Pontifícia Universidade Católica
do Rio de Janeiro

Ilana Nigri

Comparação entre controles *look-and-move* e servo-visual utilizando transformadas SIFT em manipuladores do tipo *eye-in-hand*

Dissertação de Mestrado

Dissertação apresentada ao Programa de Pós-graduação em Engenharia Elétrica da PUC-Rio como requisito parcial para a obtenção do título de Mestre em Ciências da Engenharia Elétrica

Orientador: Raul Queiroz Feitosa
Co-orientador: Marco Antonio Meggiolaro

Rio de Janeiro
Junho de 2009

Pontifícia Universidade Católica
do Rio de Janeiro

Ilana Nigri

Comparação entre Controles *look-and-move* e servo-visual Utilizando Transformadas SIFT em Manipuladores do Tipo *eye-in-hand*

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Programa de Pós-Graduação em Engenharia Elétrica do Departamento de Engenharia Elétrica do Centro Técnico Científico da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

Dr. Raul Queiroz Feitosa
Orientador
Departamento de Engenharia Elétrica – PUC-Rio

Dr. Marco Antônio Meggiolaro
Co- Orientador
Departamento de Engenharia Mecânica

Dr. Mauro Speranza Neto
Departamento de Engenharia Mecânica

Dr. Fernando César Lizarralde
COPPE/UFRJ

Prof. José Eugenio Leal
Coordenador Setorial do Centro
Técnico Científico

Rio de Janeiro, 03 de junho de 2009

Ilana Nigri

Graduou-se em Engenharia de Controle e Automação na PUC-Rio (Pontifícia Universidade Católica do Rio de Janeiro) em 2007.

Ficha Catalográfica

Nigri, Ilana

Comparação entre controles look-and-move e servo-visual utilizando transformadas SIFT em manipuladores do tipo eye-in-hand / Ilana Nigri ; orientador: Raul Queiroz Feitosa ; co-orientador: Marco Antonio Meggiolaro. – 2009.
109 f. : il. (color.) ; 30 cm

Dissertação (Mestrado em Engenharia Elétrica)– Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2000.
Inclui bibliografia

1. Engenharia elétrica – Teses. 2. Manipulador robótico. 3. Sistemas eye-in-hand. 4. Controle look-and-move. 5. Controle servo-visual. 6. Transformações SIFT. I. Feitosa, Raul Queiroz. II. Meggiolaro, Marco Antonio. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. IV. Título.

CDD: 621.3

Ao meu avô.

Agradecimentos

A Deus.

Ao meu orientador Dr. Raul Queiroz Feitosa pelos ensinamentos transmitidos e pelo auxílio constante.

Ao meu co-orientador Dr. Marco Antonio Meggiolaro pela convivência diária, auxiliando, ensinando e participando não só no mestrado, mas também dos 5 anos de graduação.

Ao CNPq, à PUC-Rio e à Petrobras pelas ferramentas que possibilitaram realizar todos os experimentos na mais alta qualidade.

Aos meus pais que souberam entender minha ausência, sem deixar de me apoiar todos os dias.

Ao meu irmão que sempre esteve presente, alegrando os meus dias mais difíceis.

A minha avó maravilhosa, que sempre tinha uma palavra de incentivo.

Aos meus amigos do Laboratório de Robótica, por estarem sempre dispostos a me ajudar: Júlio, Alexandre, Camila, Gui Franco, Gui Porto, Esguerda, Fabiano, Pet, Tico, Michel, Emo, Mourão, Paulete

Aos meus amigos do LVC pelo companheirismo e incentivo: Dário, Gilson, Paula, Cecília e Denis.

Aos meus queridos amigos da dança, minha segunda família.

Resumo

Nigri, Ilana; Feitosa, Raul Queiroz; Meggiolaro, Marco Antonio. **Comparação entre controles *Look-and-Move* e Servo-Visual utilizando transformadas SIFT em manipuladores do tipo *eye-in-hand***. Rio de Janeiro, 2009, 109p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Visão Computacional pode ser utilizada para calibrar e auto-localizar robôs. Existem diversas aplicações de auto-localização e controle aplicadas a manipuladores industriais e robôs móveis. Em particular, o controle visual pode ser útil em intervenções submarinas, nas quais um manipulador robótico é acoplado a um ROV (Veículo de Operação Remota) para execução de tarefas em grandes profundidades, como o manuseio de válvulas de equipamentos como manifolds. Este trabalho tem como objetivo desenvolver e implementar técnicas de controle visual para auto-localização e posicionamento de manipuladores robóticos. Assume-se que o manipulador possui uma câmera presa em sua extremidade (configuração *eye-in-hand*). Duas técnicas de controle visual são estudadas: *look-and-move* e servo-visual, que diferem entre si pela realimentação do controle. A primeira utiliza sensores de posição, a partir de uma única imagem capturada no início da movimentação. A segunda utiliza diversas imagens capturadas durante o processo. A principal contribuição deste trabalho está no uso da transformada SIFT, robusta a rotações, translações, mudança de escala e iluminação, para obter e correlacionar pontos-chave entre as imagens de referência e capturadas em tempo real. A metodologia é validada experimentalmente através de um manipulador robótico baseado na estrutura mecânica de uma mesa x-y-θ. Um sistema eletrônico é utilizado como interface entre o robô e o software de controle, onde estão implementadas todas as técnicas propostas. Testes iniciais são realizados com imagens de objetos circulares, sem o uso de transformações como o SIFT. Em seguida, são feitos testes com a imagem de um painel real de um manifold, utilizando transformadas SIFT para determinar a localização do manipulador em relação ao painel e controlá-lo até uma pose desejada. Os resultados mostram que o desempenho do controle servo-visual depende muito do tempo de processamento de cada imagem, ao contrário do *look-and-move*. No entanto, o controle servo-visual apresenta erros finais de posicionamento muito menores. O método

SIFT é apropriado para uso em ambos os controles, desde que a resolução das imagens seja alta o suficiente para evitar correlações falsas.

Palavras-chaves

Manipulador robótico; sistemas *eye-in-hand*; controle *look-and-move*; controle servo-visual; Transformações SIFT.

Abstract

Nigri, Ilana; Feitosa, Raul Queiroz; Meggiolaro, Marco Antonio. **Comparison between look-and-move and visual servo control using SIFT transforms in eye-in-hand manipulator systems**. Rio de Janeiro, 2009, 109p. MSc. Dissertation – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Computer vision can be used to calibrate and self-localize robots. There are many applications in self-localization and control applied to industrial manipulators and mobile robots. In particular, visual control can be useful in submarine interventions, where a robotic manipulator is mounted on a Remote Operated Vehicle (ROV) to execute tasks at high depths, such as handling manifold valves. This work has the objective to develop and implement visual control techniques to self-localize and position robotic manipulators. It is assumed that a monocular camera is attached to the robot end-effector (eye-in-hand configuration). Two visual control techniques are studied: look-and-move and visual servo control. Their main difference is related to the adopted feedback sensors. The first technique uses position sensors with the aid of a single image captured at the beginning of the robot movement. The second technique relies on several images captured in real time during the robot movement. The main contribution of this work is the use of the SIFT transform, robust to rotation, translation, changes in scale and illumination, to obtain and correlate key-points between reference images and images captured in real time. The methodology is experimentally validated using a manipulator based on the mechanical structure of an x-y-θ coordinate table. An electronic system was developed to control the robot through a software in a computer, where were implemented all the techniques proposed. Preliminary tests are performed on simple circular-shaped objects, without the need for SIFT transforms. Next, tests are performed with a photo of an actual manifold panel typically used in submarine interventions, using SIFT transform to find the localization of the manipulator with respect to the panel. The results show that the performance of the visual servo control depends on the image processing time, unlike the look-and-move. However, the visual-servo control presents smaller positioning errors. The SIFT method is appropriate for both controls, since image resolution be high enough to avoid false matching.

Keywords

Robotic manipulator; eye-in-hand; look-and-move control; visual servo control; SIFT Transform.

