T \boldsymbol{X_c}(t_{i+1}) \boldsymbol{B_a}]^{-1} [\boldsymbol{B_a}^T \boldsymbol{X_c}(t_{i+1}) \boldsymbol{A_a} + \boldsymbol{S_a}^T]$$
$$\boldsymbol{X_c}(t_i) = \boldsymbol{X_a} + \boldsymbol{A_a}^T \boldsymbol{X_c}(t_{i+1}) \boldsymbol{A_a} - [\boldsymbol{B_a}^T \boldsymbol{X_c}(t_{i+1}) \boldsymbol{A_a} + \boldsymbol{S_a}^T]^T \boldsymbol{K_c}(t_i)$$

Utilizando os valores numéricos de $\boldsymbol{A_a}$, $\boldsymbol{B_a}$, $\boldsymbol{X_a}$, $\boldsymbol{S_a}$ e $\boldsymbol{U}$ e o software Scilab, a solução da equação é calculada e fornece a matriz de ganhos do controlador dada por:

$$\boldsymbol{K_c} = \begin{bmatrix} 2,8040 & 0 & 1,0066 & 0 \\ 0 & 2,8040 & 0 & 1,0066 \end{bmatrix}$$

Logo:

$$\boldsymbol{K_{c1}} = \begin{bmatrix} 2,8040 & 0 \\ 0 & 2,8040 \end{bmatrix} \qquad \boldsymbol{K_{c2}} = \begin{bmatrix} 1,0066 & 0 \\ 0 & 1,0066 \end{bmatrix}$$

6.4 Controlador Proporcional com Estimação da Perturbação

Este controlador parte do princípio de que os valores das variáveis de perturbação são conhecidos e podem ser deduzidos das variáveis de estado, de modo que uma simples ação proporcional se torna suficiente para a estabilização do sistema. Para tanto, uma estimação do vetor de perturbações se faz necessária, sendo realizada através da utilização de um filtro de Kalman.

O modelo escolhido para as perturbações é o mesmo que o utilizado por Papanikolopoulos, Khosla e Kanade (1993), correspondente ao chamado passeio aleatório (FLEURY, 1981/1982):

$$\mathbf{d}(t_{i+1}) = \mathbf{d}(t_i) + T.\mathbf{w_d}(t_i) \tag{6.4.1}$$

onde:

$\mathbf{w_d}$ = vetor de ruído branco com estatística conhecida:

$$\mathrm{E}[\mathbf{w_d}(t_i)] = \mathbf{0}$$

$$\mathrm{E}[\mathbf{w_d}(t_i)\mathbf{w_d}^T(t_j)] = \begin{cases} \boldsymbol{Q_d}, & t_i = t_j \\ \mathbf{0}, & t_i \neq t_j \end{cases}$$

O vetor de perturbações é modelado como o equivalente em tempo discreto da saída de um integrador forçado por ruído branco, calculado a partir de uma condição inicial conhecida. Utilizando um modelo aumentado com as equações (4.1.2) e (4.1.3), temos:

$$\left| \begin{array}{l} \begin{bmatrix} \mathbf{x}(t_{i+1}) \\ \mathbf{d}(t_{i+1}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{E} \\ \mathbf{0} & \mathbf{I_2} \end{bmatrix} \begin{bmatrix} \mathbf{x}(t_i) \\ \mathbf{d}(t_i) \end{bmatrix} + \begin{bmatrix} \boldsymbol{B} \\ \mathbf{0} \end{bmatrix} \mathbf{u}(t_i) + T \begin{bmatrix} \mathbf{0} \\ \mathbf{I_2} \end{bmatrix} \mathbf{w_d}(t_i) \\ \qquad \mathbf{y}(t_i) = \begin{bmatrix} \boldsymbol{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}(t_i) \\ \mathbf{d}(t_i) \end{bmatrix} + \mathbf{v}(t_i) \end{array} \right. \Rightarrow$$

$$\Rightarrow \left| \begin{array}{l} \mathbf{x_a}(t_{i+1}) = \boldsymbol{A_a}\mathbf{x_a}(t_i) + \boldsymbol{B_a}\mathbf{u}(t_i) + \boldsymbol{G_a}\mathbf{w_d}(t_i) \\ \qquad \mathbf{y}(t_i) = \boldsymbol{C_a}\mathbf{x_a}(t_i) + \mathbf{v}(t_i) \end{array} \right. \tag{6.4.2}$$

Pode-se observar que o vetor $\mathbf{w}$ de incertezas sobre o modelo não está presente na equação. Assume-se que essas incertezas estão englobadas de forma implícita nas do modelo das perturbações ($\mathbf{w_d}$).

A matriz $\boldsymbol{K_f}$ é determinada a partir de (6.2.2), (6.2.3) e (6.2.5). Dessa maneira, de (6.2.6)

e (6.2.7), as equações a serem utilizadas no filtro passam a ser:

$$\hat{\mathbf{x}}_\mathbf{a}(t_i^-) = \boldsymbol{A_a}\hat{\mathbf{x}}_\mathbf{a}(t_{i-1}^+) + \boldsymbol{B_a}\mathbf{u}(t_{i-1})$$
$$\hat{\mathbf{x}}_\mathbf{a}(t_i^+) = \hat{\mathbf{x}}_\mathbf{a}(t_i^-) + \boldsymbol{K_f}[\mathbf{y}(t_i) - \boldsymbol{C_a}\hat{\mathbf{x}}_\mathbf{a}(t_i^-)] =$$
$$= \hat{\mathbf{x}}_\mathbf{a}(t_i^-) + \begin{bmatrix}\boldsymbol{K_{f1}}\\ \boldsymbol{K_{f2}}\end{bmatrix}[\mathbf{y}(t_i) - \boldsymbol{C_a}\hat{\mathbf{x}}_\mathbf{a}(t_i^-)]$$

Substituindo os vetores e as matrizes do modelo aumentado por suas componentes e os valores de $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ e $\boldsymbol{E}$ do modelo da câmera:

$$\hat{\mathbf{x}}(t_i^-) = \hat{\mathbf{x}}(t_{i-1}^+) + T.\mathbf{u}(t_{i-1}) + T.\hat{\mathbf{d}}(t_{i-1}^+) \tag{6.4.3}$$
$$\hat{\mathbf{d}}(t_i^-) = \hat{\mathbf{d}}(t_{i-1}^+) \tag{6.4.4}$$

$$\hat{\mathbf{x}}(t_i^+) = \hat{\mathbf{x}}(t_i^-) + \boldsymbol{K_{f1}}[\mathbf{y}(t_i) - \hat{\mathbf{x}}(t_i^-)] \tag{6.4.5}$$
$$\hat{\mathbf{d}}(t_i^+) = \hat{\mathbf{d}}(t_i^-) + \boldsymbol{K_{f2}}[\mathbf{y}(t_i) - \hat{\mathbf{x}}(t_i^-)] \tag{6.4.6}$$

Calculada a estimativa do vetor de estados aumentado, o passo seguinte é determinar o controlador do sistema. Entretanto, utilizando o software Scilab, pode ser verificado, através das matrizes $\boldsymbol{A_a}$ e $\boldsymbol{B_a}$, que o modelo dado por (6.4.2) não é controlável (FLEURY, 1981/1982). Dessa maneira, não é possível projetar um controlador para o sistema. Para contornar este problema, Papanikolopoulos, Khosla e Kanade (1993) utilizam uma estratégia que tem como base a definição de um vetor de controle modificado $\mathbf{u_n}$.

$$\mathbf{u_n}(t_i) = \mathbf{u}(t_i) + \mathbf{d}(t_i)$$

Substituindo em (4.1.2), ignorando as entradas de incertezas ($\mathbf{w}$) e lembrando que $\boldsymbol{E} = \boldsymbol{B}$:

$$\mathbf{x}(t_{i+1}) = \boldsymbol{A}\mathbf{x}(t_i) + \boldsymbol{B}\mathbf{u_n}(t_i)$$

Com este modelo simplificado, um controlador pode ser projetado, de modo a minimizar a seguinte função de custo:

$$J = \sum_{i=0}^{\infty}\left\{\frac{1}{2}\begin{bmatrix}\mathbf{x}(t_i)\\ \mathbf{u_n}(t_i)\end{bmatrix}^T\begin{bmatrix}\boldsymbol{X} & \boldsymbol{S_n}\\ \boldsymbol{S_n}^T & \boldsymbol{U_n}\end{bmatrix}\begin{bmatrix}\mathbf{x}(t_i)\\ \mathbf{u_n}(t_i)\end{bmatrix}\right\}$$

Novamente, utilizando as equações (6.1.1), (6.1.2) e (6.1.3), obtém-se:

$$\mathbf{u_n}(t_i) = -\boldsymbol{K_c}\mathbf{x}(t_i) \Rightarrow$$
$$\Rightarrow \mathbf{u}(t_i) = -\boldsymbol{K_c}\mathbf{x}(t_i) - \mathbf{d}(t_i)$$

Utilizando as estimativas realizadas pelo filtro de Kalman:

$$\mathbf{u}(t_i) = -\boldsymbol{K_c}\hat{\mathbf{x}}(t_i^+) - \hat{\mathbf{d}}(t_i^+) \tag{6.4.7}$$

O diagrama de blocos do sistema com este controlador, utilizando as equações (6.4.3), (6.4.4), (6.4.5), (6.4.6) e (6.4.7), é dado na Figura 6.2.

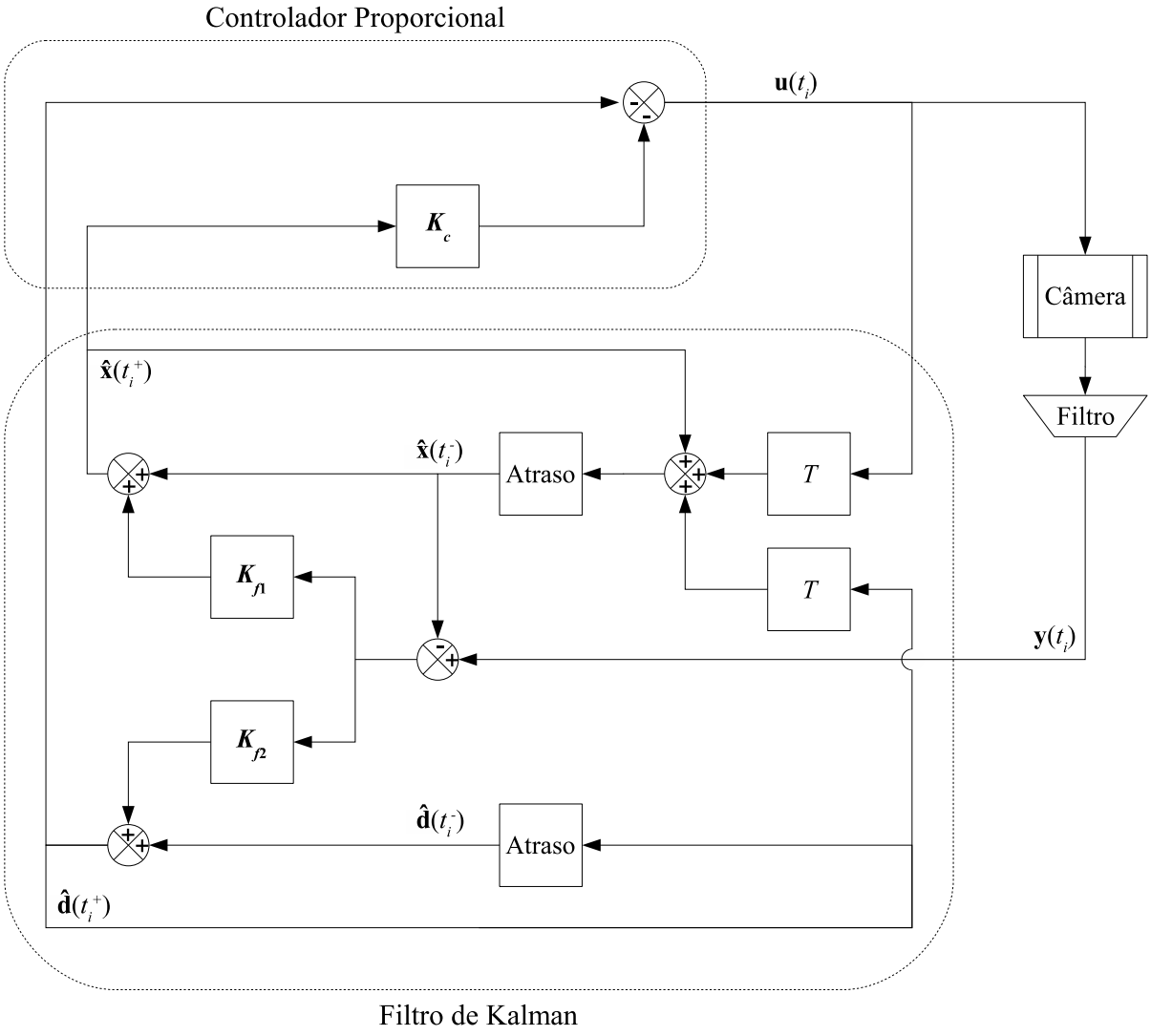

Figura 6.2: Diagrama de blocos do sistema com o controlador linear quadrático gaussiano.

Para o cálculo da matriz de ganhos de Kalman ($\boldsymbol{K_f}$) e da matriz de ganhos do controlador

($\boldsymbol{K_c}$), considera-se novamente um período de amostragem $T = 0,5$ s. Em relação à região-alvo utilizada no filtro, ela é considerada uma janela de tamanho 50×50 pixels centralizada na imagem. Já os valores de desvio padrão para os vetores $\mathbf{w_d}$ e $\mathbf{v}$ são adotados como correspondentes a 10 % e 5 % do tamanho da janela, respectivamente. Entretanto, pelas equações (4.1.2) e (6.4.1), percebe-se que o efeito do vetor de ruídos $\mathbf{w_d}$ sobre o vetor de estados $\mathbf{x}$ é multiplicado por T^2 (ou seja, a unidade de $\mathbf{w_d}$ é pixels/s^2). Assim, para proporcionar um desvio de 10 % do tamanho da janela, o valor real do desvio de $\mathbf{w_d}$ deve ser $\frac{(0,10)\times 50}{T^2} = 20$ pixels/s^2, enquanto o desvio de $\mathbf{v}$ é igual a $(0,05) \times 50 = 2,5$ pixels. Dessa maneira, as matrizes de covariâncias são dadas por:

$$\boldsymbol{Q_d} = \begin{bmatrix} 400 & 0 \\ 0 & 400 \end{bmatrix} \qquad \boldsymbol{R} = \begin{bmatrix} 6,25 & 0 \\ 0 & 6,25 \end{bmatrix}$$

Neste caso, os termos cruzados das matrizes também são ignorados. Substituindo as matrizes do modelo aumentado (6.4.2) em (6.2.2), (6.2.3) e (6.2.5):

$$\begin{aligned} \boldsymbol{P}(t_i^-) &= \boldsymbol{A_a}\boldsymbol{P}(t_{i-1}^+)\boldsymbol{A_a}^T + \boldsymbol{G_a}\boldsymbol{Q_d}\boldsymbol{G_a}^T \\ \boldsymbol{K_f}(t_i) &= \boldsymbol{P}(t_i^-)\boldsymbol{C_a}^T[\boldsymbol{C_a}\boldsymbol{P}(t_i^-)\boldsymbol{C_a}^T + \boldsymbol{R}]^{-1} \\ \boldsymbol{P}(t_i^+) &= \boldsymbol{P}(t_i^-) - \boldsymbol{K_f}(t_i)\boldsymbol{C_a}\boldsymbol{P}(t_i^-) \end{aligned}$$

Substituindo os valores de $\boldsymbol{A_a}$, $\boldsymbol{C_a}$, $\boldsymbol{G_a}$, $\boldsymbol{Q_d}$ e $\boldsymbol{R}$ e resolvendo as equações anteriores progressivamente utilizando o software Scilab, assumindo uma condição inicial nula ($\boldsymbol{P}(t_0^+) = \boldsymbol{P_0} = \mathbf{0}$), o valor de estabilização obtido para a matriz de Kalman é dado por:

$$\boldsymbol{K_f} = \begin{bmatrix} 0,8803 & 0 \\ 0 & 0,8803 \\ 1,3841 & 0 \\ 0 & 1,3841 \end{bmatrix}$$

Logo:

$$\boldsymbol{K_{f1}} = \begin{bmatrix} 0,8803 & 0 \\ 0 & 0,8803 \end{bmatrix} \qquad \boldsymbol{K_{f2}} = \begin{bmatrix} 1,3841 & 0 \\ 0 & 1,3841 \end{bmatrix}$$

Em relação ao controlador, considera-se novamente uma matriz de peso para $\mathbf{x}$ com maior contribuição do que $\mathbf{u}$ e os termos diagonais nulos.

$$\begin{bmatrix} \boldsymbol{X} & \boldsymbol{S_n} \\ \boldsymbol{S_n}^T & \boldsymbol{U_n} \end{bmatrix} = \begin{bmatrix} 10.\mathbf{I_2} & \mathbf{0} \\ \mathbf{0} & \mathbf{I_2} \end{bmatrix}$$

Assim, as equações (6.1.3) e (6.1.2) se tornam:

$$\boldsymbol{K_c}(t_i) = [\boldsymbol{U_n} + \boldsymbol{B}^T \boldsymbol{X_c}(t_{i+1})\boldsymbol{B}]^{-1}[\boldsymbol{B}^T \boldsymbol{X_c}(t_{i+1})\boldsymbol{A} + \boldsymbol{S_n}^T]$$
$$\boldsymbol{X_c}(t_i) = \boldsymbol{X} + \boldsymbol{A}^T \boldsymbol{X_c}(t_{i+1})\boldsymbol{A} - [\boldsymbol{B}^T \boldsymbol{X_c}(t_{i+1})\boldsymbol{A} + \boldsymbol{S_n}^T]^T \boldsymbol{K_c}(t_i)$$

Novamente, substituindo os valores numéricos de $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{X}$, $\boldsymbol{S_n}$ e $\boldsymbol{U_n}$, o software Scilab fornece a solução da equação anterior, de modo que a matriz de ganhos do controlador é dada por:

$$\boldsymbol{K_c} = \begin{bmatrix} 1,5311 & 0 \\ 0 & 1,5311 \end{bmatrix}$$

Portanto, os controladores desenvolvidos são capazes de receber os valores de posição $\mathbf{x} = [x_1, x_2]^T$ e determinar os valores de $\mathbf{u} = [u_1, u_2]^T$ que realimentam o sistema visando anular os valores de posição. Contudo, como explicado no Capítulo 4, antes de movimentar a câmera é necessária ainda a conversão dos valores de $\mathbf{u}$ para velocidades de rotação (pan–tilt), através das equações (4.1.9) e (4.1.8).