Sumário

Lista de figuras

Lista de tabelas

1
Introdução

Após o crescimento individual das áreas de robótica e visão computacional, percebe-se hoje em dia uma fusão de ambas. A visão computacional é uma ciência que tem como objetivo extrair informações de imagens capturadas por dispositivos como câmera de vídeo, *scanner*, etc. Por sua vez, a robótica está sendo muito explorada no campo industrial, a fim de garantir uma melhora significativa na eficácia e qualidade do trabalho, além da possibilidade de substituir o trabalho humano em locais perigosos e de difícil acesso.

Sistemas robóticos podem ser entendidos como dispositivos eletro-mecânicos equipados com sensores e atuadores, controlados através de um sistema computacional. Estes sensores são utilizados para medir diferentes grandezas no ambiente de trabalho, como por exemplo posição, velocidade e força.

Atualmente, a maior parte dos robôs industriais é programada para seguir uma trajetória pré-definida. Isso é suficiente quando o robô trabalha num ambiente fixo, onde os objetos de interesse estão sempre a uma distância pré-definida do robô. Entretanto, se a posição do robô é alterada, todas as trajetórias devem ser reprogramadas para que este ainda seja capaz de realizar as devidas operações (Augustson, 2007).

A visão estereoscópica ou estéreo diz respeito à reconstrução da informação em três dimensões. É a reprodução artificial da visão binocular natural. A visão monocular permite examinar a posição e a direção dos objetos dentro do campo da visão humana em um único plano. Permite reconhecer nos objetos a forma, as cores e as dimensões. A fotografia simples é uma reprodução da visão monocular. Por outro lado, a visão binocular permite a percepção de profundidade, que é dada pela diferença de ângulos com que as imagens são percebidas.

A estereoscopia é utilizada para a obtenção da posição de um ponto no espaço tridimensional. Para isso, no entanto, é necessário determinar projeções de pontos da cena em um par de imagens. Diversos algoritmos para este fim foram

propostos na literatura, tais como o algoritmo proposto por (Tomasi e Kanade, 1991), e (Harris e Stephens, 1988).

Um dos mais recentes algoritmos utilizados na área de visão computacional para este fim é o algoritmo SIFT (*Scale Invariant Feature Transform*), proposto por David Lowe (2004). Este algoritmo é invariante a rotação, a iluminação, a escala e a posição da câmera, e garante um bom desempenho para diferentes tipos de condições de captura das imagens.

Com o objetivo de explorar as vantagens da visão computacional e da robótica, a PETROBRAS patrocina projetos de identificação de pose de manipuladores durante intervenções submarinas. O manipulador TA-40 (Figura 1), acoplado a um ROV (*Remote Operating Vehicle*), tem como objetivo principal operar válvulas em painéis submarinos sob grandes profundidades. Um sistema baseado em visão computacional seria capaz de determinar automaticamente a posição de um objeto alvo. Já o ROV é o veículo responsável pela locomoção do manipulador, capaz de operar em grandes profundidades.

Figura 1 - Manipulador TA-40

Inspirado na aplicação acima, o presente trabalho tem como objetivo comparar diferentes técnicas de controle servo-visual através do desenvolvimento de um sistema robótico experimental composto de uma mesa automatizada XYθ e

uma câmera acoplada à sua extremidade. Esta câmera é responsável pela captura de imagens que servirão de base para o sistema de visão computacional determinar e controlar a posição da mesa robótica. O projeto é composto de quatro partes distintas: a primeira referente à mecânica do robô em questão, onde uma mesa XYθ comercial deverá ser automatizada e controlada por um computador. A segunda refere-se à eletrônica utilizada como interface entre o robô e o computador. A terceira parte refere-se ao *software* de visão computacional, responsável por determinar, a partir de imagens obtidas pela câmera, a posição do alvo a ser alcançada pelo robô. Será utilizado o algoritmo SIFT para a determinação dos pontos chaves nas imagens. A última parte diz respeito às diferentes técnicas de controle a serem aplicadas no posicionamento do robô. Através do controle, espera-se que o robô seja capaz de alcançar diferentes configurações em relação aos objetos alvos. Espera-se que o robô atinja seu alvo utilizando a realimentação de sensores de posição (*encoders*), e através de imagens capturadas ao longo de seu percurso. Smith e Papanikolopoulos (1996) chamam o controle com realimentação exclusiva por sensores de posição de controle "cego", uma vez que, determinada a posição do alvo, o robô se movimenta sem o auxilio da câmera. Já o controle com realimentação por imagem utiliza vários *frames* da câmera, e por isso é capaz de compensar erros de ruídos e de posicionamento durante o percurso. Espera-se, no entanto, que este apresente um desempenho computacional inferior devido ao tempo de processamento de cada imagem.

Diversos autores já desenvolveram pesquisas nestas áreas. Inoue e Shirai (1971) utilizaram um manipulador robótico de 7 graus de liberdade, com uma câmera na extremidade do último elo (sistema *eye-in-hand*), para encaixar um objeto dentro de um orifício do mesmo formato. Através de programação própria, o controle determina visualmente a posição deseja e a real, e através apenas de imagens o objeto alcança o alvo desejado. Allota e Colombo (1999) implementaram um sistema robótico com câmera também na extremidade. As características visuais foram obtidas através de contornos, e com o desenvolvimento de um controle 2D/3D baseado em imagens. O sistema foi capaz de realizar tarefas de posicionamento e movimentação de objetos. Houshangi (1990) desenvolveu um sistema para capturar objetos em movimento usando uma câmera fixa e pré-calibrada.

A contribuição deste trabalho está na comparação entre as principais arquiteturas em um mesmo sistema experimental, especialmente desenvolvido para este propósito. Além disto, este trabalho apresenta uma formulação eficiente para o controle servo-visual usando SIFT quando os pontos de referência se encontram em um mesmo plano.

Esta dissertação está dividida em seis capítulos, descritos da seguinte forma: o capítulo 2 apresenta a teoria necessária para a compreensão do trabalho, dividida entre os assuntos referentes à área de controle e robótica e os assuntos ligados a área de visão computacional. O capítulo 3 apresenta a integração entre as áreas de visão computacional e controle robótico, descrevendo todo o equacionamento e métodos utilizados. O capítulo 4 descreve o procedimento experimental realizado para validar o projeto e para a extração de resultados. As equações apresentadas no capítulo 3 são adaptadas de forma a torná-las compatíveis com o experimento. O capítulo 5 apresenta os resultados alcançados usando as diferentes técnicas de controle. É feita uma comparação entre as técnicas implementadas, apontando as vantagens e desvantagens de cada uma. O capítulo 6 apresenta as conclusões do trabalho.

2
Visão Computacional

Este capítulo descreve tópicos relacionados à visão computacional subjacentes ao presente trabalho.

A primeira seção apresenta como determinar parâmetros intrínsecos e extrínsecos da câmera e consequentemente a transformada entre a posição da câmera e do objeto, a partir de pares de pontos no espaço e na imagem.

A segunda seção apresenta uma técnica muito conhecida na área de visão computacional chamada SIFT. Este algoritmo é capaz de gerar pares de pontos correspondentes entre duas imagens de diferentes vistas de um mesmo objeto, em diferentes condições de iluminação, rotação ou escala das imagens.

2.1.
Processamento da imagem

Um sistema de visão estereoscópica baseia-se na relação entre as coordenadas de pontos no espaço tridimensional e as coordenadas das projeções destes pontos na imagem. Normalmente estas equações referem-se ao sistema de coordenadas da câmera, sendo as projeções dos pontos na imagem dadas pelas coordenadas linha e coluna em número de *pixels*.

No modelo conhecido como câmera de orifício, apresentado na Figura 2, denomina-se distância focal f a distância entre o plano da imagem e o centro O de coordenadas da câmera. A linha que passa por O e é perpendicular ao plano da imagem é conhecida como eixo ótico. E o ponto onde ocorre à interseção é chamado de ponto principal ou centro da imagem o (Trucco & Verri, 1998).