7 EQUIPAMENTO UTILIZADO E IMPLANTAÇÃO DO SOFTWARE

Após o desenvolvimento de algoritmos de rastreamento e controle servo visual, o passo seguinte é implementá-los em uma câmera pan-tilt. Dessa maneira, é possível a realização de testes que permitam uma análise de desempenho dos algoritmos. O equipamento concebido para este trabalho consiste de um módulo pan-tilt integrado a uma webcam, sendo que ambos são conectados a um microcomputador (ver a Figura 7.1). No microcomputador funciona um software, no qual estão implementados os algoritmos de rastreamento e controle. O programa tem como entrada as imagens fornecidas pela webcam e transmite como saída as posições angulares em que o módulo pan-tilt deve ser posicionado.

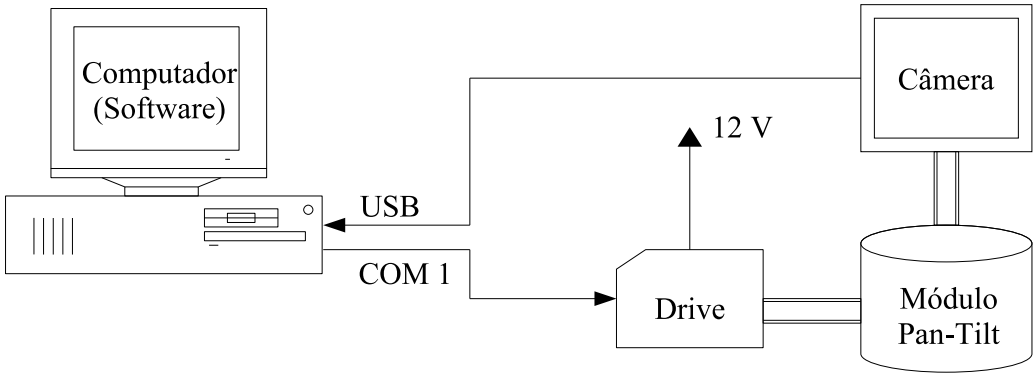

Figura 7.1: Esquema do equipamento utilizado.

A seguir, encontram-se detalhados os componentes do equipamento utilizado e o software desenvolvido para o seu funcionamento.

7.1 Câmera

A câmera utilizada neste trabalho é uma webcam comum, fabricada pela Shenzhen Akkord Electronics Co., modelo VC2P. Ela possui sensor CMOS de $1/4$", se comunica com o computador via porta USB 1.1 e é configurada para resolução de 320×240 pixels, com taxa de atualização de 30 fps. As imagens são adquiridas em formato RGB 24-bit, ou seja, cada pixel é representado por uma combinação de três componentes de cores (vermelho, verde e azul), cada uma representada por um byte (valores de 0 a 255). Como os algoritmos de rastreamento admitem como entrada uma imagem em tons de cinza, uma conversão é necessária. Essa transformação é realizada convertendo o modelo de cores da imagem de RGB para YIQ

(MATTHYS, 2001). Assim, a informação relativa à luminosidade (intensidade) da imagem é dada pela componente Y, desacoplada das componentes responsáveis pela cor propriamente dita (I e Q). Dessa maneira, a imagem em tons de cinza é dada pela componente Y:

$$I = \frac{0,299 \times I_R + 0,587 \times I_G + 0,114 \times I_B}{255}$$

onde:

I = valor de um pixel da imagem em tons de cinza

I_R = valor da componente vermelha de um pixel da imagem colorida

I_G = valor da componente verde de um pixel da imagem colorida

I_B = valor da componente azul de um pixel da imagem colorida

A divisão por 255 realizada na equação busca normalizar os valores dos pixels para um intervalo entre 0 a 1, de modo a evitar possíveis problemas relativos a aproximações numéricas.

Um dado importante ainda não determinado corresponde ao valor da distância focal da lente da câmera, utilizada no cálculo das rotações pan-tilt. Ela é determinada no Apêndice A através de dois métodos diferentes. O valor utilizado nos algoritmos é o obtido pelo método experimental, que corresponde a $f = 332,01$ pixels.

7.2 Módulo Pan-tilt

O dispositivo pan-tilt responsável pela movimentação da câmera foi desenvolvido na Escola Politécnica da Universidade de São Paulo. O módulo é composto pelo mecanismo pan-tilt propriamente dito, por dois motores responsáveis pelas rotações pan e tilt e por um drive para o acionamento dos atuadores. Os atuadores correspondem a servo-motores da linha “Hobbico Command Servos” (CS–61), fabricados pela Futaba. O seu posicionamento é realizado pelo drive de acionamento, que, por sua vez, recebe as informações de posição pan e tilt do microcomputador. O recebimento desses dados é realizado via comunicação serial (padrão RS-232), sendo que a porta de entrada física do drive é um conector DB-09, que é ligado por meio de um cabo a uma porta COM do computador. O drive é alimentado por uma tensão contínua de 12 V e a transmissão de dados é configurada como assíncrona, com taxa de transferência de 9600 bps, 8 bits de dados, nenhum bit de paridade, 1 bit de parada e sem controle de fluxo.

O curso permitido para cada servo-motor é de 128 graus, sendo que a resolução de posicionamento é de 1 grau. A movimentação do módulo é realizada através do envio de um byte

do computador para o drive. O byte pode assumir 256 valores diferentes, entre 0 e 255. Um valor entre 0 e 127 realiza o acionamento de um motor (eixo tilt), enquanto que um valor entre 128 e 255 aciona o outro (eixo pan). Os valores 0 e 127 deslocam o eixo tilt para o início e o fim do curso do motor, respectivamente. Os valores intermediários posicionam o eixo em posições interpostas. Dessa maneira, os valores de 0 a 127 posicionam o eixo tilt em 128 posições diferentes consecutivas, defasadas de 1 grau. Para o eixo pan, o procedimento é análogo, considerando os valores entre 128 e 255.

Entretanto, considerando a modelagem realizada do sistema, as saídas do controlador são valores de velocidade de rotação, e não de posição. Assim, o algoritmo de controle não pode ser aplicado diretamente. A conversão utilizada entre valores de velocidade e posição é bem simples. Como as velocidades de rotação calculadas (R_X e R_Y) são constantes entre instantes de amostragem consecutivos, o efeito dessas rotações pode ser reproduzido no dispositivo através do deslocamento de sua posição de $R_X \times T$ e $R_Y \times T$, durante cada período de amostragem. Para que essa conversão seja válida, é necessário que o servo-motor seja rápido o suficiente para que a posição determinada seja atingida antes do instante de amostragem seguinte. De acordo com o Apêndice B, a rotação do servo-motor é suficiente para os deslocamentos típicos do alvo.

$$pan(t_i) = pan(t_{i-1}) + R_Y \times T \times \frac{180}{\pi}$$
$$tilt(t_i) = tilt(t_{i-1}) + R_X \times T \times \frac{180}{\pi}$$

onde:

$pan(t_i)$ = posição angular da câmera segundo a direção pan (graus)
$tilt(t_i)$ = posição angular da câmera segundo a direção tilt (graus)

Como as posições angulares a serem fornecidas ao módulo são dadas em graus, as rotações R_X e R_Y devem ser convertidas de rad/s para graus/s.

Dado que a resolução de posicionamento pan-tilt é de 1 grau, é possível estimar o seu equivalente no sistema de coordenadas da imagem. Esse valor corresponde ao menor deslocamento do alvo que é possível de se compensar com este equipamento.

Durante um período de amostragem ($T = 0,5$ s), um deslocamento de 1 grau corresponde à velocidade de $1 \div 0,5 = 2$ graus/s, que equivale a $0,0349$ rad/s. De acordo com o Apêndice B, a equação simplificada (B.0.1) relaciona a rotação tilt com a velocidade linear vertical correspondente no plano da imagem. Utilizando o valor de distância focal $f = 332,01$ pixels:

$$0,0349 = \frac{u_2}{332,01} \Rightarrow u_2 = 11,59 \text{ pixels/s}$$

Considerando o tempo de um período de amostragem, como na determinação da rotação de 2 graus/s, o deslocamento correspondente a essa velocidade é de $11,59 \times 0,5 = 5,794$ pixels. Assim, a resolução de 1 grau do movimento da câmera corresponde a $5,794$ pixels na imagem.

7.3 Computador e Software

O computador utilizado possui um processador Pentium 3 – 800MHz, 304 MB RAM e sistema operacional Microsoft Windows XP. O desenvolvimento do software de controle do dispositivo é realizado em ambiente Microsoft Visual C++ 6.0.

A criação do software é realizada como uma aplicação MFC, com o auxílio de duas classes externas. A primeira é a **CSerial**, desenvolvida por Klein (2004) para comunicação via porta serial. Ela é responsável por transmitir as posições desejadas para o módulo pan-tilt. A outra classe é a **CVMR_Capture**, desenvolvida por Sivasagar (2004), para aquisição de imagens da webcam. Ela possui uma função que fornece os três valores RGB (azul, verde e vermelho) de cada um dos 76800 (320×240) pixels de uma imagem. Nesta aplicação é adicionado o código dos algoritmos de rastreamento e dos controladores desenvolvido para este trabalho.

As Figuras 7.2, 7.3 e 7.4 ilustram a janela do software durante o seu funcionamento. O painel “Operação” permite a escolha de dois modos de atividade: “Manual” e “Rastrear”. Com o modo em “Manual”, o programa permite a movimentação da câmera pelo usuário a partir dos botões situados abaixo da imagem. O botão “0”, em particular, desloca a câmera para a sua posição inicial, que corresponde ao centro dos cursos de movimentação dos servo-motores ($tilt = 0$ corresponde ao envio de um byte de valor 63 e $pan = 0$ ao de valor 196). Assim, são permitidas posições pan e $tilt$ com valores entre -63 e 64 graus. Através dos menus “Algoritmo de análise” e “Algoritmo de controle”, podem-se escolher, respectivamente, o método de rastreamento (filtro) da imagem (“Algoritmo base”, “Est. incremental” ou “Est. em multiresolução”) e o procedimento de controle (“PI” ou “LQG”). Com a mudança do modo de operação para “Rastrear”, o padrão de referência é obtido a partir da primeira imagem adquirida e o sistema servo visual entra em ação. Após o recebimento de uma nova imagem e a aplicação dos algoritmos de rastreamento e controle, é mostrado o tempo de cálculo gasto pelos processos. Uma outra opção possível é dada pelo botão “Enquadrar”. Caso esteja selecionada, é mostrada na imagem os limites das regiões-alvo (se escolhido o algoritmo de “Estimação em multiresolução”, são

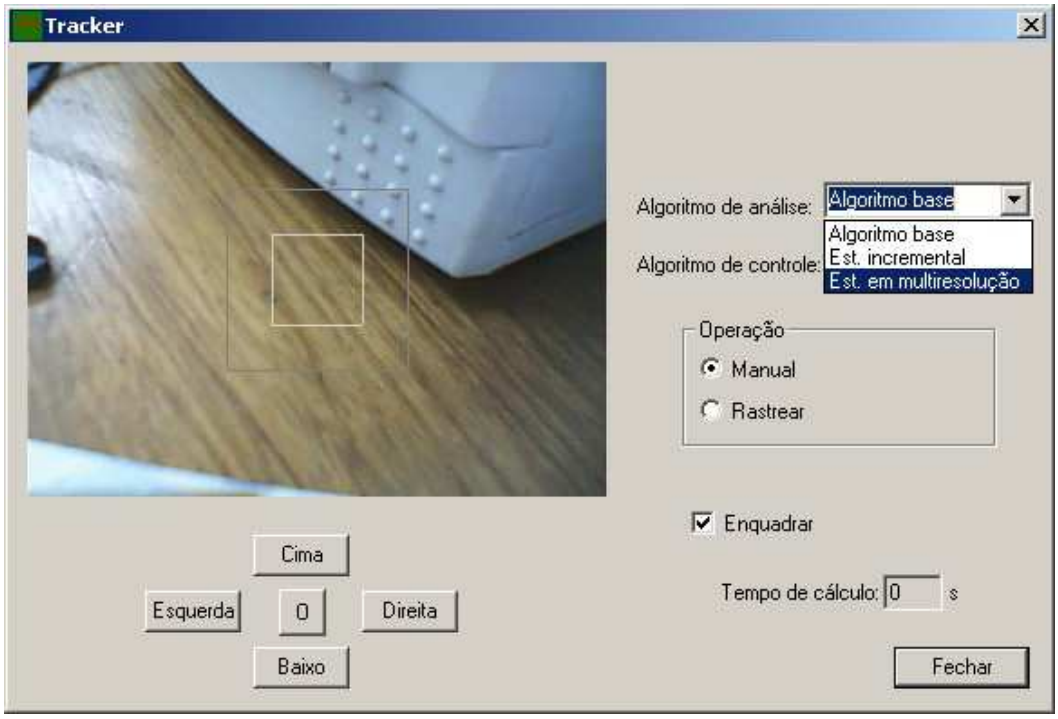

Figura 7.2: Ilustração do software em função manual (I).

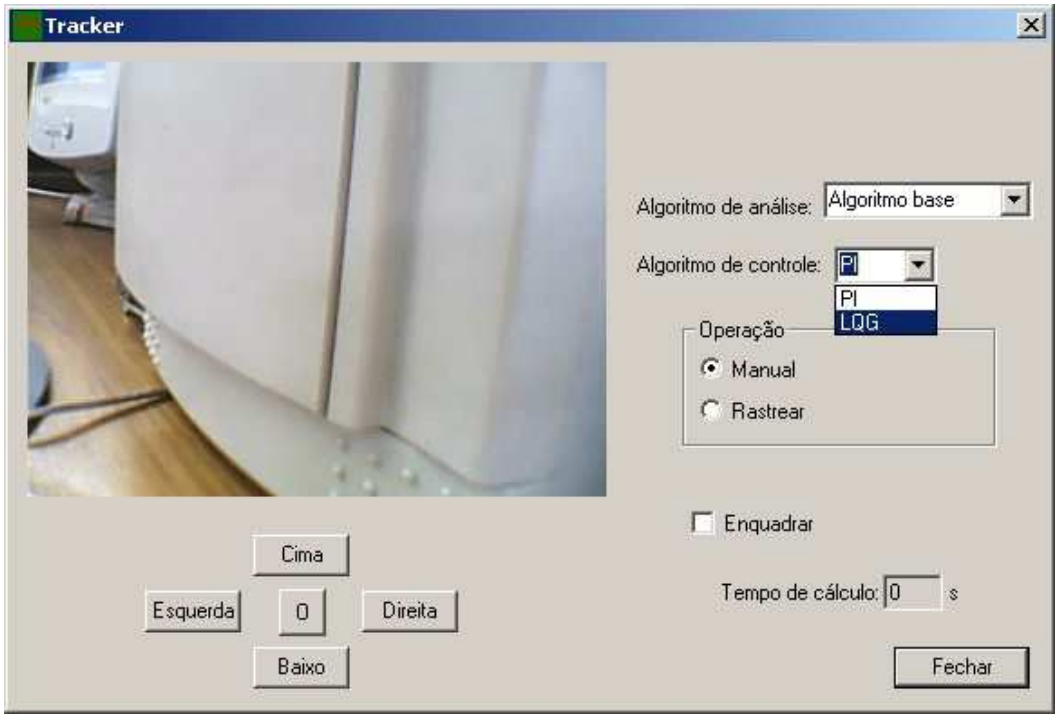

Figura 7.3: Ilustração do software em função manual (II).

mostrados os limites para as imagens original e em menor resolução).

O período de amostragem de imagens utilizado, como já mencionado no cálculo dos con-

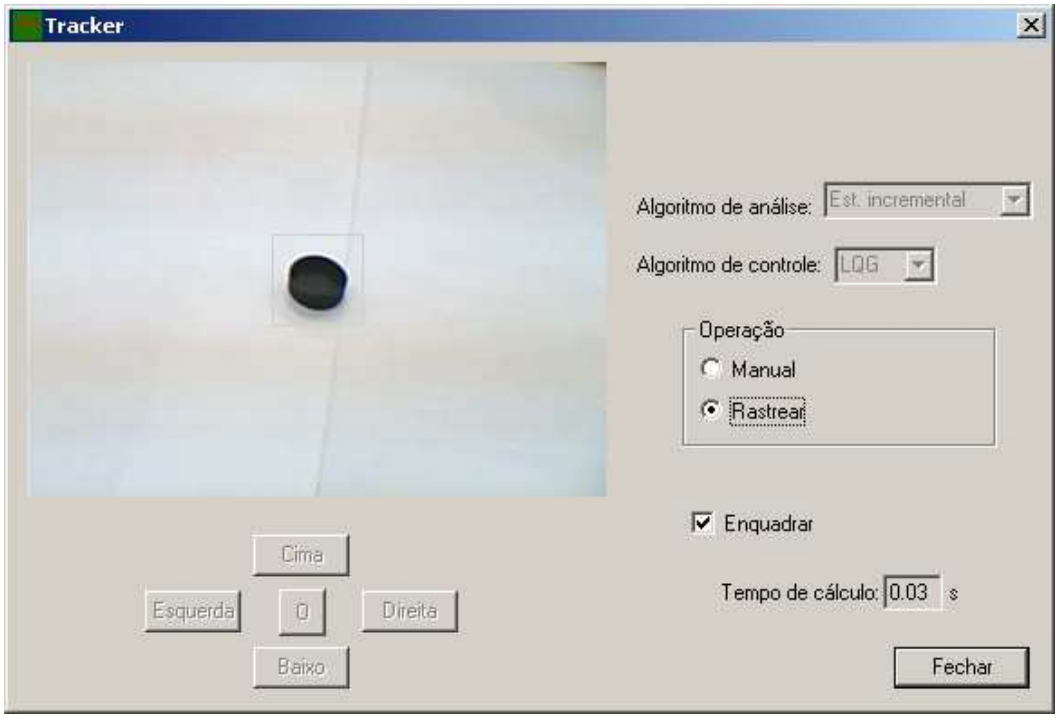

Figura 7.4: Ilustração do software em função de rastreamento.

troladores e na determinação do menor valor de deslocamento possível de ser compensado, vale $T = 0,5$ s. Este é o menor valor possível de ser utilizado que garantiu um desempenho confiável do equipamento durante os testes.

8 RESULTADOS EXPERIMENTAIS

Após o desenvolvimento do software que implementa os algoritmos de rastreamento e controle servo visual em um equipamento adequado, é possível a realização de testes que permitam verificar o desempenho do sistema. Foram realizados três testes distintos. O primeiro tem como objetivo a análise comparativa da performance de cada algoritmo de rastreamento somente (não a malha de controle servo visual como um todo). No segundo, a malha de controle é fechada e o sistema completo é observado. São obtidas respostas para cada combinação entre os procedimentos de rastreamento e controle apresentados, obtidas a partir de uma perturbação conhecida imposta ao alvo. Por último, é realizado um teste do sistema completo considerando um conjunto de deslocamentos do alvo ao longo do tempo, visando simular uma situação mais realística.

8.1 Teste do Filtro

Este primeiro teste tem como objetivo comparar os alcances dos algoritmos de rastreamento visual, utilizando diferentes tipos e tamanhos de deslocamento do alvo. Para tanto, é definido inicialmente um determinado padrão de referência. Em seguida, é fornecido um conjunto de imagens em que o alvo apresenta diferentes valores e tipos de deslocamento em relação à referência. Assim, para cada imagem, os algoritmos são aplicados e obtêm-se os valores de deslocamento, para que possam ser comparados posteriormente.

A imagem de referência que contém o alvo a ser rastreado (região-alvo de tamanho 50×50 pixels, centralizada na imagem) é dada pela Figura 8.1. O conjunto de imagens de teste utilizado é dado na Figura 8.2. As imagens utilizadas possuem tamanho 320×240 pixels. A região de referência $\big(\mathbf{I}(t_0)\big)$ está destacada na Figura 8.1 (quadrado tracejado menor). O quadrado maior corresponde ao equivalente da região para uma imagem de resolução duas vezes menor (utilizada no algoritmo estendido por multiresolução). A seqüência de imagens na Figura 8.2 indica o tipo de movimento analisado: nas imagens de 01 a 04, os deslocamentos predominantes são de translação, nas de 05 a 09 o alvo sofre rotações, as imagens de 10 a 14 mostram um afastamento ou uma aproximação do alvo e a última imagem apresenta uma combinação de todos esses tipos de deslocamento.

A configuração do alvo em cada imagem é considerada uma perturbação isolada de sua

Figura 8.1: Imagem contendo a região de referência utilizada no teste do filtro.

posição baseada em sua disposição inicial, e não uma seqüência considerando o alvo em movimento contínuo. Em outras palavras, cada uma das imagens de 01 a 15 da Figura 8.2 é considerada a primeira imagem adquirida depois de obtida a referência no início do algoritmo, ou seja, a obtida no instante de tempo t_1. O número máximo de iterações permitido é igual a 30 e os valores de erro máximo permitido são $0,05$, para as translações, e $0,005$, para rotação e escalonamento. Esses valores foram ajustados de forma empírica, após vários testes, e são aplicados não apenas neste teste como nos outros dois apresentados nas próximas seções.

Os resultados obtidos são dados pelos gráficos das Figuras 8.3 a 8.6. Os valores "reais" de deslocamento mostrados foram calculados através de uma análise visual sobre as imagens, auxiliada por ferramentas de translação, rotação e escalonamento do programa PhotoFiltre.

Em relação ao primeiro algoritmo (baseado na minimização do erro quadrático), os resultados mostram que seu desempenho, mesmo para deslocamentos em torno de 5 pixels da imagem 01, não é satisfatório. Assim, dado que o menor deslocamento que a câmera é capaz de compensar é aproximadamente $5,8$ pixels (Capítulo 7), justifica-se a necessidade da incorporação das extensões utilizadas.

O desempenho do algoritmo estendido por estimação incremental se mostra superior ao do anterior. Na primeira imagem, com valores de translação em torno de 5 pixels, o resultado obtido é correto. Entretanto, na imagem seguinte, com deslocamentos em torno de 10 pixels, o seu desempenho já se compromete um pouco. Para os deslocamentos de rotação, um movimento de aproximadamente 9 graus é devidamente rastreado (imagem 06). Já em relação ao escalonamento, o maior valor proporcionado de ampliação que possibilita uma detecção correta é de cerca de $1,2$.

Já o algoritmo estendido por estimação em multiresolução apresenta o maior alcance, permitindo o rastreamento dos maiores deslocamentos dentre os três procedimentos. Os valores máximos medidos adequadamente no teste são de, aproximadamente, 12 pixels (translação, para a imagem 03), 45 graus (rotação, para a imagem 08) e $1,4$ (escalonamento, para a imagem


Imagem nº 01


Imagem nº 02


Imagem nº 03


Imagem nº 04


Imagem nº 05


Imagem nº 06


Imagem nº 07


Imagem nº 08


Imagem nº 09


Imagem nº 10


Imagem nº 11


Imagem nº 12


Imagem nº 13


Imagem nº 14


Imagem nº 15

Figura 8.2: Conjunto de imagens utilizado no teste do filtro.

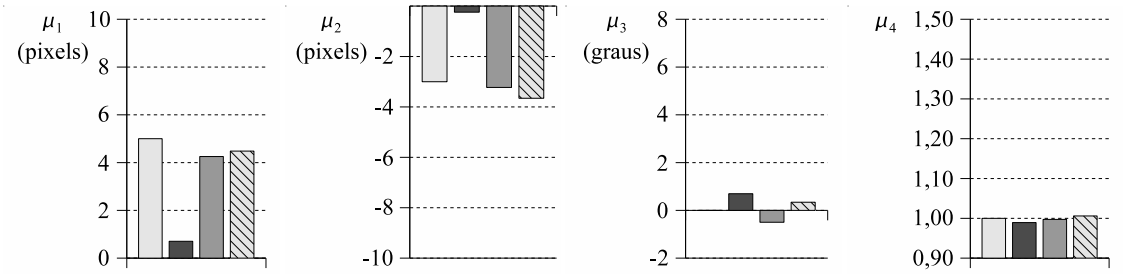

Imagem nº 01 (translação do alvo)

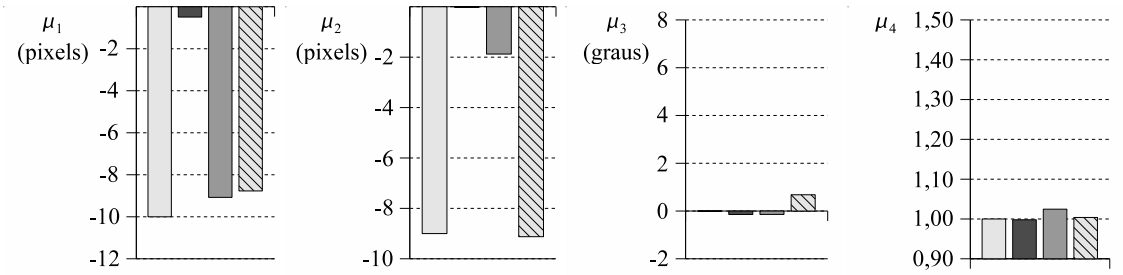

Imagem nº 02 (translação do alvo)

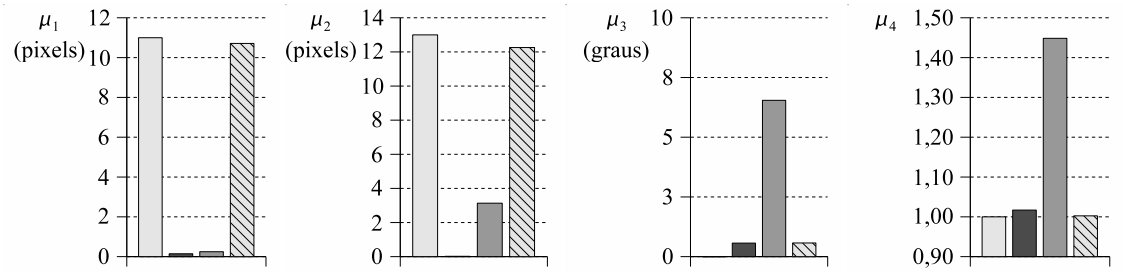

Imagem nº 03 (translação do alvo)

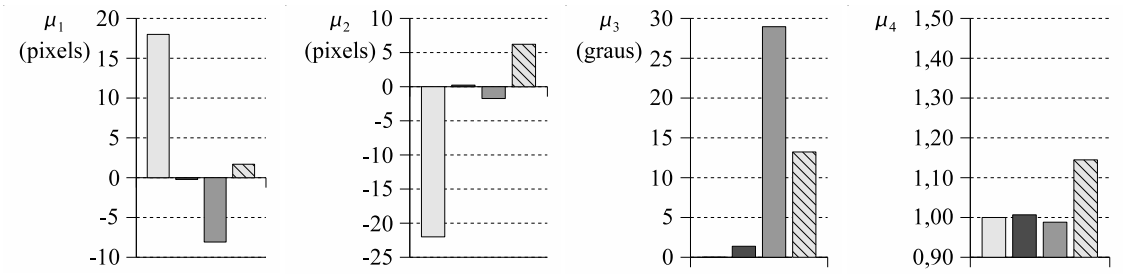

Imagem nº 04 (translação do alvo)

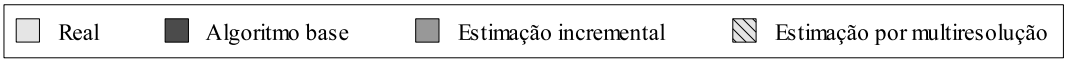

Figura 8.3: Valores de deslocamento obtidos no teste do filtro.

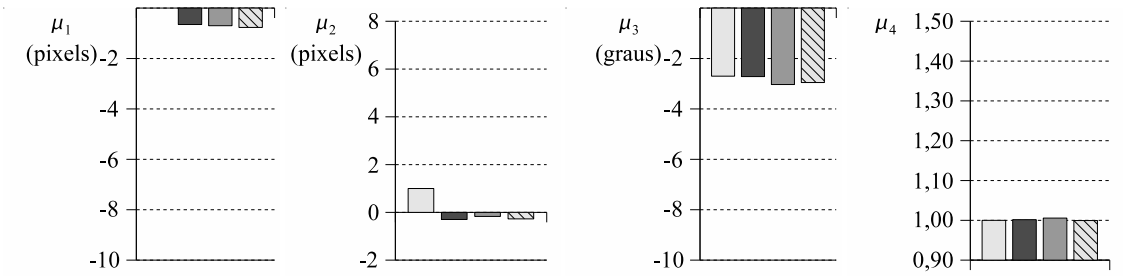

Imagem nº 05 (rotação do alvo)

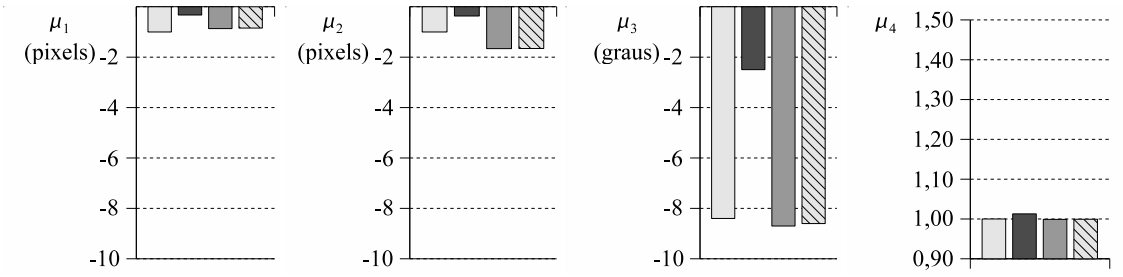

Imagem nº 06 (rotação do alvo)

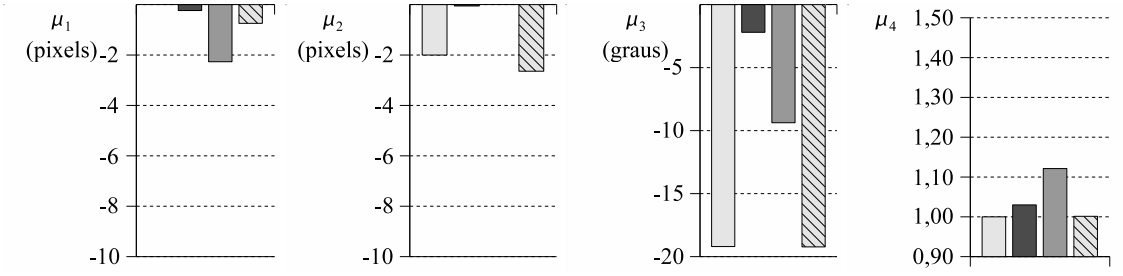

Imagem nº 07 (rotação do alvo)

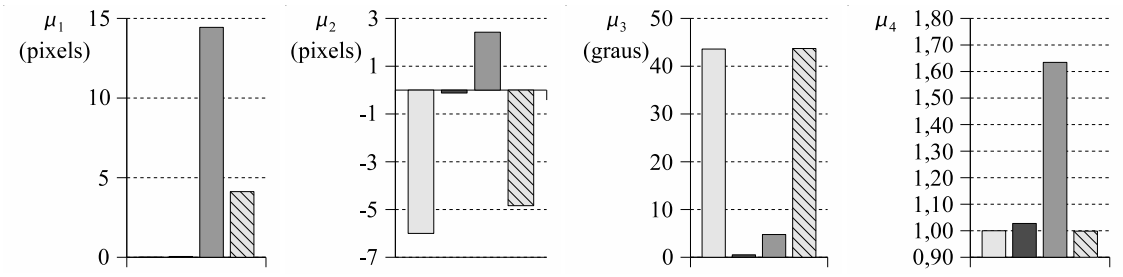

Imagem nº 08 (rotação do alvo)

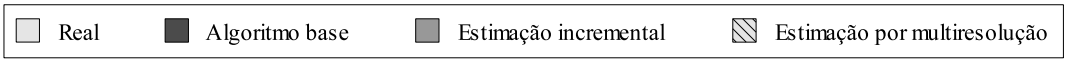

Figura 8.4: Valores de deslocamento obtidos no teste do filtro (continuação).