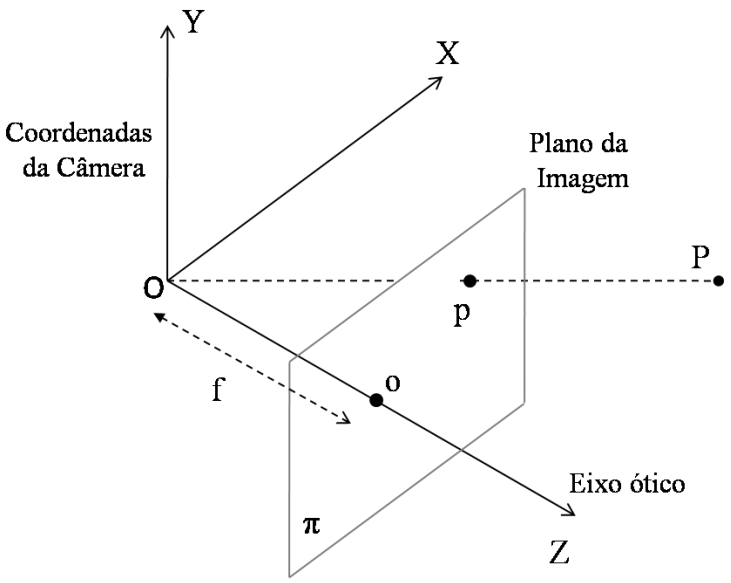


Figura 2 - Modelo de Projeção da Câmera

As equações básicas do sistema de coordenadas da câmera são:

$$x = f\frac{X}{Z} \tag{1}$$

$$y = f\frac{Y}{Z} \tag{2}$$

sendo (x, y) as coordenadas do ponto na imagem, e (X, Y, Z) as coordenadas do ponto no espaço, conhecidas na literatura de visão computacional como coordenadas no mundo.

A relação entre as coordenadas de um ponto no mundo e as coordenadas de sua projeção na imagem, envolve dois grupos de parâmetros chamados parâmetros extrínsecos e intrínsecos.

Os parâmetros extrínsecos definem a localização e orientação de um ponto no sistema de coordenadas da câmera em relação ao mundo. Já os parâmetros intrínsecos são necessários para relacionar o ponto nas coordenadas do *pixel* com o sistema de coordenadas da câmera.

2.1.1. Parâmetros extrínsecos

Os parâmetros extrínsecos definem a transformação geométrica que relaciona unicamente o sistema de coordenadas da câmera e o sistema de coordenadas do mundo. São eles:

- um vetor de translação 3D, T, que descreve a posição relativa entre as origens dos dois sistemas,
- uma matriz ortogonal 3x3 R, que expressa a rotação relativa entre os eixos de um e outro sistema de coordenadas.

Dessa forma, a relação entre o sistema de coordenadas do ponto P no mundo P^m e na câmera P^c é:

$$P^c = R\,P^m + T \tag{3}$$

na qual

$$R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \tag{4}$$

e

$$T = \begin{pmatrix} T_x \\ T_y \\ T_z \end{pmatrix} \tag{5}$$

como é ilustrado na Figura 3.

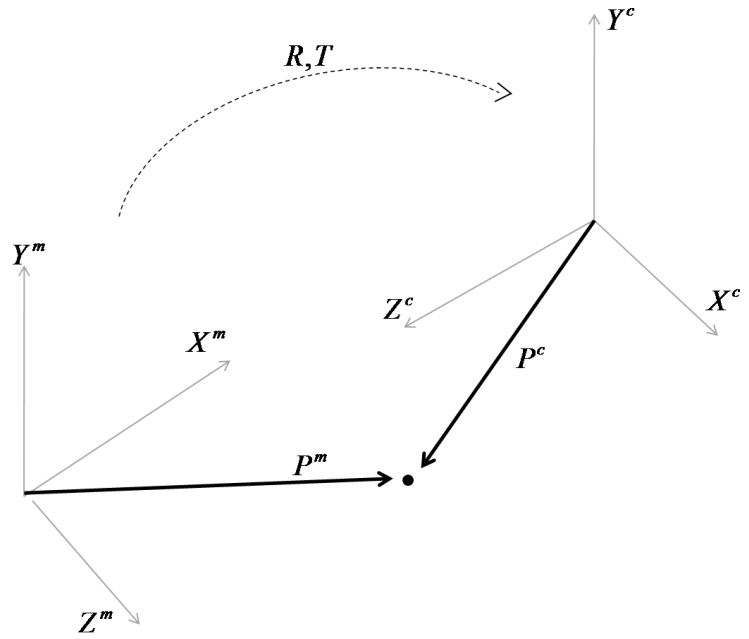


Figura 3 - Transformação do sistema de coordenadas do mundo para sistema de coordenadas da câmera

2.1.2.
Parâmetros intrínsecos

Os parâmetros intrínsecos caracterizam as influências óticas, geométricas e digitais da câmera na imagem. São eles:

- a distância focal;
- as dimensões horizontal e vertical dos pixels na matriz de sensores;
- as coordenadas do vetor que representa deslocamento da origem do sistema de coordenadas da imagem relativamente ao ponto em que o eixo ótico atinge o plano de projeção;
- coeficientes relativos à distorção geométrica causada pela ótica.

Ao determinar a relação de um ponto no sistema de coordenadas da imagem em pixel $\left(x^{im}, y^{im}\right)$ e do mesmo ponto no sistema de coordenadas da câmera $\left(x^{c}, y^{c}\right)$, tem-se que:

$$x^{c} = -\left(x^{im} - o_x\right)s_x \tag{6}$$

$$y^{c} = -\left(y^{im} - o_y\right)s_y \tag{7}$$

onde $\left(o_x, o_y\right)$ são as coordenadas em *pixels* do centro da imagem e $\left(s_x, s_y\right)$ o tamanho efetivo do pixel (em milímetros) nas direções horizontal e vertical.

Na maioria dos casos, as influências óticas podem ser modeladas simplesmente como distorções radiais, da seguinte forma:

$$x^{c} = x_{dis}\left(1 + k_1 r^2 + k_2 r^4\right) \tag{8}$$

$$y^{c} = y_{dis}\left(1 + k_1 r^2 + k_2 r^4\right) \tag{9}$$

sendo $\left(x_{dis}, y_{dis}\right)$ as coordenadas distorcidas dos pontos e $r^2 = x_{dis}^2 + y_{dis}^2$ (Trucco & Verri, 1998). Pode-se observar nas equações (8) e (9) que a distorção no centro da imagem é sempre nula, e cresce conforme o afastamento do centro. Estas distorções podem ser significativas dependendo da câmera utilizada, no entanto são desprezadas no presente trabalho.

2.1.3.
Modelo da câmera

Uma vez determinados os valores dos parâmetros intrínsecos e extrínsecos, pode-se relacionar diretamente o sistema de coordenadas da imagem com o do mundo, sem que seja necessário explicitar o sistema de coordenadas da câmera. Desta forma, substituindo as equações (4) a (7) em (3), tem-se que:

$$x^{im} - o_x = -f_x \frac{r_{11}X^m + r_{12}Y^m + r_{13}Z^m + T_x}{r_{31}X^m + r_{32}Y^m + r_{33}Z^m + T_z} \tag{10}$$

$$y^{im} - o_y = -f_y \frac{r_{21}X^m + r_{22}Y^m + r_{23}Z^m + T_y}{r_{31}X^m + r_{32}Y^m + r_{33}Z^m + T_z} \tag{11}$$

onde $f_x = f / s_x$ e $f_y = f / s_y$.