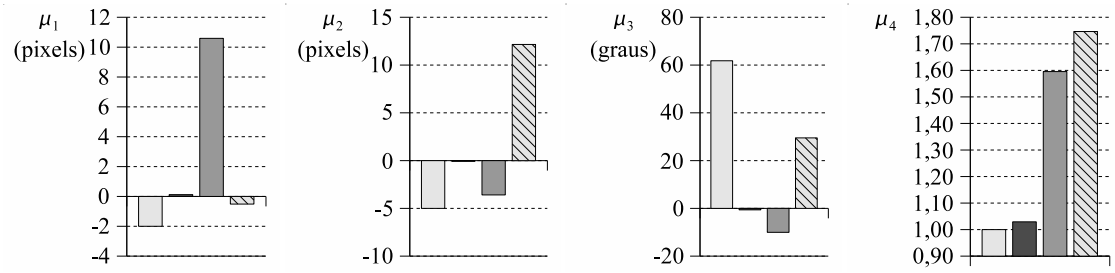

Imagem nº 09 (rotação do alvo)

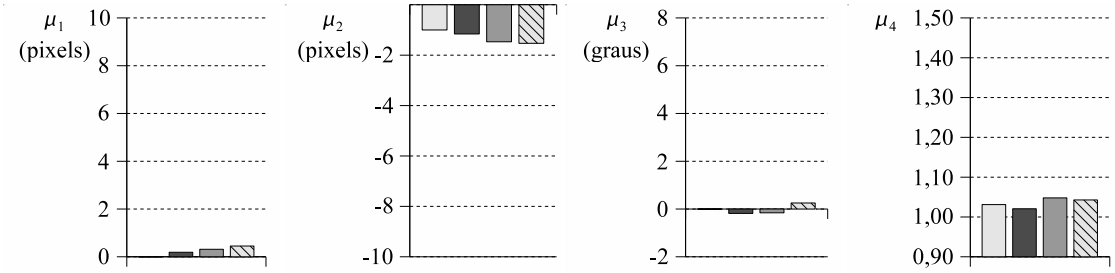

Imagem nº 10 (escalonamento do alvo - aproximação)

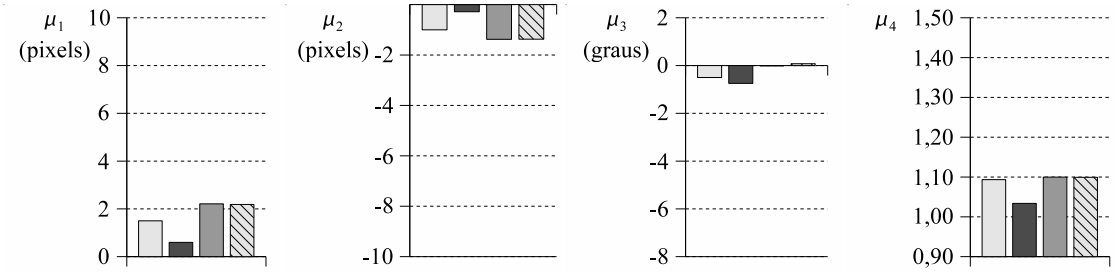

Imagem nº 11 (escalonamento do alvo - aproximação)

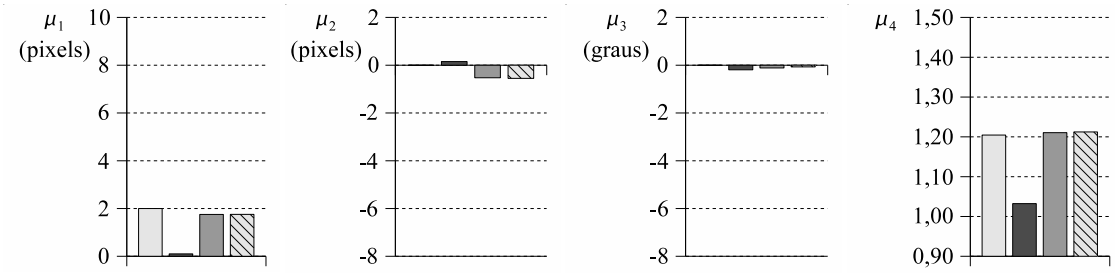

Imagem nº 12 (escalonamento do alvo - aproximação)

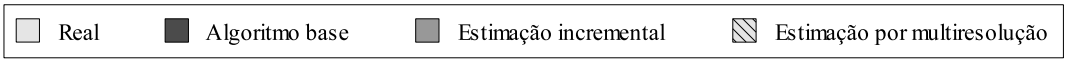

Figura 8.5: Valores de deslocamento obtidos no teste do filtro (continuação).

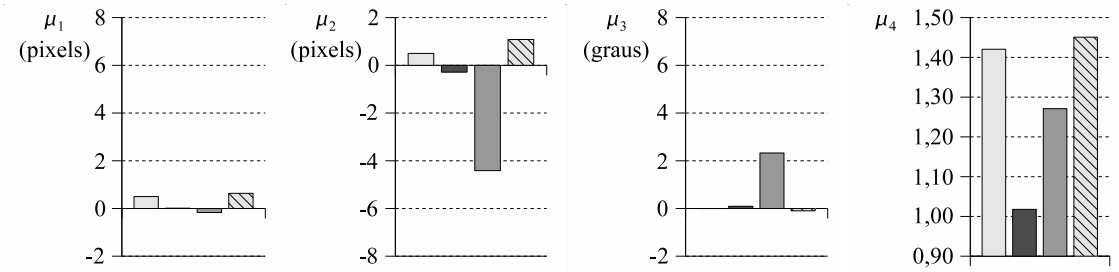

Imagem nº 13 (escalonamento do alvo - aproximação)

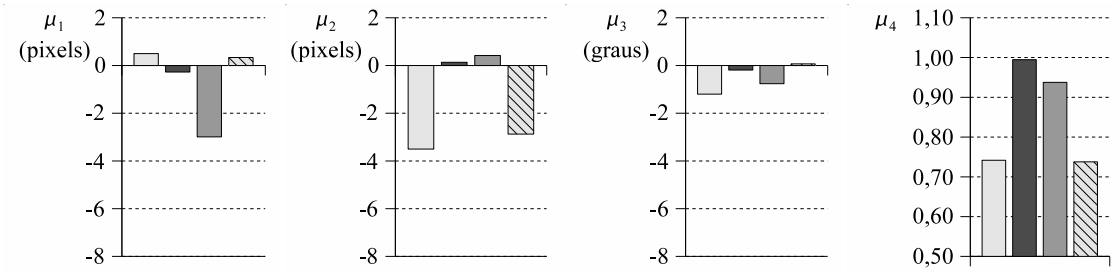

Imagem nº 14 (escalonamento do alvo - afastamento)

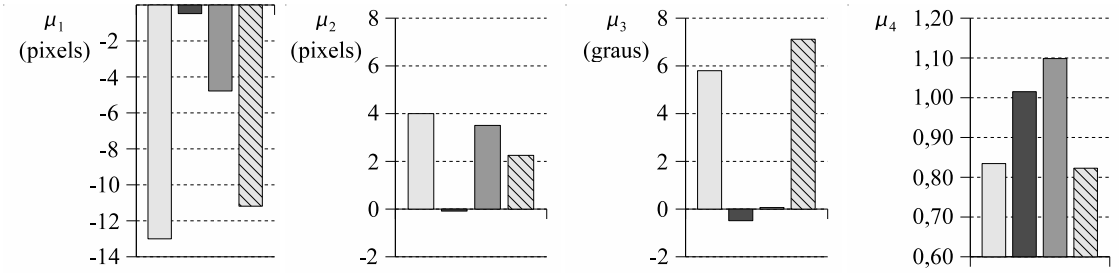

Imagem nº 15 (combinação de deslocamentos do alvo)

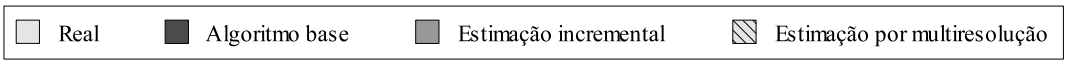

Figura 8.6: Valores de deslocamento obtidos no teste do filtro (continuação).

13).

Assim, conforme o esperado, o melhor desempenho obtido é o do algoritmo de multiresolução, seguido pelo de estimação incremental, para todos os tipos de movimento permitidos ao alvo. Para a imagem 15, em que são combinados deslocamentos de translação (cerca de 12 pixels), rotação (6 graus) e escalonamento (entre $0,8$ e $0,9$), o algoritmo de multiresolução também apresenta um resultado satisfatório. Além da análise comparativa, outro fator positivo do teste é a possibilidade de um vislumbre da capacidade de rastreamento em relação a escalo-

namentos e rotações, que não é intuitiva. Por exemplo, os fatos de uma rotação de cerca de 45 graus dada na imagem 08 e de uma considerável diferença visual de escalonamento dadas pelas imagens 13 e 14 serem estimadas corretamente pelo algoritmo de multiresolução não eram esperados.

Deve-se lembrar que o objetivo deste teste é realizar uma análise qualitativa, e não quantitativa, do alcance de cada algoritmo de rastreamento. Dessa maneira, os valores obtidos podem servir como estimativas da capacidade de cada procedimento, mas não como referências que indicam os valores máximos de deslocamento que o alvo pode apresentar. O desempenho quantitativo de cada algoritmo depende de fatores como as características do objeto a ser rastreado, do ambiente em que se encontra e da configuração de sua projeção no interior do padrão de referência escolhido.

8.2 Teste do Sistema

O objetivo deste teste é verificar o real comportamento do sistema como um todo. Para tanto, considera-se agora o sistema servo visual implementado de maneira completa. A entrada utilizada no teste é um deslocamento horizontal do alvo, simulando um pulso instantâneo (ou seja, de duração menor que um período de amostragem). São obtidas respostas do sistema para cada algoritmo de rastreamento e procedimento de controle apresentados, para que possam ser analisadas comparativamente.

Inicialmente, o teste consiste em movimentar a câmera pan-tilt, através do software implementado no computador, de modo a posicionar o alvo desejado no interior da região-alvo. Essa configuração é utilizada para a obtenção do padrão de referência. Uma amostra da imagem de referência contendo o alvo a ser rastreado é dada na Figura 8.7.

Após posicionar o equipamento devidamente, são selecionados o filtro e o controlador desejados e o modo de rastreamento é iniciado no software de controle. Em seguida, o sistema é submetido a um deslocamento horizontal do alvo em relação à câmera, de modo que é esperada uma reação do equipamento para que o alvo retorne à sua posição original na imagem. O deslocamento imposto ao alvo está ilustrado pela imagem da Figura 8.8, e vale cerca de 14 pixels para a esquerda. O procedimento é realizado para cada uma das seis combinações entre os três algoritmos de rastreamento e os dois de controle apresentados neste trabalho. Para cada combinação, são armazenados os valores do vetor de parâmetros $\mu(t_i)$ (deslocamentos horizontal, vertical, angular e de escala do alvo) estimados pelo filtro e da componente horizontal $u_1(t_i)$ do

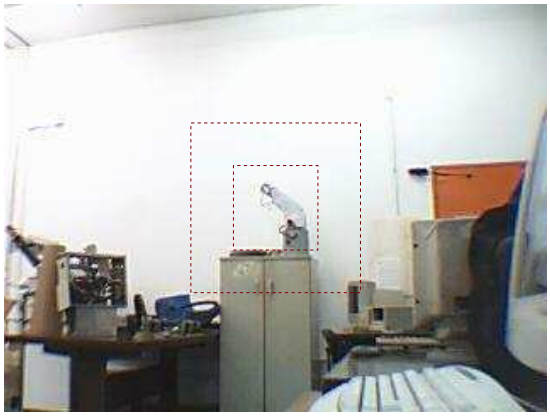

Figura 8.7: Imagem contendo uma amostra da região de referência utilizada no teste do sistema.

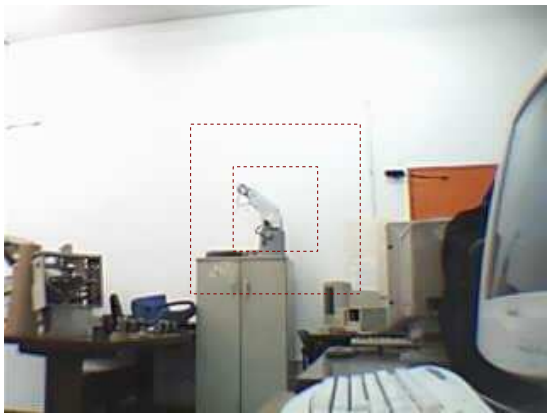

Figura 8.8: Imagem de amostra indicando o deslocamento imposto ao alvo no teste do sistema.

vetor de controle para cada instante de amostragem. Os gráficos correspondentes são dados nas Figuras 8.9 a 8.14.

De acordo com as Figuras 8.9 e 8.10, comprova-se que o algoritmo base (minimização do erro quadrático) não produz resultados satisfatórios. O menor valor de deslocamento que a câmera é capaz de compensar ($5,8$ pixels) ultrapassa a capacidade de detecção do algoritmo base, de modo que, mesmo com os procedimentos de controle utilizados, não foi possível estabilizar o alvo no centro da imagem. Os gráficos referentes aos deslocamentos angulares e de escala comprovam a instabilidade do sistema.

As Figuras 8.11 e 8.12 mostram os resultados para o software utilizando o algoritmo estendido por estimação incremental. Para o deslocamento imposto, os dois controladores foram capazes de trazer o alvo para o centro da imagem de maneira satisfatória. Percebe-se a presença de oscilações nas respostas, principalmente na Figura 8.11. Quanto ao tempo de resposta, após a detecção do deslocamento externo, o controlador LQG estabiliza a posição do alvo (descon-

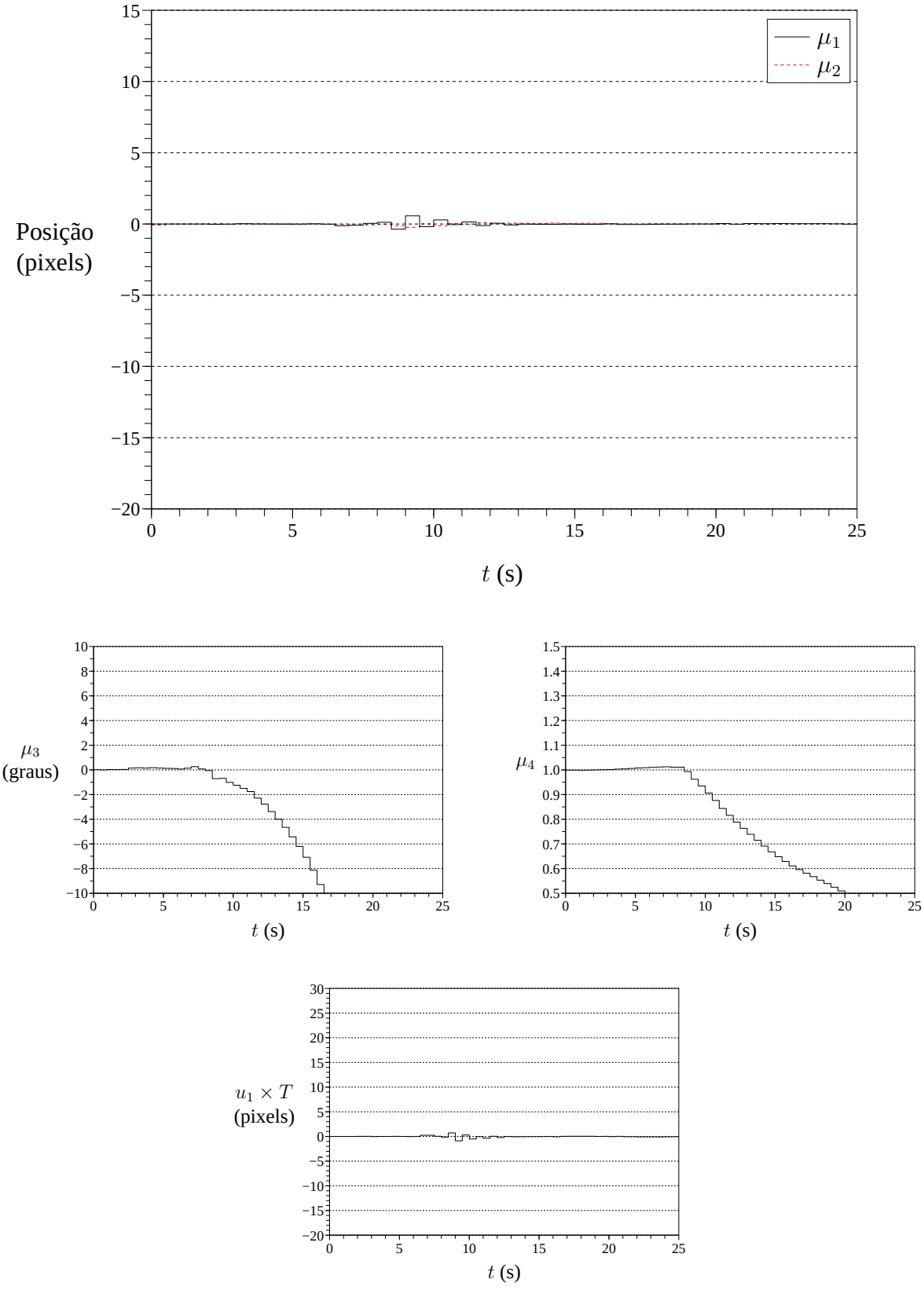

Figura 8.9: Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo base de rastreamento e o controlador PI.