Observando que as equações (10) e (11) possuem o mesmo denominador, pode-se assumir para cada par de pontos $\left(\left(X_i^m, Y_i^m, Z_i^m\right), \left(x_i^{im}, y_i^{im}\right)\right)$ a equação

$$\begin{aligned} &\left(x_i^{im} - o_x\right) f_y \left(r_{21}X_i^m + r_{22}Y_i^m + r_{23}Z_i^m + T_y\right) = \\ &\left(y_i^{im} - o_y\right) f_x \left(r_{11}X_i^m + r_{12}Y_i^m + r_{13}Z_i^m + T_x\right) \end{aligned} \tag{12}$$

Generalizando, podem-se considerar as coordenadas transladadas $\left(x_i^{im} - o_x, y_i^{im} - o_y\right) = \left(x_i^{im}, y_i^{im}\right)$, considerando que a origem das coordenadas da imagem está no ponto (0,0). Considerando também $\alpha = fx / fy$, a equação (12) pode ser representada como uma equação linear de 8 coeficientes desconhecidos:

$$\begin{aligned} &x_i^{im} X_i^m v_1 + x_i^{im} Y_i^m v_2 + x_i^{im} Z_i^m v_3 + x_i^{im} v_4 - \\ &y_i^{im} X_i^m v_5 + y_i^{im} Y_i^m v_6 + y_i^{im} Z_i^m v_7 + y_i^{im} v_8 = 0 \end{aligned} \tag{13}$$

onde

$$\begin{aligned} v_1 &= r_{21} & v_5 &= \alpha\, r_{11} \\ v_2 &= r_{22} & v_6 &= \alpha\, r_{12} \\ v_3 &= r_{23} & v_7 &= \alpha\, r_{13} \\ v_4 &= T_y & v_8 &= \alpha\, T_x. \end{aligned}$$

2.1.4.
Calibração da Câmera

O processo de determinar os valores dos parâmetros do modelo da câmera é chamado de calibração da câmera. Escrevendo a equação (13) para N pontos cujas coordenadas num referencial do mundo $\left[X_i^m, Y_i^m, Z_i^m\right]$ e suas projeções $\left[x_i^{im}, y_i^{im}\right]$

na imagem são conhecidas, tem-se um sistema homogêneo de N equações lineares como abaixo:

$$Av = 0 \tag{14}$$

onde $v = [v_1,...,v_8]^T$ e a matriz A de dimensão Nx8 é dada por:

$$A = \begin{bmatrix} x_1^{im}X_1^m & x_1^{im}Y_1^m & x_1^{im}Z_1^m & x_1^{im} & -y_1^{im}X_1^m & -y_1^{im}Y_1^m & -y_1^{im}Z_1^m & -y_1^{im} \\ x_2^{im}X_2^m & x_2^{im}Y_2^m & x_2^{im}Z_2^m & x_2^{im} & -y_2^{im}X_2^m & -y_2^{im}Y_2^m & -y_2^{im}Z_2^m & -y_2^{im} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N^{im}X_N^m & x_N^{im}Y_N^m & x_N^{im}Z_N^m & x_N^{im} & -y_N^{im}X_N^m & -y_N^{im}Y_N^m & -y_N^{im}Z_N^m & -y_N^{im} \end{bmatrix} \tag{15}$$

A solução do sistema linear homogêneo da equação (14) é dada pelo auto-vetor correspondente ao menor autovalor de A^TA (Forsyth e Ponce, 2003). Considerando que todos os parâmetros da câmera possuem um fator de escala γ, tem-se que $v = \bar{v}$, e que:

$$\bar{v} = \gamma(r_{21}, r_{22}, r_{23}, T_y, \alpha r_{11}, \alpha r_{12}, \alpha r_{13}, \alpha T_x) \tag{16}$$

Pelas propriedades da matriz de rotação, sabe-se que $r_{21}^2 + r_{22}^2 + r_{23}^2 = 1$, e a partir disso pode-se concluir que:

$$\sqrt{\bar{v}_1^2 + \bar{v}_2^2 + \bar{v}_3^2} = \sqrt{\gamma^2\left(r_{21}^2 + r_{22}^2 + r_{23}^2\right)} = |\gamma| \tag{17}$$

De forma similar, assumindo que $r_{11}^2 + r_{12}^2 + r_{13}^2 = 1$ e que $\alpha > 0$, tem-se que:

$$\sqrt{\bar{v}_5^2 + \bar{v}_6^2 + \bar{v}_7^2} = \sqrt{\gamma^2\alpha^2\left(r_{11}^2 + r_{12}^2 + r_{13}^2\right)} = \alpha|\gamma| \tag{18}$$

Nesta etapa, já foram determinadas as duas primeiras linhas da matriz de rotação, as componentes x e y do vetor de translação, bem como o fator de escala γ e o fator de forma α. Sabendo que a terceira linha da matriz de rotação pode ser determinada pelo produto vetorial entre as duas primeiras linhas, resta apenas determinar a componente z do vetor de translação e a distância focal na direção horizontal f_x.

Para determinar estes últimos parâmetros, retorna-se às equações (10) ou (11) e obtém-se a solução do sistema minimizando o erro quadrático de:

$$A\begin{pmatrix} T_z \\ f_x \end{pmatrix} = b \tag{19}$$

.

Utilizando os mesmos N pares de pontos, tem-se

$$A = \begin{bmatrix} x_1^{im} & \left(r_{11}X_1^m + r_{12}Y_1^m + r_{13}Z_1^m + T_x\right) \\ x_2^{im} & \left(r_{11}X_2^m + r_{12}Y_2^m + r_{13}Z_2^m + T_x\right) \\ \vdots & \vdots \\ x_N^{im} & \left(r_{11}X_N^m + r_{12}Y_N^m + r_{13}Z_N^m + T_x\right) \end{bmatrix} \quad (20)$$

e

$$b = \begin{pmatrix} -x_1^{im}\left(r_{31}X_1^m + r_{32}Y_1^m + r_{33}Z_1^m\right) \\ -x_2^{im}\left(r_{31}X_2^m + r_{32}Y_2^m + r_{33}Z_2^m\right) \\ \vdots \\ -x_N^{im}\left(r_{31}X_N^m + r_{32}Y_N^m + r_{33}Z_N^m\right) \end{pmatrix} \quad (21)$$

Desta forma, determinam-se todos os parâmetros intrínsecos e extrínsecos da câmera: matriz de rotação R, vetor de translação T, distância focal nas direções horizontal e vertical f_x e f_y, e fator de forma α.

2.2. SIFT (*Scale Invariant Feature Transform*)

O método SIFT (Lowe, 2004) tem como principal objetivo a extração de feições invariantes, chamados pontos chave, em uma imagem, podendo ser utilizado para estabelecer correspondências entre diferentes vistas de um objeto ou de uma cena. Estas feições são invariantes quanto à escala e à rotação da imagem, e são robustas contra distorção, mudanças no ponto de vista do ponto, ruídos na imagem, e variações na iluminação da cena.

Um exemplo do resultado do método SIFT aplicado a uma imagem pode ser visto na Figura 4, onde em azul estão indicados os pontos-chaves determinados pelo algoritmo, bem como a "orientação" do ponto de acordo com o algoritmo.

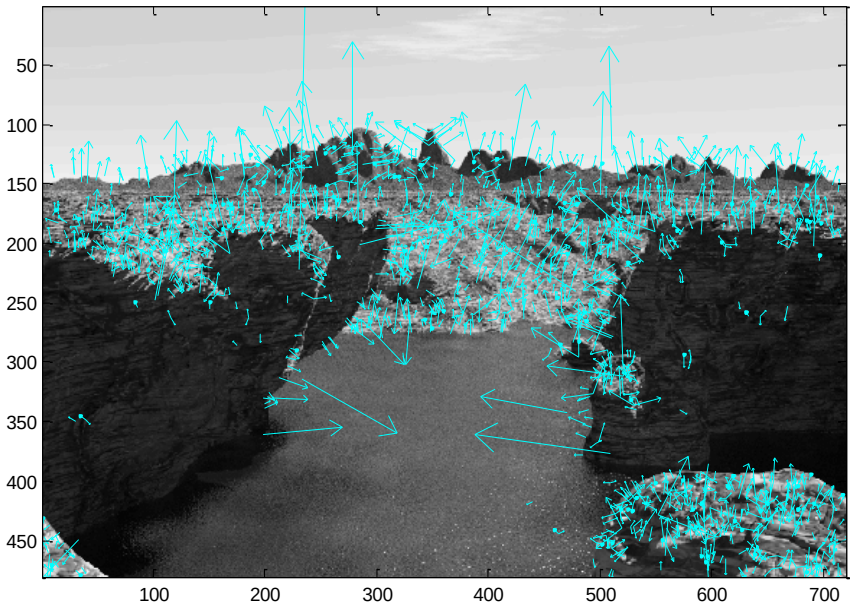


Figura 4 - Resultado do método SIFT aplicado a uma imagem

Apresenta-se a seguir uma descrição sucinta do algoritmo SIFT que consiste de 5 passos sequenciais:

a. Detecção dos pontos chave;
b. Eliminação dos pontos chave "fracos";
c. Determinação da orientação dos pontos chave;
d. Cálculo dos descritores dos pontos chave;
e. Pareamento.

2.2.1. Detecção dos pontos chave

O método tem como entrada uma imagem inicial $I(x, y)$ a partir da qual são criadas diversas imagens em diferentes escalas. A cada uma das imagens produzidas é aplicado um filtro gaussiano $G(x, y, \sigma)$ de suavização com diferentes valores de desvio padrão σ, criando-se assim imagens em vários níveis ou escalas, ou seja

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{22}$$

onde $*$ representa a operação de convolução em x e y, $L(x, y, \sigma)$ representa a imagem suavizada pela gaussiana, e

$$G(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/2\sigma^2} \quad (23)$$

Alcançando um certo nível ocorre uma sub amostragem da imagem que reduz suas dimensões à metade. Cada conjunto de imagens de mesmas dimensões formam uma oitava.

Em seguida imagens de níveis adjacentes são subtraídas para produzir as DOG's (Diferença-de-Gaussianas):

$$D(x,y,\sigma)=(G(x,y,k\sigma)-G(x,y,\sigma))*I(x,y) \quad (24)$$

$$=L(x,y,k\sigma)-L(x,y,\sigma) \quad (25)$$

onde k é o fator de suavização aplicado às gaussianas.

Na assim chamada pirâmide DOG (apresentadas na Figura 5) procuram-se os pontos extremos (máximos ou mínimos) alongo das 3 dimensões (x,y,σ). Cada ponto da pirâmide DOG é comparado com seus 8 vizinhos na mesma escala, além dos 9 pontos vizinhos nas escalas acima e abaixo, conforme mostra a Figura 6. O ponto só é selecionado como ponto chave se seu valor na pirâmide DOG for maior (ou menor) do que o de todos os seus vizinhos.

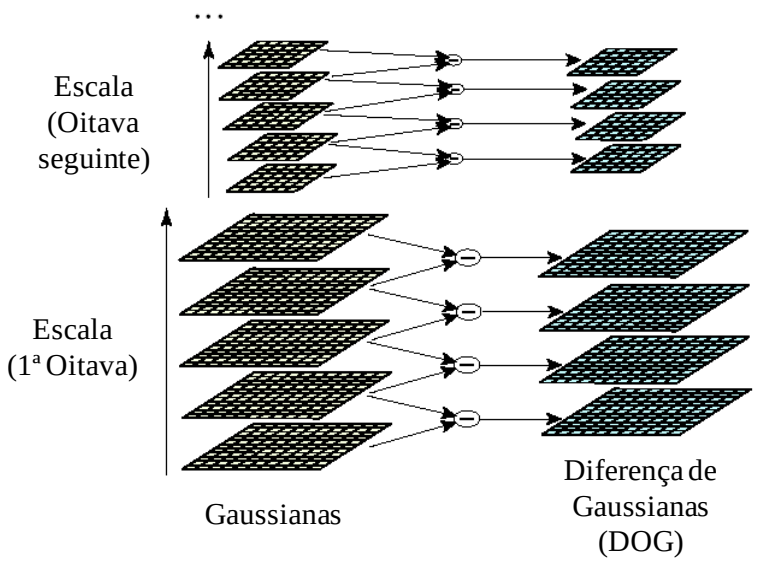


Figura 5 - Gaussianas aplicadas a cada oitava, e a partir de suas subtrações surgem as DOG's (Diferença-de-Gaussianas)

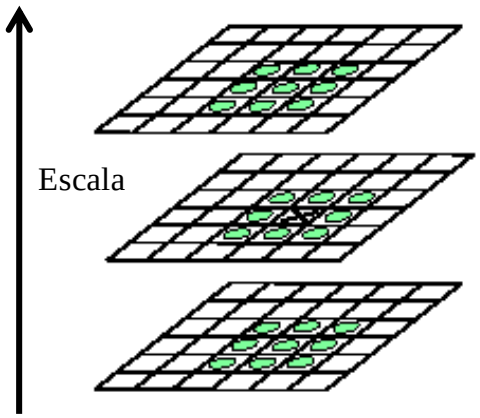


Figura 6 - Detecção dos máximos e mínimos nas diferenças entre gaussianas (DOG's)

2.2.2. Eliminação dos pontos chave "fracos"

Nesta etapa é feita uma filtragem dos pontos chave considerados "fracos", ou seja, que estão sobre arestas e/ou cuja vizinhança tem baixo contraste. A condição do contraste baseia-se no valor na pirâmide DOG sobre o ponto. A condição quanto a localizar-se sobre arestas é verificada utilizando um algoritmo que tem a mesma base teórica dos algoritmos de detecção de cantos já mencionados (Harris e Stephens, 1988).

2.2.3. Determinação da orientação dos pontos chave

Para cada ponto resultante da etapa anterior, calculam-se a magnitude e a orientação do gradiente em cada posição pertencente à vizinhança em torno do ponto, aplicando-se as equações abaixo:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \qquad (26)$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y))) \qquad (27)$$

Monta-se a partir destes valores um histograma de orientações, contendo 36 faixas, cobrindo os 360°. Cada ponto é acrescido ao histograma amortecido por uma gaussiana circular. Picos no histograma de orientação representam a direção dominante do gradiente local. O maior pico de orientação no histograma é encontrado juntamente com quaisquer outros picos 80% do pico mais alto, e são

utilizados para criar *keypoints* com sua orientação. Mais de uma orientação pode ser associada a um *keypoint*. Sendo assim, cada *keypoint* possui 4 dimensões: localização em x, localização em y, escala e orientação.

2.2.4. Cálculo dos descritores dos pontos chave

A Figura 7 ilustra o cálculo do descritor do *keypoint*. Primeiramente a magnitude e orientação do gradiente são amostradas ao redor da localização do *keypoint*. A fim de garantir invariância à orientação, as coordenadas do descritor e a orientação do gradiente são rotacionadas de acordo com a orientação do *keypoint*. Os gradientes são representados no lado esquerdo da Figura 7, por setas indicando assim sua orientação.

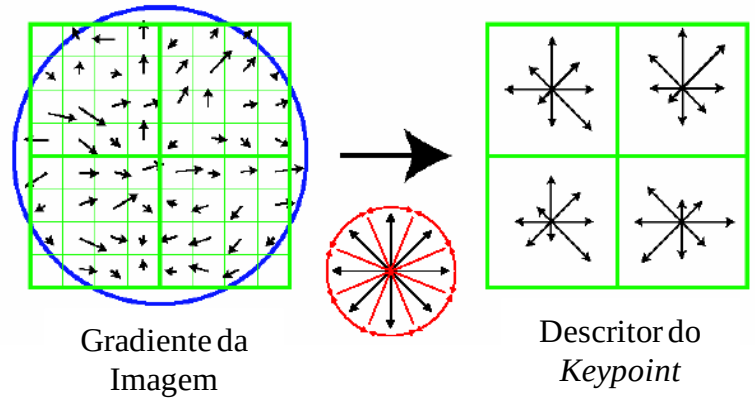


Figura 7 - Orientação dos pontos da vizinhança (esquerda), e descritor do *keypoint* (direita)

O descritor do *keypoint* pode ser observado no lado direito da Figura 7. Cada quadrante do descritor contém a soma dos gradientes onde as setas, em 8 diferentes direções, contém o comprimento equivalente a magnitude do histograma.

A fim de evitar influências nas fronteiras do descritor, aplica-se um peso maior aos pontos centrais. Este peso será igual a $1-d$, sendo d a distância entre o ponto e o centro (horizontal e vertical) da vizinhança.