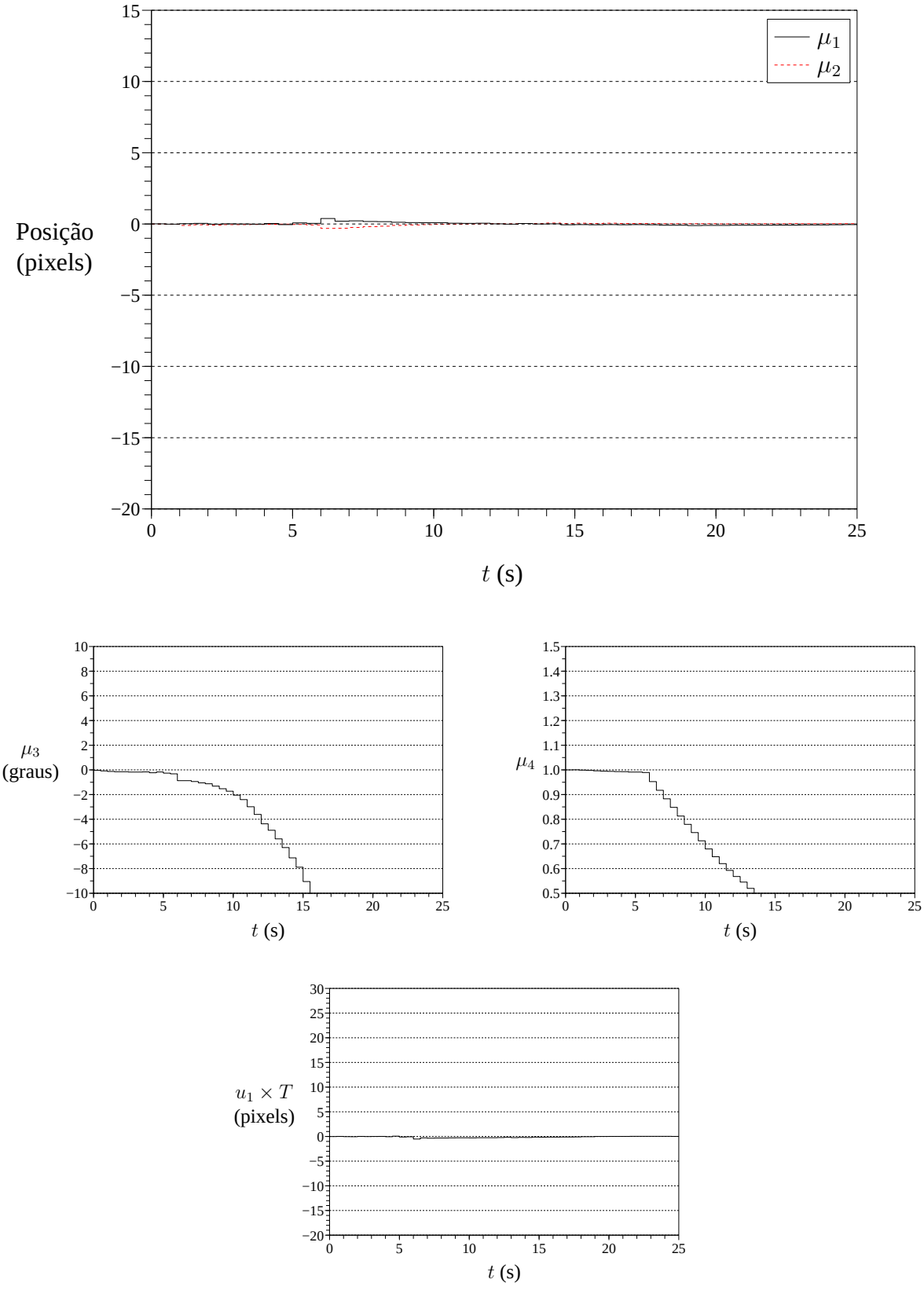

Figura 8.10: Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo base de rastreamento e o controlador LQG.

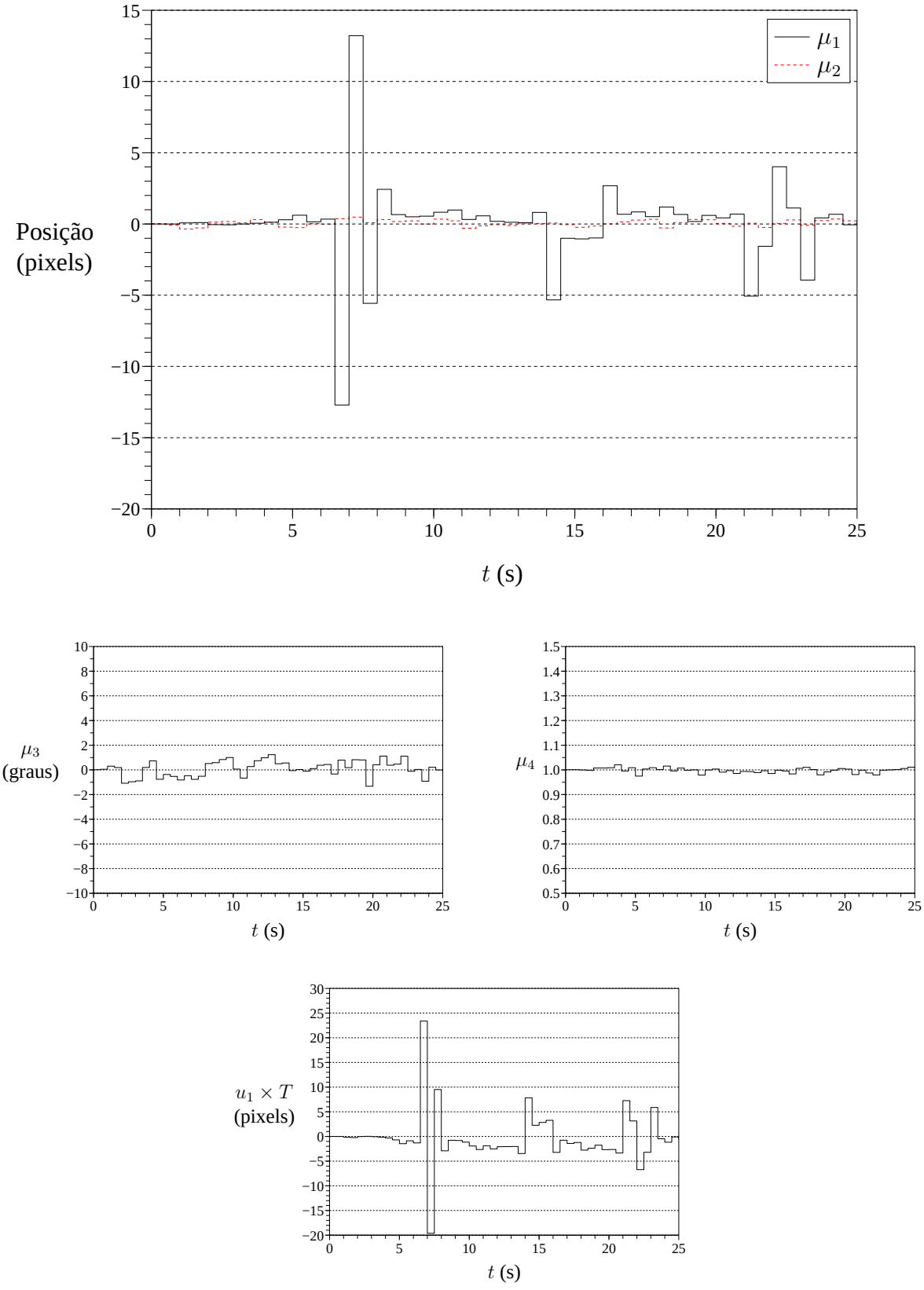

Figura 8.11: Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo de estimação incremental e o controlador PI.

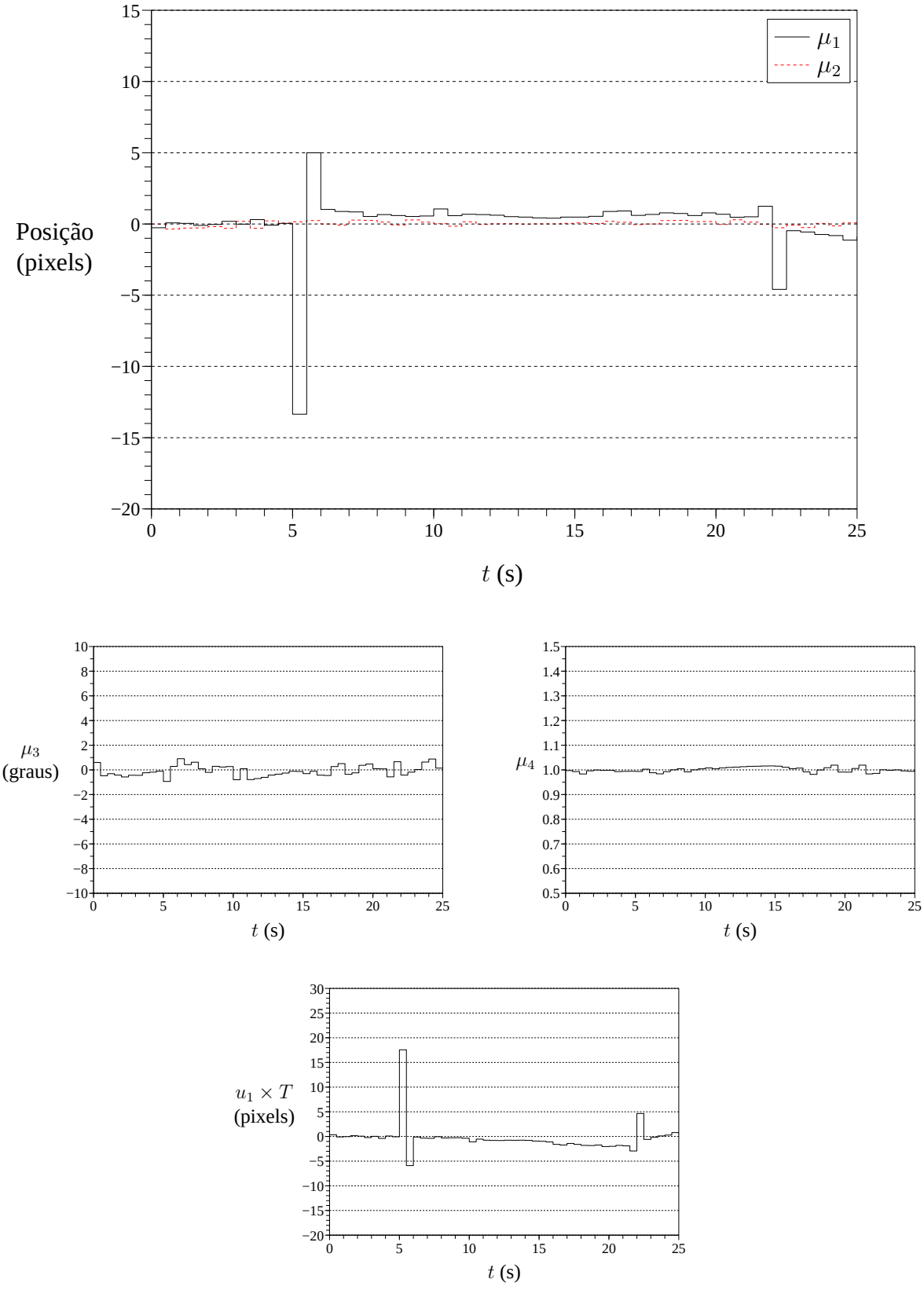

Figura 8.12: Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo de estimação incremental e o controlador LQG.

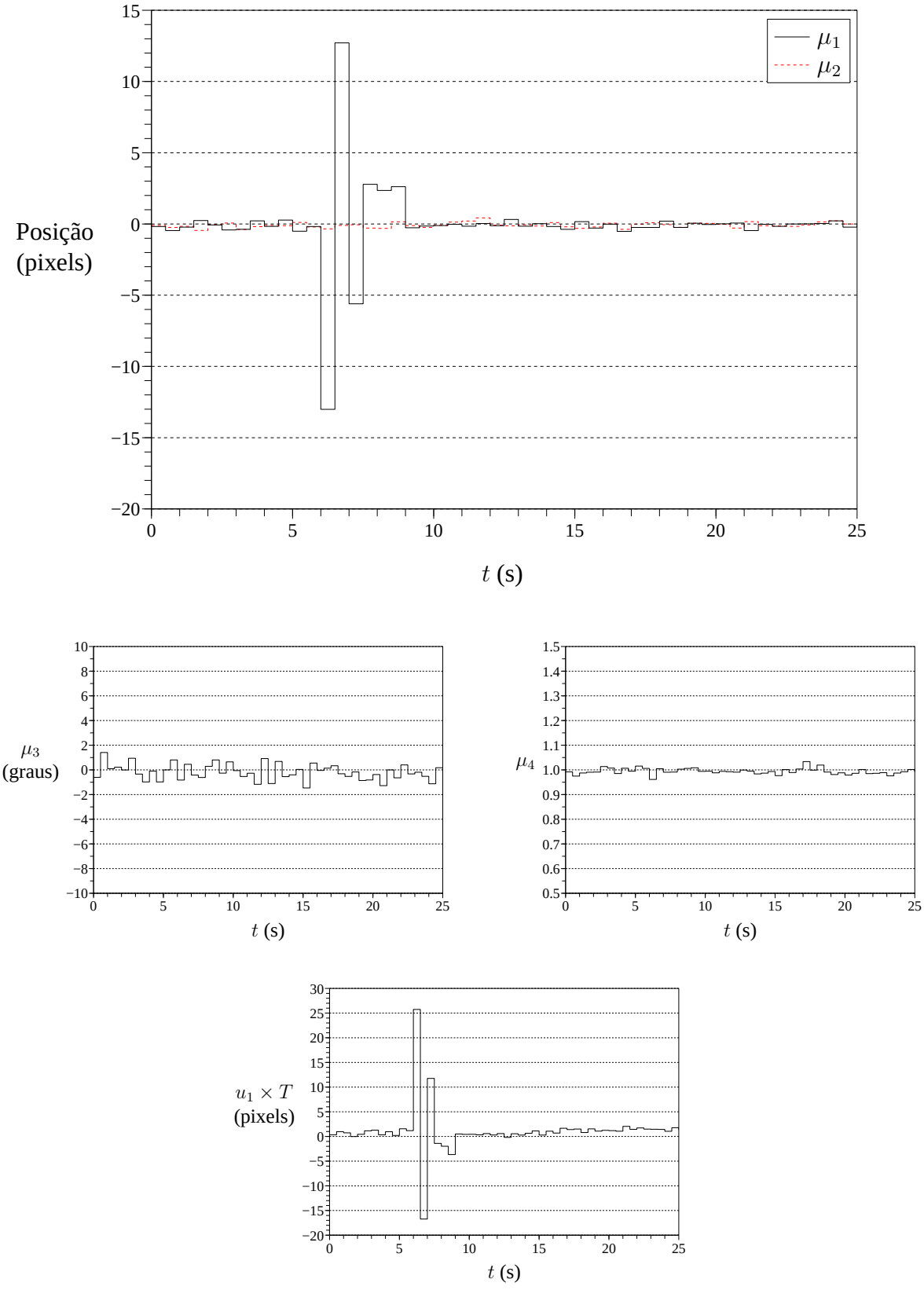

Figura 8.13: Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo de estimação em multiresolução e o controlador PI.