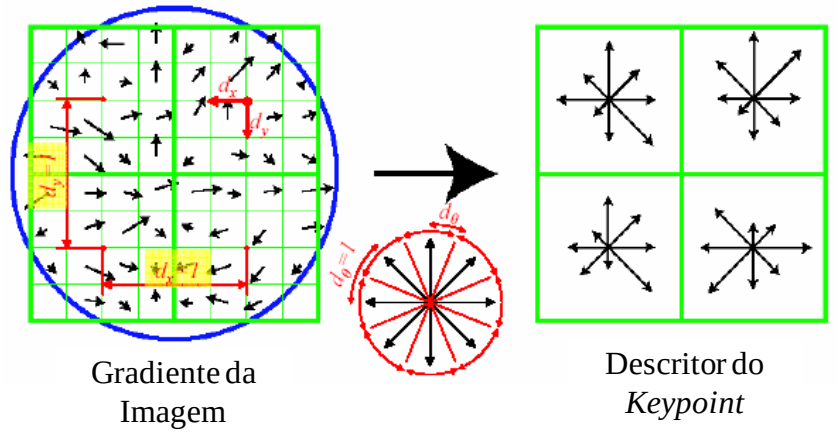


Figura 8 - Descritor dos pontos

Para garantir a invariância à iluminação, o vetor descritor do ponto deve ser normalizado. Suas componentes são limitadas em 0,2, e então o vetor é normalizado novamente.

Assim é criado o descritor de cada *keypoint*. Em seu artigo Lowe apresenta um vetor 2x2 descritor partindo de 8x8 pontos de vizinhança. No entanto, os experimentos realizados tanto em seu artigo, quanto neste trabalho, utilizaram um vetor descritor de tamanho 4x4, admitindo assim uma vizinhança ao redor do ponto de 16x16.

2.2.5. Pareamento

Seguindo todas as etapas mencionadas acima, o método é capaz de determinar os pontos-chaves a partir de uma imagem de entrada, e seus descritores.

Para iniciar o processo de *matching* (determinação dos pares de pontos correspondentes), um par de imagens deve ser submetido ao processo do SIFT, obtendo assim o descritor de cada imagem.

A medida de correlação entre dois pontos é dada pela distância euclidiana entre seus descritores. A fim de determinar o correspondente de um ponto da primeira imagem na segunda, cada ponto do primeiro descritor é comparado com todos os pontos do segundo descritor, e o que apresentar a menor distância será considerado. Para evitar correlações falsas, se a diferença entre os dois primeiros pontos com menor distância for maior que 80%, os pontos são descartados.

Na Figura 9 pode-se observar um exemplo de resultado do método de *matching*.

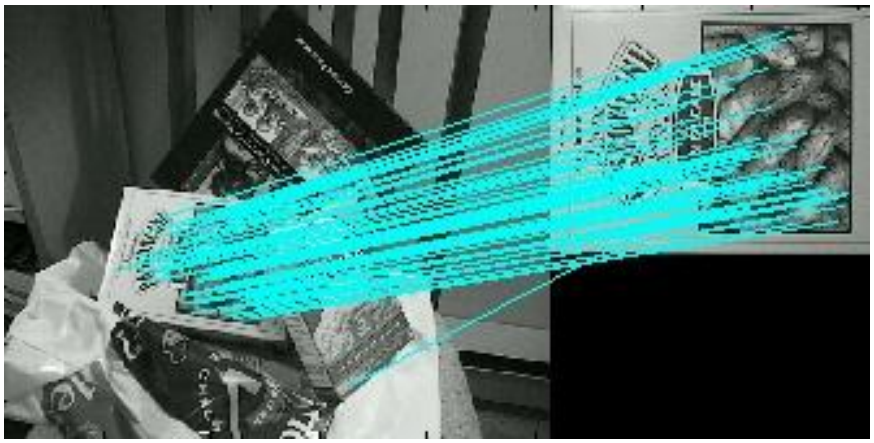


Figura 9 - Resultado do algoritmo de detecção de pontos correspondentes (Lowe, 2004)

3
Controle Visual

Este capítulo irá apresentar de forma geral o problema do controle visual de robôs (manipuladores). Serão apresentadas na seção 3.1 diferentes arquiteturas de controles correlacionando as duas técnicas abordadas neste trabalho, com diferentes variáveis de controle. O sistema apresentado na Figura 10 indica como se relacionam os sistemas mecânicos, eletrônicos e computacionais em um projeto de controle.

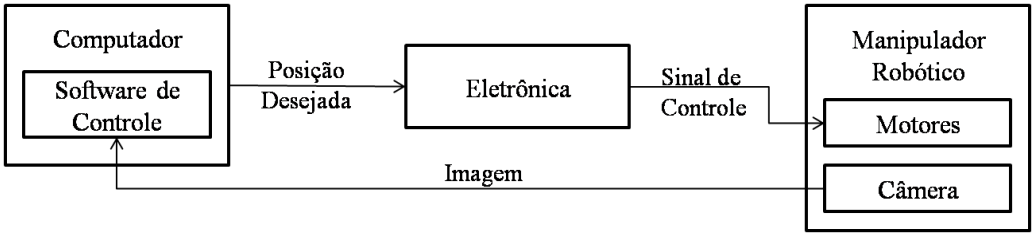


Figura 10 - Esquema de um sistema genérico

O exemplo de controle visual mais comum na literatura é o de um manipulador robótico de N graus de liberdade onde se deseja alcançar um objeto no espaço com o auxílio de uma câmera acoplada à sua extremidade. Conforme apresentado no capítulo anterior, uma vez extraída uma imagem do objeto, o processamento da imagem feito em um *software* de controle, se encarrega de determinar as coordenadas a serem atingidas pelo robô. Conhecidas estas coordenadas, cabe à cinemática inversa indicar quais deslocamentos deverão ser aplicados a cada junta do robô (apresentado na seção 3.2). Estes são enviados a um sistema eletrônico capaz de acionar os motores acoplados às juntas, de forma a atingir a posição desejada.

Para determinar as tensões necessárias a ser enviada aos motores do robô para que este atinja uma pose desejada, será utilizado o controle PID (Proporcional-Integral-Derivativo), descrito na terceira seção.

Por fim é descrita a integração entre o processamento da imagem e o software de controle, com o sistema eletrônico e mecânico do robô.

3.1. Arquiteturas de Controle

Esta seção irá apresentar as diversas técnicas propostas na literatura para controlar sistemas robóticos com processamento de imagens. As principais diferenças entre elas dizem respeito à realimentação e às variáveis desejadas do sistema.

Sanderson e Weiss (1980) introduziram dois conceitos, utilizados até hoje, para classificar sistemas servo-visuais. O sistema *look-and-move* utiliza visão computacional para gerar posições desejadas (*set-points*) para as juntas utilizando apenas uma imagem, capturada no início do movimento, sem realimentação visual. Por outro lado, se o sistema utiliza sucessivas imagens para corrigir em tempo real erros nas juntas, este sistema pode ser referenciado como servo-visual.

Cada uma destas técnicas pode ser implementada seguindo duas diferentes escolhas para variáveis de estado: variáveis baseadas em poses (posições e orientações do robô), ou variáveis baseadas em características da imagem. No controle baseado em poses, é escolhida uma pose relativa desejada entre uma câmera presa à extremidade do robô (sistema *eye-in-hand*) e o objeto de interesse, a ser controlada. Já no controle baseado em características de imagem, o sistema apenas recebe uma imagem associada à posição final desejada, enquanto que o controle se encarrega de mover o robô até que a câmera em sua extremidade visualize uma imagem com mesmas características.

Um dos exemplos mais comuns de controle servo-visual consiste em levar a extremidade de um manipulador até um alvo desejado. Uma câmera acoplada na extremidade do manipulador permite que a posição final (desejada), seja calculada em relação à posição atual (real) do manipulador (Hutchinson, Hager, & Corke, 1996).

Sabendo que o controle pode ser classificado não só pelo uso de uma ou múltiplas imagens, mas também pela variável controlada (poses ou características de imagem), formam-se quatro tipos diferentes de controle, conforme apresentado nos itens a seguir.

3.1.1.
Controle *look-and-move* baseado em pose

Conforme descrito anteriormente, a presença de sensores de posição no controle do tipo *look-and-move* faz com que a realimentação do sistema seja feita apenas nas juntas, cabendo aos sensores de posição a confirmação e realimentação do sistema.

O controle se inicia quando o robô captura uma imagem do ambiente, e a partir desta são extraídas características capazes de determinar a posição real do robô em relação ao alvo (Figura 11). O usuário por sua vez define uma posição desejada em relação ao alvo. A diferença entre a posição desejada e a real inicia o controle, onde sensores de posição acoplados ao robô são capazes de verificar a posição alcançada. A realimentação entre os sensores e o controle garante que a posição final será atingida.