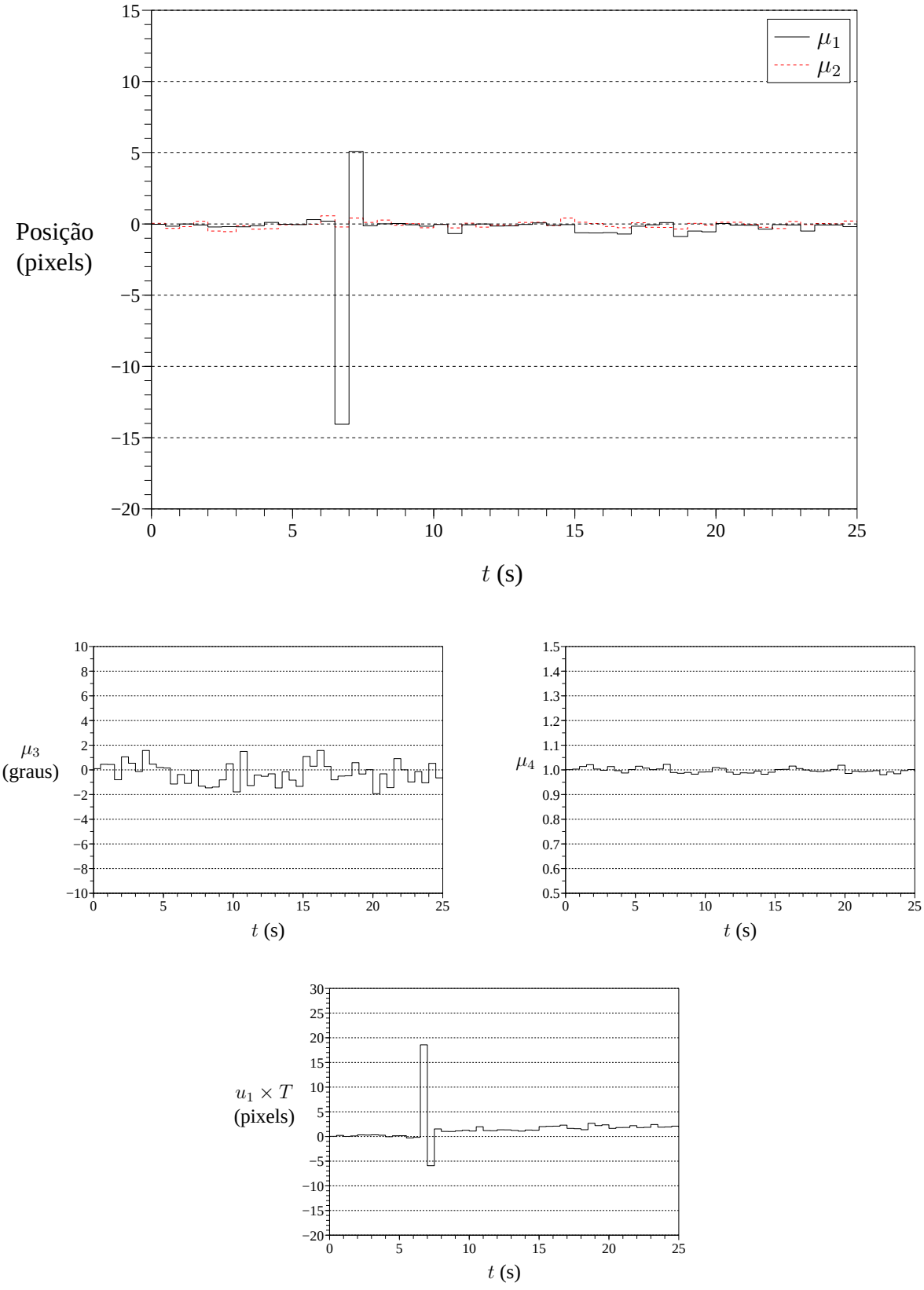

Figura 8.14: Gráficos de valores de deslocamento obtidos no teste do sistema utilizando o algoritmo de estimação em multiresolução e o controlador LQG.

siderando as oscilações posteriores) mais rapidamente que o PI. Além disso, para este último controlador, a resposta apresenta um pico de magnitude próxima à da perturbação, mas em sentido contrário, antes de se estabilizar. Isso se deve à característica do controlador PI de estabilizar o sistema apenas na presença de perturbações externas constantes. Neste teste, com uma entrada de pulso instantâneo, a situação é similar ao caso em que ocorre uma perturbação constante em um certo instante de tempo, e, no instante de amostragem seguinte, o sistema recebe uma nova perturbação constante, com mesma magnitude que a primeira mas sentido oposto. Assim, o controlador detecta duas perturbações consecutivas, fornecendo dois picos de resposta. Além disso, em relação às flutuações nos gráficos, a resposta do controlador PI oscila de maneira mais freqüente do que a do LQG.

Os gráficos das Figuras 8.13 e 8.14 mostram os resultados do sistema com o algoritmo estendido por estimação em multiresolução, que, sob alguns aspectos, são semelhantes aos do algoritmo de estimação incremental. O deslocamento imposto ao alvo foi também devidamente compensado, mas as oscilações das respostas se mostraram mais suaves. Em relação à velocidade das respostas, a do controlador LQG, novamente, se estabilizou mais rapidamente do que a do PI.

Embora sejam apresentados gráficos para somente uma situação de teste, deve-se ressaltar que o procedimento foi repetido por diversas vezes para cada combinação de algoritmos de rastreamento e controle. Estão apresentadas neste trabalho algumas amostras que representam os comportamentos mais comuns obtidos. Assim, deve ser ressaltado que as respostas do sistema com o algoritmo de estimação incremental, independentemente do controlador utilizado, tenderam a apresentar mais variações de comportamento em relação às do de multiresolução. Aliando o fato de as oscilações de suas respostas serem menos freqüentes, pode-se concluir que o algoritmo estendido por estimação em multiresolução é o mais preciso dos três, para o deslocamento imposto.

Em relação aos algoritmos de controle, embora o LQG tenha se mostrado mais rápido do que o PI em estabilizar o sistema, uma conclusão definitiva não pode ser realizada, já que o teste trata de apenas uma situação particular (perturbação em forma de pulso durante um instante de amostragem). Por outro lado, em relação às oscilações presentes nas respostas, percebe-se que as flutuações são mais suaves nos exemplos com o controlador LQG, como era esperado. Isso se reflete principalmente nos gráficos relativos ao algoritmo de estimação incremental, em que o nível de incerteza é superior ao do de multiresolução.

8.3 Teste do Sistema para um Alvo em Movimento

O teste anterior é baseado em um único deslocamento artificialmente imposto ao alvo durante todo o período considerado. Buscando realizar uma simulação mais próxima de uma situação real, este último teste tem como base o rastreamento de um alvo se movimentando de maneira mais freqüente e natural. Para acrescentar mais realismo ao experimento, o objeto de detecção utilizado consiste de uma face humana, alvo usualmente empregado em aplicações dessa natureza (como em videoconferências e ensino a distância, vide Seção 1.2). E como os dois testes anteriores já permitiram uma análise razoável da eficiência dos algoritmos de rastreamento visual no equipamento utilizado, este teste se concentra na comparação de desempenho entre os dois algoritmos de controle. Para tanto, considera-se fixo o procedimento de filtragem correspondente ao rastreamento em multiresolução, que apresentou uma performance superior nas seções anteriores.

O procedimento do teste consiste em, inicialmente, movimentar o dispositivo pan-tilt de modo a posicionar o rosto do indivíduo no centro da região-alvo da imagem. Em seguida, são selecionados os métodos de filtragem e controle e o modo de rastreamento é iniciado no software de controle. O movimento que a face realiza consiste em, inicialmente, permanecer imóvel durante alguns segundos. Depois disso, ocorre um deslocamento para a esquerda, seguido por um repouso de mais alguns segundos. Então, o alvo se desloca para uma posição à direita da inicial, repousa durante mais alguns segundos, e volta à posição inicial, onde permanece até o final do teste. Como o deslocamento do alvo é realizado por uma pessoa (na medida do possível, se movimentando naturalmente), a face apresenta também alguns deslocamentos verticais, de escalonamento e de rotação, inclusive durante os períodos de repouso. O procedimento foi repetido várias vezes para cada um dos controladores, sendo que dois resultados típicos estão reproduzidos nas Figuras 8.15 e 8.16 (gráficos de posição x_1 e x_2 em relação ao tempo decorrido).

É possível perceber nos gráficos três momentos em que picos de maior magnitude se destacam. Esses instantes correspondem obviamente aos três deslocamentos realizados pelo alvo. Um comportamento que se repete em relação ao teste anterior são as oscilações das respostas, mais freqüentes no sistema com controlador PI. Outra característica reincidente é o fato de o controlador PI apresentar picos de valores relativamente elevados no sentido contrário ao do deslocamento, quando o movimento do alvo cessa, enquanto que o LQG estabiliza a resposta mais rapidamente.

Um novo detalhe que pode ser percebido nos gráficos se refere aos instantes imediatamente

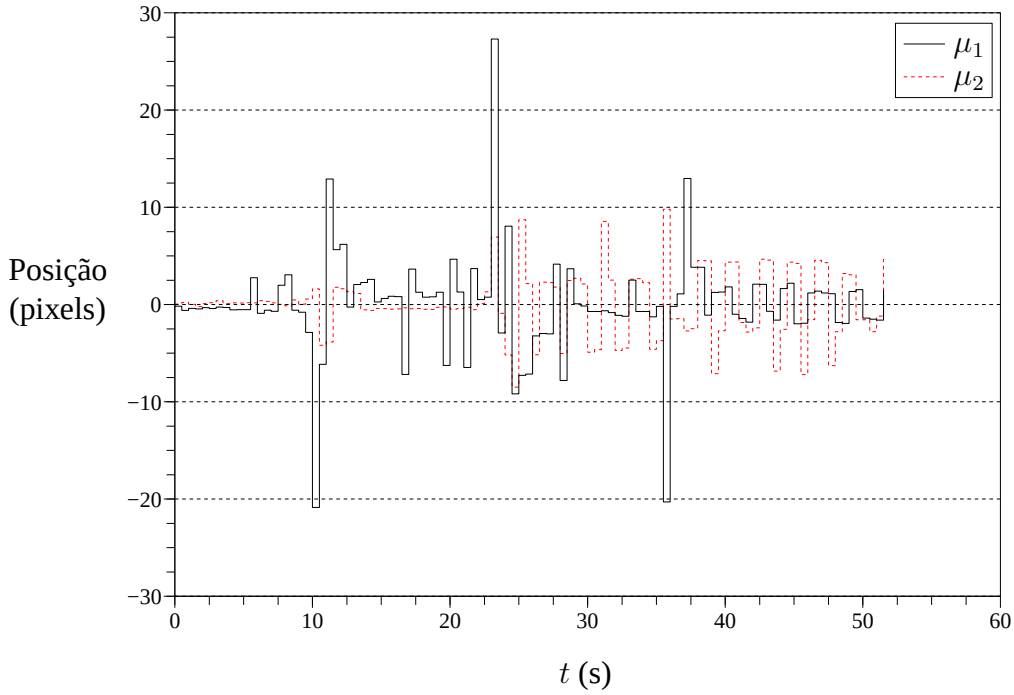

Figura 8.15: Gráficos de valores de deslocamento obtidos no teste do sistema para um alvo em movimento utilizando o controlador PI.

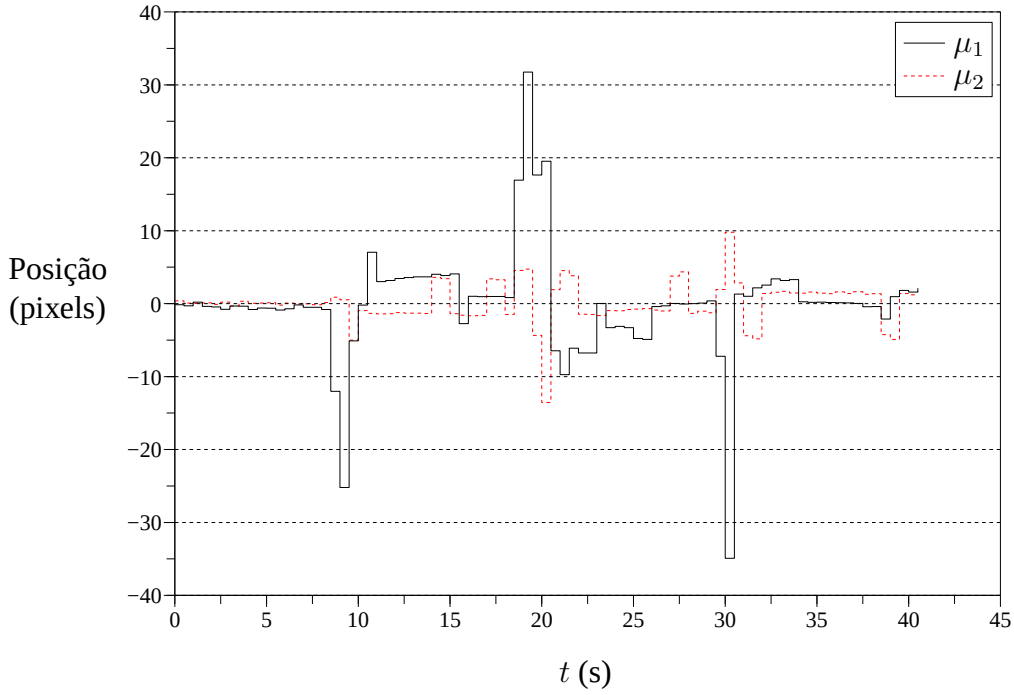

Figura 8.16: Gráficos de valores de deslocamento obtidos no teste do sistema para um alvo em movimento utilizando o controlador LQG.

após a detecção de um deslocamento do alvo. No caso do controle PI, o valor de deslocamento diminui, enquanto que no LQG tende a aumentar. Isso é justificado pela característica do controlador PI de compensar perturbações externas constantes. Assim, supondo que o alvo se desloca em velocidade constante, a resposta do sistema tende a ser mais satisfatória no caso do controlador PI, já que o LQG não se baseia neste caso particular. Entretanto, quando o movimento do alvo cessa, o desempenho do último se sobressai. Na Figura 8.15 se percebem essas tendências, principalmente durante o primeiro e o terceiro deslocamento. Quando se inicia o movimento do alvo, ocorre um pico na resposta. Em seguida, os deslocamentos relativos se tornam menores, correspondentes ao momento em que o alvo se encontra em movimento (constante). Por último, a resposta apresenta um pico em sentido contrário, o que caracteriza o fim do movimento do alvo.

Devido ao fato de apresentar respostas com oscilações mais suaves e picos de menor magnitude após o final dos movimentos, o controlador LQG se mostra mais eficiente do que o PI. O último se torna mais vantajoso em situações com movimentos em velocidade constante, mas é uma característica particular que não pode ser considerada como corriqueira no tipo de aplicação deste trabalho.

9 CONCLUSÕES

Este trabalho apresentou algumas técnicas de controle servo visual aplicadas em uma câmera pan-tilt. A solução foi dividida em duas partes principais. A primeira consistiu no desenvolvimento de um algoritmo para rastreamento visual de um alvo, o que foi denominado como filtro. Já a segunda etapa foi dada pela técnica de controle propriamente dito, produzindo o controlador responsável por utilizar as informações visuais na realimentação do sistema.

Após o estudo de técnicas de rastreamento visual, selecionou-se para este trabalho um algoritmo de minimização do erro quadrático. Esse procedimento se baseia em um rastreamento de uma região de referência que utiliza a SSD como critério de correlação. Definiu-se também que os tipos de deslocamento do alvo que o algoritmo deve ser capaz de detectar são translações, rotações e escalonamentos. Contudo, os testes realizados mostraram que o desempenho do algoritmo no equipamento utilizado não é satisfatório, justificando as modificações efetuadas sobre ele. A primeira consistiu na utilização de uma extensão baseada em um princípio de estimação incremental, que produziu resultados satisfatórios. Sobre esta última, uma nova extensão foi acrescentada, baseada no princípio de multiresolução. Dessa maneira, o algoritmo tornou-se ainda mais eficiente, possibilitando maior alcance e maior precisão na detecção de deslocamentos de um alvo.

Em relação ao controle servo visual propriamente dito, foi selecionada para este trabalho uma técnica de controle que utiliza um modelo simples e linear. Esse modelo tem como base o princípio de fluxo óptico e utiliza um jacobiano de imagem de valor fixo, considerando uma projeção perspectiva do objeto rastreado sobre a imagem. Foram desenvolvidos dois controladores para o modelo: um proporcional integral (PI) e um proporcional com estimação de perturbações externas através de um filtro de Kalman (LQG). Ambos foram calculados utilizando um critério de minimização linear quadrático. O desempenho do controlador LQG se mostrou superior nos testes, apresentando respostas com menos oscilações e picos mais suaves, principalmente em situações com maior variação de tipos de perturbação sobre o sistema.

Considerando o desempenho do equipamento com o algoritmo em sua configuração mais eficiente (portanto, com o algoritmo de multiresolução e o controlador LQG), pode-se dizer que ele é satisfatório para as típicas aplicações que utilizam uma câmera pan-tilt, desde que respeitadas suas limitações. Durante o teste do filtro, o algoritmo de multiresolução foi capaz de detectar deslocamentos por volta de 12 pixels, mas, no teste do sistema para um alvo em movimento, deslocamentos de 30 pixels em um período de amostragem de $0,5$ s foram rastreados com sucesso. Alvos com maiores velocidades provavelmente podem ser rastreados com

freqüências de amostragem mais altas, mas um hardware mais potente também seria necessário (não só para proporcionar maiores velocidades de captação de imagens, como também respostas de posicionamento pan-tilt mais rápidas). Neste caso, deve-se levar em conta que, com menores períodos de amostragem, o alcance do algoritmo de rastreamento não precisa ser muito elevado. Além disso, o alcance do rastreamento está diretamente relacionado com o tempo de cálculo gasto (como estimativas, os cálculos do sistema nos testes com o algoritmo base consumiam até aproximadamente 40 ms por iteração, com o de estimação incremental cerca de 80 ms e com o de multiresolução por volta de 100 ms). Por outro lado, o tempo de cálculo também diminui com a utilização de um hardware mais avançado. Portanto, neste caso, há várias variáveis a serem analisadas para a escolha do melhor procedimento de filtragem (algoritmo base, de estimação incremental ou de multiresolução, lembrando que a utilização do algoritmo base depende também de um dispositivo pan-tilt com maior resolução que a do utilizado neste trabalho).

Vale também ressaltar que, para um objeto se movimentando no espaço, sua velocidade projetada na imagem se torna menor conforme a distância entre o objeto e a câmera aumenta $\big($vide equação (4.1.6) ou (4.1.7)$\big)$. Dessa maneira, um posicionamento adequado do equipamento é importante, assim como a escolha do padrão de referência. Contudo, há situações em que a distância entre o objeto a ser rastreado e a câmera não pode ser modificada, seja por motivos técnicos (como no caso de uma câmera com base fixa) ou pelo fato de a região-alvo não enquadrar adequadamente o objeto se muito distante. Nesse caso, uma opção razoável é iniciar o rastreamento em uma distância mais próxima e, em seguida, transportar a câmera (ou o alvo) para sua posição de trabalho, deixando a cargo do algoritmo de filtragem realizar a compensação de escala proveniente do afastamento.

Outra maneira de permitir maiores velocidades ao alvo é ampliar o alcance do filtro através de um aumento do tamanho da região-alvo ou ainda com a adição de mais um passo de multiresolução. Entretanto, isso acarreta em um maior tempo de processamento do algoritmo, o que remete à questão do hardware utilizado. Além disso, um aumento relativo no tamanho da região de rastreamento em relação às dimensões do alvo proporciona um crescimento no tamanho do plano de fundo da imagem. Alguns testes realizados mostraram que, quanto mais detalhes particulares o fundo de imagem apresenta, pior é a performance do algoritmo (assim como uma maior quantidade de detalhes do alvo possibilita um desempenho superior, o que remete à questão de escolha adequada do padrão de referência). Assim, se o fundo (ambiente em que se encontra o objeto rastreado) apresenta muitos detalhes (e que variam conforme o deslocamento da câmera), a performance do algoritmo pode se comprometer. Em relação à adição de um novo passo de multiresolução, outro problema que prejudicaria o rastreamento é relaci-

onado à diminuição do nível de detalhamento (resolução) da imagem que ocorre a cada novo passo utilizado. Uma solução possível para isso seria utilizar uma câmera com maior resolução.

No trabalho original de onde foi extraída a técnica de minimização do erro quadrático (HAGER; BELHUMEUR, 1998), há ainda dois modelos adicionados sobre a função SSD que não foram utilizados, compensando situações não consideradas neste trabalho. Eles correspondem a um modelo para variação de iluminação e outro de estimação robusta para o caso em que ocorre oclusão do alvo. A adição desses procedimentos provavelmente melhoraria o desempenho do algoritmo, principalmente em ambientes que apresentam grandes variações em suas condições. Em relação aos procedimentos de controle, o modelo utilizado é linear, de modo que outros tipos de controlador podem ser desenvolvidos e facilmente implementados. Dessa maneira, o algoritmo desenvolvido neste trabalho possibilita uma abertura em que possíveis melhorias podem ser implementadas sem modificações drásticas e grandes dificuldades.

Referências Bibliográficas

ABDEL-HAKIM, A. E.; FARAG, A. A. Robust virtual forces-based camera positioning using a fusion of image content and intrinsic parameters. In: INTERNATIONAL CONFERENCE ON INFORMATION FUSION, 2005, Filadélfia, EUA. **2005 8th International Conference on Information Fusion**. EUA: Institute of Electrical and Electronics Engineers, Inc., 2005. v. 1, p. 522–529.

AXIS COMMUNICATIONS. **Technical Guide to Network Video**. [S.l.], 2006.

BROOKES, M. **The matrix reference manual**. Londres, Reino Unido, 2005. Disponível em: <http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/intro.html>. Acesso em: 18 jan. 2007.

CHAUMETTE, F.; RIVES, P.; ESPIAU, B. Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1991, Sacramento, EUA. **Proceedings of the 1991 IEEE International Conference on Robotics and Automation**. EUA: Institute of Electrical and Electronics Engineers, Inc., 1991. v. 3, p. 2248–2253.

CHEN, K.-Y.; CHENG, M.-Y.; TSAI, M.-C. Design and implementation of a real-time pan-tilt visual tracking system. In: INTERNATIONAL CONFERENCE ON CONTROL APPLICATIONS, 11., 2002, Glasgow, Escócia, Reino Unido. **Proceedings of the 2002 IEEE International Conference on Control Applications**. EUA: Institute of Electrical and Electronics Engineers, Inc., 2002. v. 2, p. 736–741.

CLADY, X.; COLLANGE, F.; JURIE, F.; MARTINET, P. Object tracking with a pan-tilt-zoom camera: application to car driving assistance. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 18., 2001, Seul, Coréia. **Proceedings 1998 ICRA IEEE International Conference on Robotics and Automation**. Coréia: Kyung Hee Information Printing Co., Ltd., 2001. v. 2, p. 1653–1658.

CORKE, P. I. Visual control of robot manipulators – a review. In: HASHIMOTO, K. (Ed.). **Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback**. [S.l.]: World Scientific, 1994, (Robotics and Automated Systems, v. 7). p. 1–31.

CRÉTUAL, A.; CHAUMETTE, F. Dynamic stabilization of a pan and tilt camera for submarine image visualization. **Computer Vision and Image Understanding**, v. 79, n. 1, p. 47–65, jul. 2000.

CUBBER, G. D.; BERRABAH, S. A.; SAHLI, H. Color-based visual servoing under varying illumination conditions. **Robotics and Autonomous Systems**, Países Baixos, v. 47, n. 4, p. 225–249, jul. 2004.

ESPIAU, B.; CHAUMETTE, F.; RIVES, P. A new approach to visual servoing in robotics. **IEEE Transactions on Robotics and Automation**, EUA, v. 8, n. 3, p. 313–326, jun. 1992.

FAYMAN, J. A.; SUDARSKY, O.; RIVLIN, E. Zoom tracking. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 15., 1998, Leuven, Bélgica. **Proceedings 1998 IEEE International Conference on Robotics and Automation**. EUA: Omnipress, 1998. v. 4, p. 2783–2788.

FLEURY, A. de T. **Filtro de Kalman aplicado à navegação aérea**. [S.l.], 1981/1982. (Notas de Aula do Curso Oferecido à EMBRAER, INPE).

GHIDARY, S. S.; NAKATA, Y.; TAKAMORI, T.; HATTORI, M. Human detection and localization at indoor environment by home robot. In: INTERNATIONAL CONFERENCE ON SYSTEMS, MAN, AND CYBERNETICS, 13., 2000, Nashville, EUA. **SMC 2000 Conference Proceedings**. EUA: Institute of Electrical and Electronics Engineers, Inc., 2000. v. 2, p. 1360–1365.

GRASSI JUNIOR, V. **Sistema de Visão Omnidirecional Aplicado no Controle de Robôs Móveis**. 2002. 85 p. Tese (Mestrado) — Escola Politécnica, Universidade de São Paulo, São Paulo, SP, 2002.

HAGER, G. D.; BELHUMEUR, P. N. Efficient region tracking with parametric models of geometry and illumination. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, EUA, v. 20, n. 10, p. 1025–1039, out. 1998.

HARITAOGLU, I.; HARWOOD, D.; DAVIS, L. S. Active outdoor surveillance. In: INTERNATIONAL CONFERENCE ON IMAGE ANALYSIS AND PROCESSING, 10., 1999, Veneza, Itália. **ICIAP'99**. EUA: The Printing House, 1999. p. 1096–1099.

HUANG, Y.-W.; HSIEH, B.-Y.; CHIEN, S.-Y.; CHEN, L.-G. Simple and effective algorithm for automatic tracking of a single object using a pan-tilt-zoom camera. In: INTERNATIONAL CONFERENCE ON MULTIMEDIA AND EXPO, 2002, Lausanne, Suíça. **ICME2002**. EUA: Institute of Electrical and Electronics Engineers, Inc., 2002. v. 1, p. 789–792.

HUTCHINSON, S. A.; HAGER, G. D.; CORKE, P. I. A tutorial on visual servo control. **IEEE**

Transactions on Robotics and Automation, EUA, v. 12, n. 5, p. 651–670, out. 1996.

JUNG, B.; SUKHATME, G. S. Detecting moving objects using a single camera on a mobile robot in an outdoor environment. In: INTERNATIONAL CONFERENCE ON INTELLIGENT AUTONOMOUS SYSTEMS, 8., 2004, Amsterdã, Países Baixos. **IAS'04**. Amsterdã: IOS Press, 2004. p. 980–987.

JURIE, F.; DHOME, M. A simple and efficient template matching algorithm. In: INTERNATIONAL CONFERENCE ON COMPUTER VISION, 8., 2001, Vancouver, Canadá. **Proceedings Eighth IEEE International Conference on Computer Vision ICCV 2001**. EUA: The Printing House, 2001. v. 2, p. 544–549.

KLEIN, R. de. **Serial Communication Library for C++**. 2004. Documentação da biblioteca de comunicação serial. Disponível em: <http://home.ict.nl/~ramklein/Projects/Serial.html>. Acesso em: 18 jan. 2007.

LOWE, D. G. Object recognition from local scale-invariant features. In: INTERNATIONAL CONFERENCE ON COMPUTER VISION, 1999, Corfu, Grécia. **ICCV'99**. EUA: Institute of Electrical and Electronics Engineers, Inc., 1999. v. 2, p. 1150–1157.

MATTHYS, D. R. **Digital image processing**: Color models. Milwaukee, EUA, 2001. (Notas de aula de disciplina oferecida pela Marquette University). Disponível em: <http://academic.mu.edu/phys/matthysd/web226/L0221.htm>. Acesso em: 18 jan. 2007.

MAYBECK, P. S. **Stochastic models, estimation, and control**. Nova Iorque, EUA: Academic Press, 1979. (Mathematics in Science and Engineering, v. 3).

MAYBECK, P. S. **Stochastic models, estimation, and control**. Nova Iorque, EUA: Academic Press, 1979. (Mathematics in Science and Engineering, v. 1).

MURRAY, D.; BASU, A. Motion tracking with an active camera. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, EUA, v. 16, n. 5, p. 449–459, mai. 1994.

ODOBEZ, J. M.; BOUTHEMY, P. Robust multiresolution estimation of parametric motion models. **International Journal of Visual Communication and Image Representation**, v. 6, n. 4, p. 348–365, dez. 1995.

PAPANIKOLOPOULOS, N.; KHOSLA, P.; KANADE, T. Visual tracking of a moving target

by a camera mounted on a robot: a combination of control and vision. **IEEE Transactions on Robotics and Automation**, EUA, v. 9, n. 1, p. 14–35, fev. 1993.

PIEPMEIER, J. A.; MCMURRAY, G. V.; LIPKIN, H. A dynamic quasi-newton method for uncalibrated visual servoing. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1999, Detroit, EUA. **ICRA99 Proceedings**. EUA: Omnipress, 1999. v. 2, p. 1595–1600.

SHENZHEN AKKORD ELECTRONICS CO., LTD. **Shenzhen Akkord Electronics Co., Ltd. — Product Details**. 2002–2004. Informações da webcam utilizada no trabalho. Disponível em: <http://www.akkord.com.cn/en/product_details.asp?id=232>. Acesso em: 5 set. 2006.

SILVEIRA FILHO, G. F. **Controle Servo Visual de Veículos Robóticos Aéreos**. 2002. Tese (Mestrado) — Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas, Campinas, SP, 2002.

SIVASAGAR, K. R. **CodeGuru: Simultaneous Previewing & Video Capture using DirectShow**. 2004. Documentação da biblioteca de captura de imagens. Disponível em: <http://www.codeguru.com/Cpp/G-M/multimedia/video/article.php/c6973/>. Acesso em: 18 jan. 2007.

SURYADARMA, S.; ADIONO, T.; MACHBUB, C.; MENGKO, T. L. R. Camera object tracking system. In: INTERNATIONAL CONFERENCE ON INFORMATION, COMMUNICATIONS AND SIGNAL PROCESSING, 1997, Singapura. **1997 1st International Conference on Information, Communications and Signal Processing Proceedings**. EUA: Institute of Electrical and Electronics Engineers, Inc., 1997. v. 3, p. 1557–1561.

TSAKIRIS, D. P. **Visual tracking strategies**. 1988. 94 p. Tese (Mestrado) — Department of Electrical Engineering, University of Maryland, EUA, 1988.

XU, G.; SUGIMOTO, T. Rits eye: a software-based system for realtime face detection and tracking using pan-tilt-zoom controllable camera. In: INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION, 14., 1998, Brisbane, Austrália. **ICPR'98**. EUA: The Printing House, 1998. v. 2, p. 1194–1197.

YACHI, K.; WADA, T.; MATSUYAMA, T. Human head tracking using adaptive appearance models with a fixed-viewpoint pan-tilt-zoom camera. In: INTERNATIONAL CONFERENCE ON AUTOMATIC FACE AND GESTURE RECOGNITION, 4., 2000, Grenoble, França. **Proceedings Fourth IEEE International Conference on Automatic Face and Gesture**

Recognition. EUA: The Printing House, 2000. p. 150–155.

YE, Y.; TSOTSOS, J. K.; HARLEY, E.; BENNET, K. Tracking a person with pre-recorded image database and a pan, tilt, and zoom camera. **Machine Vision and Applications**, Berlin, Alemanha, v. 12, n. 1, p. 32–43, 2000.

YOSHIMI, B. H.; ALLEN, P. K. Active, uncalibrated visual servoing. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1994, San Diego, EUA. **Proceedings ICRA 1995**. EUA: Institute of Electrical and Electronics Engineers, Inc., 1994. v. 1, p. 156–161.

YOUNG, I. T.; GERBRANDS, J. J.; VLIET, L. J. van. **Image processing fundamentals**: Derivative-based operations. Delft, Países Baixos, 1997. Disponível em: <http://www.ph.tn.tudelft.nl/Courses/FIP/noframes/fip-Derivati.html>. Acesso em: 18 jan. 2007.

APÊNDICE A – Cálculo da distância focal da câmera

Para determinar o valor da distância focal da câmera, são utilizados a seguir dois procedimentos: no primeiro, a distância é calculada com base nos parâmetros da câmera, e, no segundo, ela é determinada empiricamente através de um experimento. O cálculo empírico permite verificar na prática se o modelo "pin-hole" e a hipótese de um valor constante para a distância focal, utilizados no primeiro procedimento, são razoáveis.