Neste caso, admite-se que o processamento da imagem inicial determinará a posição relativa entre a câmera e o objeto, e que qualquer problema neste processamento irá resultar numa posição final com erros.

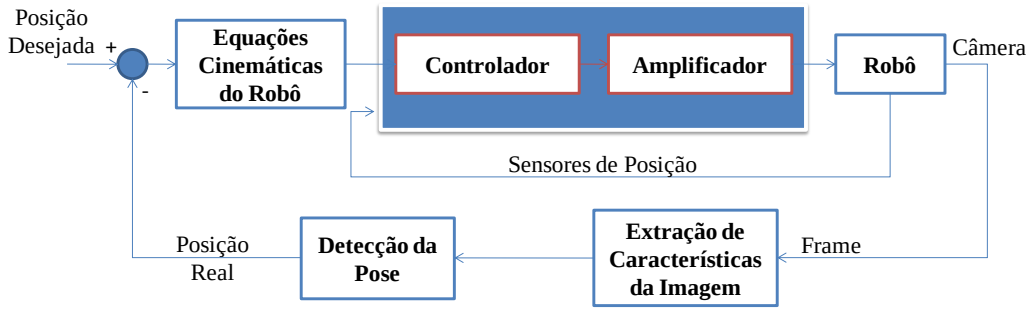


Figura 11 - Controle *look-and-move* baseado em pose

3.1.2.
Controle *look-and-move* baseado em imagem

Esta técnica se difere da mencionada acima em relação aos alvos a serem estabelecidos. No controle baseado em imagens, é necessário determinar as características da imagem a serem alcançadas. Uma vez recebida uma imagem de referência, o robô deverá se posicionar de forma que a imagem extraída naquela posição seja igual à desejada pelo usuário. Para tal, características são extraídas da imagem real e da desejada, e comparadas.

Pela Figura 12, pode-se perceber a presença das matrizes Jacobianas da imagem, que são responsáveis por converter a diferença entre as características desejadas e as reais em parâmetros de entrada do controlador.

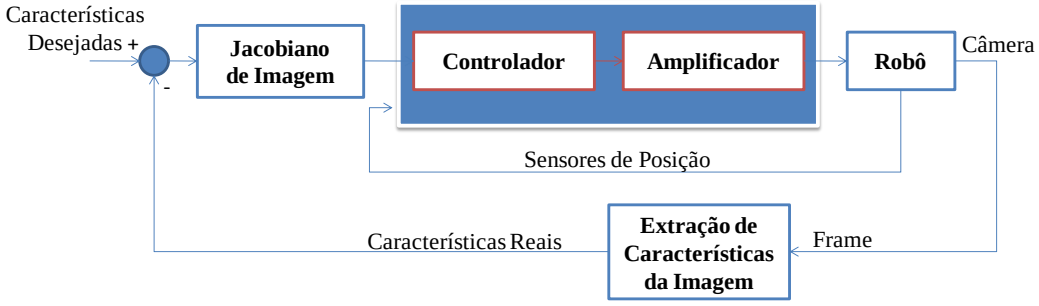


Figura 12 - Controle *look-and-move* baseado em imagem

3.1.3. Controle servo-visual baseado em pose

O controle servo-visual não precisa utilizar informações dos sensores das juntas do robô. A realimentação do sistema é feita através de imagens obtidas em tempo real, onde a cada instante de tempo um novo *frame* é capturado pela câmera, e é determinada uma nova diferença entre a posição desejada e a real. Este ciclo se repete até o momento em que o alvo é atingido (Figura 13).

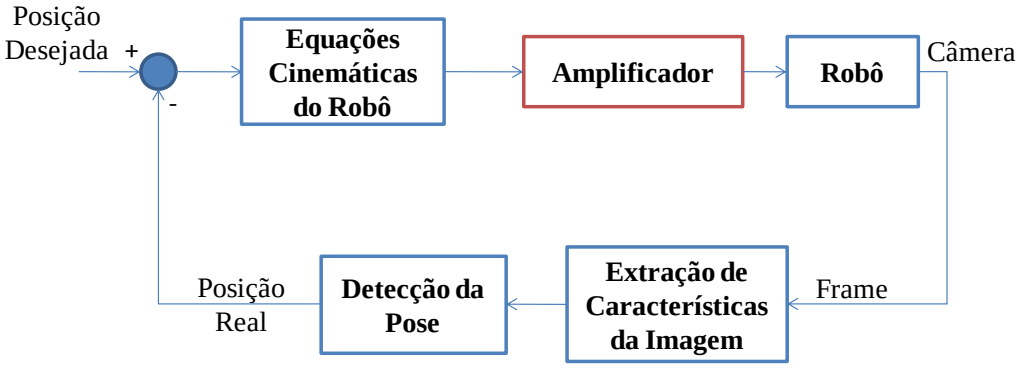


Figura 13 - Controle servo-visual baseado em pose

3.1.4. Controle servo-visual baseado em imagem

Por último, o controle servo-visual baseado em imagem se difere do baseado em pose em relação à variável de estado utilizada. Neste caso o controle se baseia em características da imagem e não mais em ângulos estimados das juntas. O objetivo do robô é atingir uma posição de onde a imagem extraída seja

igual à imagem de referência. Novamente, a presença das matrizes Jacobianas de imagem são as responsáveis pela conversão entre as características da imagem e os parâmetros de entrada do robô (Figura 14).

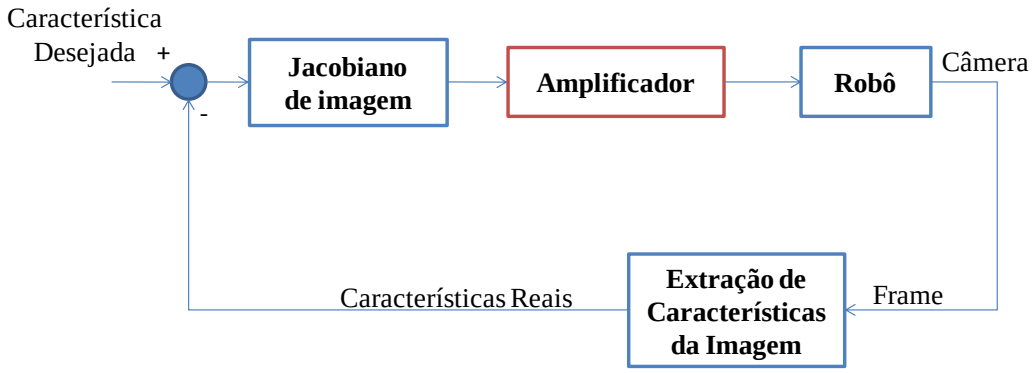


Figura 14 - Controle servo-visual baseado em imagem

3.2. Cinemática Inversa

Uma vez conhecida a transformação entre a posição da câmera e o objeto alvo, se faz necessário determinar as movimentações das juntas do manipulador, capaz de levar a sua extremidade a uma posição (x, y, z) no espaço. Estes valores podem ser obtidos utilizando a cinemática inversa do manipulador. Muitos manipuladores não possuem uma expressão algébrica para as equações da cinemática inversa. Desse modo, é comum utilizar métodos numéricos para a obtenção da cinemática inversa, com o auxílio de matrizes Jacobianas. Para uma melhor compreensão, serão apresentados a seguir o cálculo da matriz Jacobiana em dois casos: o primeiro para um caso particular de um manipulador de 2 graus de liberdade, e em seguida um caso genérico de um manipulador de N graus de liberdade.

3.2.1. Jacobiano de um manipulador plano de 2 graus de liberdade

Para o caso de um manipulador de 2 graus de liberdade com juntas rotativas, as movimentações são restritas a um plano $x_0 - y_0$. Segundo Asada e Slotine (1986), as equações que relacionam a posição de sua extremidade (x, y) e os deslocamentos das juntas (θ_1, θ_2), são

$$x(\theta_1,\theta_2) = l_1\cos(\theta_1) + l_2\cos(\theta_1+\theta_2)$$
$$y(\theta_1,\theta_2) = l_1\sin(\theta_1) + l_2\sin(\theta_1+\theta_2) \quad (28)$$

onde l_1, l_2, θ_1 e θ_2 são definidos na Figura 15.