Cálculo baseado nos parâmetros da câmera

Segundo o fabricante da câmera (SHENZHEN AKKORD ELECTRONICS CO., LTD., 2002–2004), a distância focal da lente da câmera é igual a $3,6$ mm. Contudo, o valor utilizado no jacobiano de imagem deve ser fornecido em pixels. A conversão necessária entre unidades é possível através de uma simples regra de três. Para tanto, o outro parâmetro a ser considerado consiste na largura (tamanho horizontal) da imagem. Ainda de acordo com Shenzhen Akkord Electronics Co., Ltd. (2002–2004), o tamanho do sensor da câmera que captura a imagem é de $1/4$ de polegada. Mas este valor corresponde ao comprimento da diagonal do sensor. Segundo Axis Communications (2006), a largura correspondente de um sensor de $1/4$" é igual a $3,6$ mm. Dado que a largura da imagem é de 320 pixels, obtém-se:

$$\frac{\text{Dist. focal em pixels}}{\text{Dist. focal em mm}} = \frac{\text{Largura da imagem em pixels}}{\text{Largura da imagem em mm}} \Rightarrow \frac{f}{3,6} = \frac{320}{3,6} \Rightarrow f = 320 \text{ pixels}$$

Cálculo baseado em dados empíricos

Este cálculo é baseado na equação (4.1.4) $\big($ou (4.1.5)$\big)$. De acordo com esta equação, se as distâncias X (ou Y) e Z de um objeto forem conhecidas, a distância focal pode ser calculada a partir da posição x_1 (ou x_2) da imagem projetada.

O experimento realizado é baseado no esquema da Figura A.1. Ele consiste no uso de objetos retangulares com dimensões (largura L_1 e altura L_2 constantes) conhecidas. Os objetos são posicionados perpendicularmente sobre o eixo óptico da câmera e situados a algumas distâncias conhecidas (D variável). Para cada distância, é adquirida uma imagem e, analisando-as, é possível estimar a largura (l_1 variável) e a altura (l_2 variável) dos objetos projetadas em cada imagem. Assim, é possível determinar valores de distância focal para cada combinação.

Dois objetos foram utilizados no experimento: o primeiro possui dimensões $L_1 = 24$ mm e $L_2 = 41$ mm, enquanto que o segundo possui $L_1 = 780$ mm e $L_2 = 1500$ mm. O primeiro objeto foi disposto em três posições diferentes ($D = 110$; 200 e 300 mm), e o segundo em duas

($D = 5620$ e 4420 mm). Os valores de distância focal obtidos são dados na Tabela A.1.

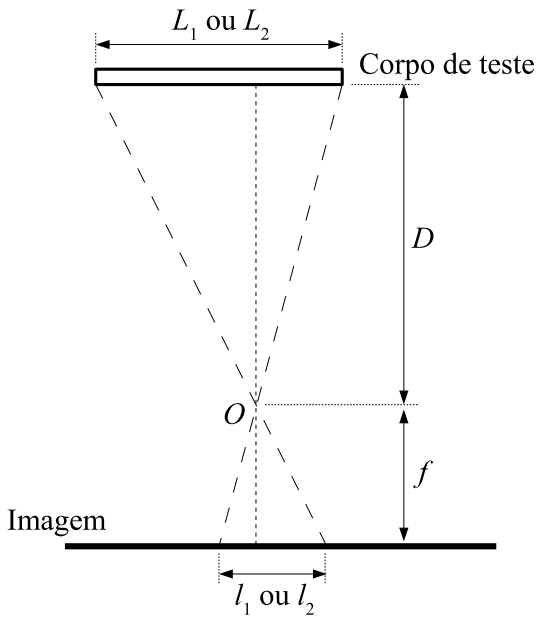

Figura A.1: Esquema das distâncias envolvidas no experimento de determinação do valor da distância focal da câmera.

Tabela A.1: Valores de distância focal obtidos através do experimento.

L_1 (mm)	L_2 (mm)	D (mm)	l_1 (pixels)	l_2 (pixels)	$f = \frac{l_1}{L_1} D$ (pixels)	$f = \frac{l_2}{L_2} D$ (pixels)
24	41	110	70	117	320,83	313,90
24	41	200	41	66	341,67	321,95
24	41	300	28	44	350,00	321,95
780	1500	5620	47	91	338,64	340,95
780	1500	4420	59	114	334,33	335,92
					$\bar{f} = 332,01$	$\sigma_f = 11,63$

O valor de desvio padrão obtido é de $3,5$ % do valor médio da distância focal. Considerando o modelo “pin-hole” e os fatos de que a medição das distâncias e o posicionamento do aparato foram realizados com instrumentos sem grande precisão, o desvio é satisfatório. Comparando a média com o valor obtido no método anterior, o erro relativo é de $3,75$ %, que é um resultado razoável. No Apêndice B, a influência desse erro sobre a tarefa de posicionamento da câmera é analisada.

APÊNDICE B – Limitações de movimento do alvo e influência de aproximações realizadas

A partir do equipamento utilizado neste trabalho e suas limitações, é possível estimar qual a velocidade máxima com que o alvo pode se deslocar de modo que os algoritmos de rastreamento tenham a possibilidade de detectá-lo. Esse dado pode servir como base de comparação para analisar o desempenho dos algoritmos em testes. Além disso, é possível estimar também outras grandezas relacionadas do sistema e verificar sua conformidade com as hipóteses adotadas.

Velocidade máxima permitida ao alvo

Inicialmente, deve-se definir qual o deslocamento máximo que o alvo pode apresentar entre imagens adquiridas em dois instantes de amostragem consecutivos. Para tanto, é necessário considerar o tamanho da região-alvo (janela) utilizada. Após o alvo ser localizado em uma determinada posição de uma imagem, o rastreamento na imagem adquirida no instante seguinte é realizado a partir de uma janela centralizada nessa posição. Se o alvo apresentar um deslocamento entre estes dois instantes que o situe fora da janela considerada, não há informação disponível sobre o alvo para que o algoritmo utilizado realize o rastreamento de maneira efetiva.

Assim, o deslocamento máximo considerado para o alvo é o que o desloca para o limite da janela, ou seja, um deslocamento de valor igual à metade do tamanho da região-alvo. Entretanto, duas considerações devem ser realizadas. A primeira, é relacionada ao tamanho do alvo: se o alvo fosse um ponto, o deslocamento considerado o posicionaria exatamente na fronteira da janela. Mas o alvo não é adimensional, de modo que o deslocamento de metade do tamanho da região-alvo posicionaria o seu centróide sobre a fronteira. Assim, uma parte do alvo estaria localizada fora da janela. Contudo, como as dimensões do alvo não são conhecidas, elas são desprezadas para simplificar as estimativas e o deslocamento considerado é o utilizado nos cálculos a seguir. A outra consideração refere-se ao algoritmo de multiresolução. Como explicado na Seção 5.3, a grosso modo, o alcance do algoritmo é dobrado devido a essa extensão. Como o objetivo da análise é estimar o deslocamento máximo permitido ao alvo, deve-se levar em conta a extensão por multiresolução e considerar a estimativa duas vezes maior. Assim, o valor do deslocamento máximo se torna igual ao tamanho da região-alvo.

Para determinar a velocidade máxima permitida ao alvo basta dividir o deslocamento máximo permitido entre dois instantes de aquisição consecutivos pelo período de amostragem. Dado que o tamanho da região alvo é igual a 50 pixels e o período de amostragem é de $0,5$ s:

$$\text{Deslocamento máximo do alvo durante um período de amostragem} = 50 \text{ pixels}$$

$$\text{Velocidade máxima do alvo} = 50 \div 0,5 = 100 \text{ pixels/s}$$

Vale lembrar que, como a região-alvo é quadrada, essa velocidade se refere tanto à componente do eixo x_1 (horizontal) quanto do eixo x_2 (vertical) do sistema de coordenadas da imagem.

Variação da rotação da câmera devido à mudança de posição do alvo

De acordo com o Capítulo 4, para aplicar os sinais de controle u_1 e u_2 corretamente sobre o sistema, é necessário que as rotações pan e tilt sejam alteradas conforme a câmera se movimenta. Assim, a análise a seguir visa mostrar o efeito da aproximação realizada de considerar as rotações constantes durante cada período de amostragem.

Toma-se, como exemplo, a rotação R_X. De acordo com a equação (4.1.8), ela é calculada através dos valores de f, u_1, u_2, x_1 e x_2. O valor da distância focal é fixo e os sinais de controle são constantes durante cada período de amostragem. Assim, a análise consiste em verificar qual a maior alteração no valor de R_X considerando as possíveis posições (x_1, x_2) do alvo, de modo a manter os valores de u_1 e u_2 constantes. A distância focal utilizada corresponde ao valor empírico $f = 332,01$ pixels. Em relação às variáveis de controle, são consideradas estimativas de seus valores máximos. Admite-se que estes valores correspondem à velocidade máxima do alvo, ou seja, que o maior movimento que a câmera realiza corresponde ao maior movimento permitido ao alvo ($u_1 = u_2 = 100$ pixels/s).

Para determinar a maior alteração no valor de R_X, é necessário obter o maior e o menor valor que a rotação pode assumir. Para tanto, como o maior deslocamento do alvo é de 50 pixels, considera-se todos os pixels em posições (x_1, x_2) com valores entre -50 e 50. Através do software Scilab, foram calculados os valores de R_X para todas essas posições. Para uma melhor visualização, foram construídos, também com o Scilab, gráficos correspondentes a esses valores, dados pelas Figuras B.1 e B.2. As rotações máxima e mínima obtidas são:

$$\text{Valor mínimo de } R_X = -0,3029 \text{ rad/s}$$
$$\text{Valor máximo de } R_X = -0,2881 \text{ rad/s}$$

Assim, o erro máximo (em valor absoluto) é igual a $0,3029 - 0,2881 = 0,0148$ rad/s, que corresponde a $0,848$ grau/s. Para calcular o erro de ângulo a cada instante de amostragem, basta multiplicar o valor obtido pelo período de amostragem:

$$\begin{matrix}\text{Erro máximo de posicionamento angular} \\ \text{a cada instante de amostragem}\end{matrix} = 0,848 \times 0,5 = 0,424 \text{ graus}$$

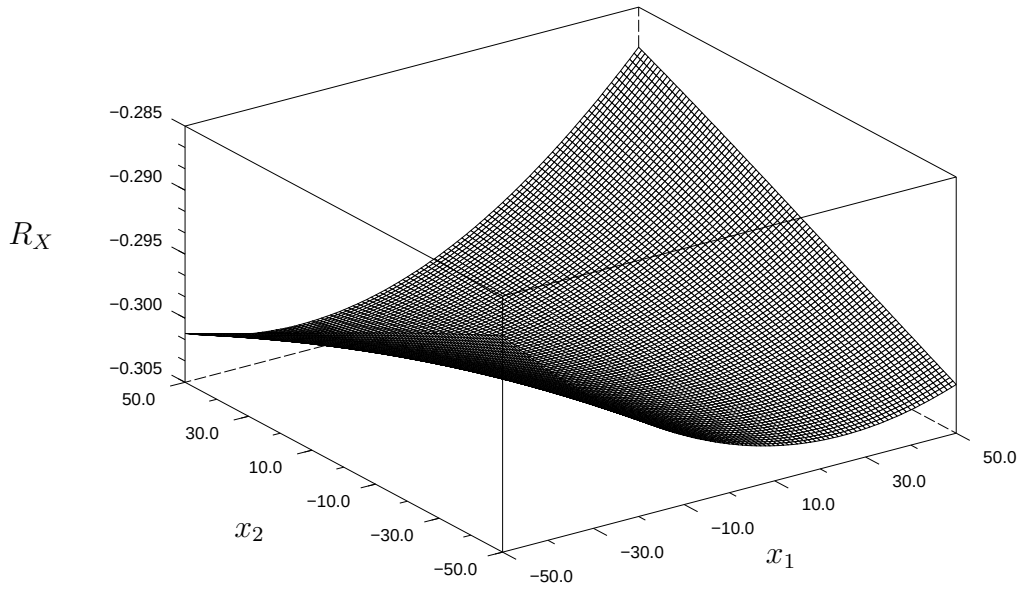

Figura B.1: Gráfico da variação de R_X com x_1 e x_2.

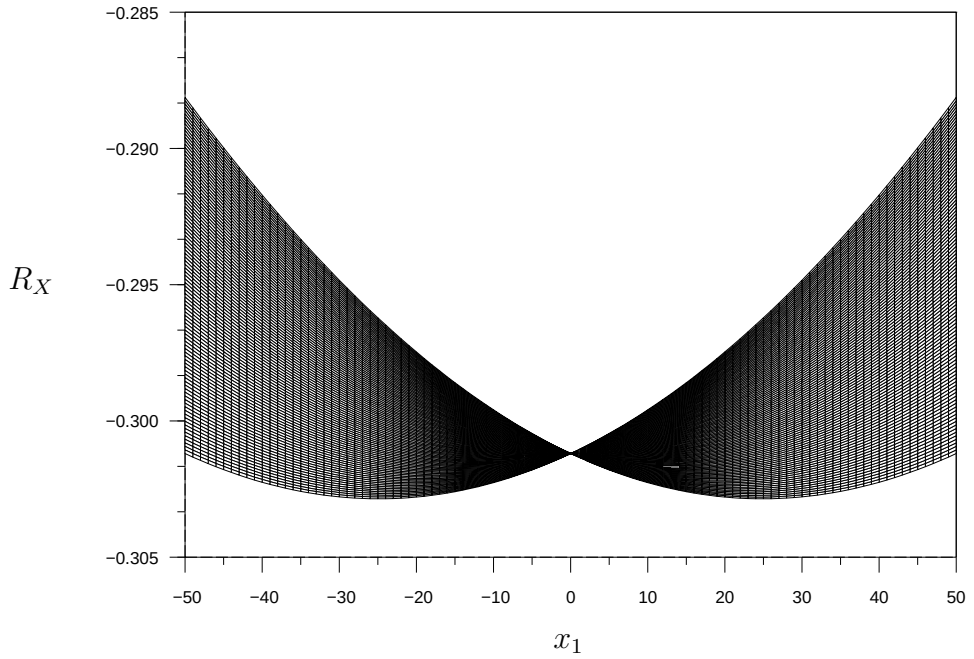

Figura B.2: Gráfico da variação de R_X com x_1 e x_2 (vista lateral).

Portanto, o maior erro de posicionamento dos ângulos pan e tilt devido à utilização de rotações constantes durante um período de amostragem é de $0,424$ grau. Como a resolução

do mecanismo pan-tilt utilizado neste trabalho é de 1 grau, a aplicação da aproximação não representa prejuízo significativo de precisão.

OBS.: A análise realizada utiliza como base o valor de R_X, mas poderia ter sido efetuada a partir de R_Y. Por outro lado, não há garantias de que os valores máximo e mínimo de rotação são obtidos para as variáveis de controle utilizadas ($u_1 = u_2 = 100$ pixels/s). Contudo, para efeitos de estimação, estes valores são aceitos como razoáveis.

Rotação máxima da câmera

A partir do maior valor calculado de rotação que a câmera pode apresentar, é necessário verificar se o equipamento utilizado é capaz de proporcionar adequadamente a velocidade desejada.

No item anterior, encontra-se uma estimativa do maior valor (em módulo) que a rotação R_X pode assumir. Este valor corresponde a $0,3029$ rad/s, ou ainda a $17,35$ graus/s. A velocidade de movimentação dos servo-motores do mecanismo pan-tilt não é precisamente conhecida, mas, garantidamente, é superior a 100 graus/s.

Outra maneira de comparar esses valores é através da quantidade de tempo disponível. Para cada período de amostragem, o deslocamento angular máximo é de $17,35 \times 0,5 = 8,68$ graus. Assim, o tempo necessário para que o mecanismo pan-tilt apresente este deslocamento é de, no máximo, $8,68 \div 100 = 0,087$ s.

Influência do valor da distância focal na rotação da câmera

No Apêndice A são apresentados dois métodos de determinação da distância focal da lente da câmera. A análise a seguir tem como objetivo verificar o valor de rotação da câmera para os dois casos.

Considerando novamente a rotação R_X, dada pela equação (4.1.8), e tomando-se como exemplo a posição (x_1, x_2) = (0, 0), a equação é simplificada para:

$$R_X = -\frac{u_2}{f} \tag{B.0.1}$$

Para o maior valor possível para a rotação, é considerado o valor máximo da variável de controle $u_2 = 100$ pixels/s. Em relação à distância focal, considera-se, no caso do valor obtido experimentalmente, o valor médio somado ao desvio padrão, para efeito de análise. Dessa maneira, as diferenças entre as distâncias focais e entre os valores a serem obtidos de R_X são maximizadas.

- Valor obtido através dos parâmetros da câmera:

$$R_X = -\frac{100}{320} = -0,3125 \text{ rad/s}$$

- Valor obtido empiricamente:

$$R_X = -\frac{100}{332,01 + 11,63} = -0,2910 \text{ rad/s}$$

A diferença (em módulo) das rotações calculadas é dada por $0,3125 - 0,2910 = 0,0215$ rad/s, ou $1,232$ graus/s. O deslocamento correspondente em um período de amostragem é de $1,232 \times 0,5 = 0,616$ graus. Novamente, dado que a resolução de posicionamento do equipamento é de 1 grau, a influência da diferença dos valores obtidos de distância focal é pouco significativa no cálculo das rotações da câmera.