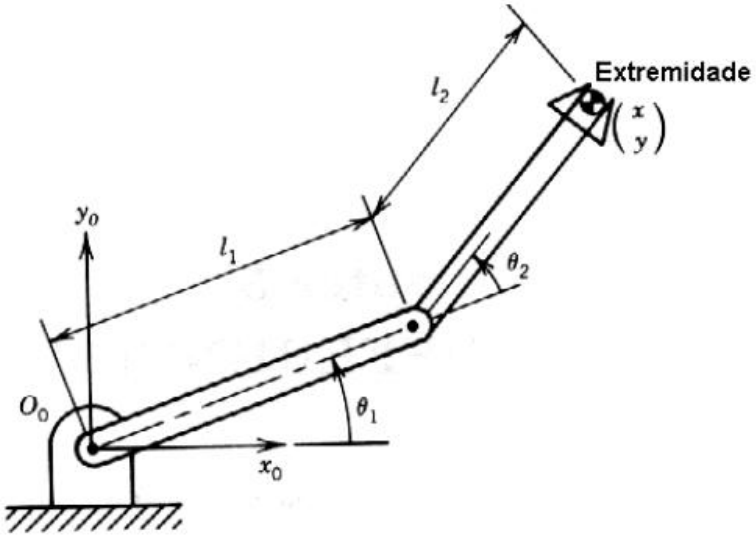


Figura 15 - Manipulador planar de 2 graus de liberdade

No caso do manipulador apresentado na Figura 15, os movimentos infinitesimais das juntas podem ser obtidos simplesmente derivando as equações (28). Sendo assim:

$$dx = \frac{\partial x(\theta_1,\theta_2)}{\partial\theta_1}d\theta_1 + \frac{\partial x(\theta_1,\theta_2)}{\partial\theta_2}d\theta_2$$
$$dy = \frac{\partial y(\theta_1,\theta_2)}{\partial\theta_1}d\theta_1 + \frac{\partial y(\theta_1,\theta_2)}{\partial\theta_2}d\theta_2 \quad (29)$$

Em forma vetorial a equação acima pode ser escrita da seguinte forma:

$$dx = J\ d\theta \quad (30)$$

onde dx e $d\theta$ representam os vetores das movimentações infinitesimais, definidos como:

$$dx = [dx \quad dy]^T \qquad d\theta = [d\theta_1 \quad d\theta_2]^T . \quad (31)$$

A partir da equação (30), é possível calcular numericamente valores de (θ_1,θ_2) associados a uma posição desejada (x,y) utilizando o algoritmo de Newton-Raphson.

A matriz Jacobiana J contém as derivadas parciais das funções $x(\theta_1,\theta_2)$ e $y(\theta_1,\theta_2)$ em relação às juntas de movimento θ_1 e θ_2, e pode ser descrita como:

$$J = \begin{bmatrix} \partial x/\partial \theta_1 & \partial x/\partial \theta_2 \\ \partial y/\partial \theta_1 & \partial y/\partial \theta_2 \end{bmatrix}. \quad (32)$$

Das equações (28) e (32), a matriz Jacobiana do manipulador planar de dois graus de liberdade é

$$J = \begin{bmatrix} -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix} \quad (33)$$

Considerando que as juntas do manipulador se movem com velocidades $\dot{\theta} = [\dot{\theta}_1, \dot{\theta}_2]^T$, e que a velocidade resultante na extremidade do manipulador seja igual a $v = [\dot{x}, \dot{y}]^T$, o Jacobiano representa a relação entre as velocidades nas juntas e na extremidade. Dividindo ambos os lados da equação (30) por dt, obtêm-se finalmente:

$$v = J \cdot \dot{\theta} \quad (34)$$

3.2.2. Jacobiana de um manipulador genérico

Nesta seção será apresentado o cálculo da matriz Jacobiana para o caso de um manipulador genérico de N graus de liberdade. Neste caso, a matriz Jacobiana estará associada à rotação e translação infinitesimal da extremidade. Na Figura 16 representam-se as translações infinitesimais da extremidade como sendo o vetor tri-dimensional dx_e, assim como as rotações infinitesimais são representadas pelo vetor tri-dimensional $d\phi_e$. Ambos os vetores são representados em relação ao sistema de coordenadas $O_0(x_0, y_0, z_0)$. Para simplificar as equações, os dois vetores serão combinados e definidos como o vetor dp de 6 dimensões:

$$dp = \begin{bmatrix} dx_e \\ d\phi_e \end{bmatrix}_{6x1} \quad (35)$$

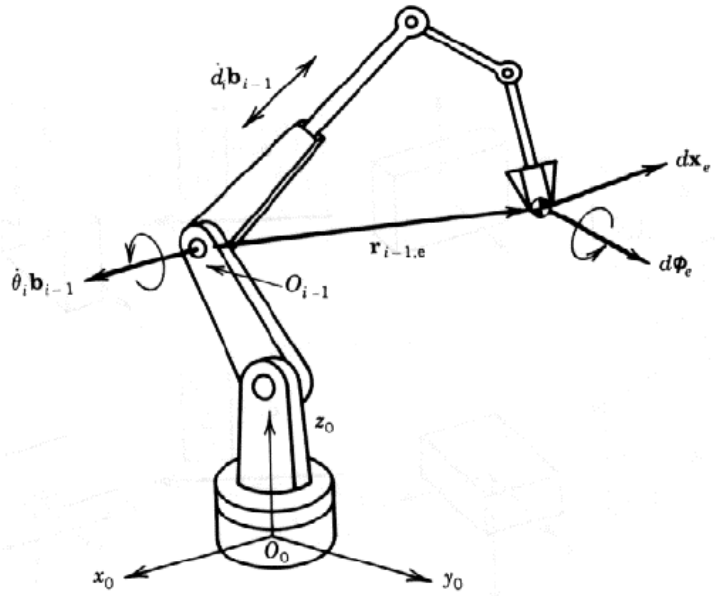


Figura 16 - Movimentações infinitesimais de um manipulador genérico

Dividindo ambos os lados por dt, obtêm-se as velocidades linear e angular da extremidade do manipulador:

$$\dot{p} = \begin{bmatrix} v_e \\ \omega_e \end{bmatrix}_{6x1}. \tag{36}$$

Assim como no primeiro caso (item 3.2.1), a velocidade na extremidade pode ser obtida em função das velocidades nas juntas:

$$\dot{p} = J \cdot \dot{q} \tag{37}$$

onde $\dot{q} = [\dot{q}_1, \dot{q}_2, ..., \dot{q}_n]^T$ é o vetor nx1 de velocidade nas juntas, e J é a matriz Jacobiana geométrica do manipulador. A dimensão da matriz Jacobiana agora é 6xn; as primeiras três linhas são compostas por vetores associados à velocidade linear v_e, já as três últimas linhas são os vetores associados à velocidade angular ω_e.

3.3. Controle PID

Uma vez que a posição/característica desejada é comparada com a real, um controlador deve ser empregado de forma que a informação desejada chegue aos motores responsáveis pelo funcionamento do robô. Diversas técnicas de controle podem ser encontradas na literatura, porém neste trabalho optou-se por

implementar um controle PID (Proporcional-Integral-Derivativo), uma das técnicas mais utilizadas na área de robótica (Astrom & Hagglund, 1995).

O controle PID é conhecido por sua facilidade de implementação, uma vez que este controla individualmente cada junta de um robô. A principal desvantagem desta técnica é o fato de não levar em consideração os efeitos não-lineares da dinâmica do robô, que no entanto não são significativos na maioria dos sistemas *eye-in-hand*, cujas velocidades são limitadas pelo tempo de processamento das imagens. Dependendo da configuração e velocidade do robô, é comum se observar a influência dinâmica de um elo em outro.

Conforme o nome, o controle PID pode ser entendido como a composição de três diferentes técnicas: proporcional, integral e derivativa. A seguir é apresentado seu equacionamento:

$$u_i = K_{Pi}\, e + K_{Ii} \int_0^t e\, dT + K_{Di}\, \dot{e} \tag{38}$$

onde:

- i representa o elo em questão do robô,
- u é o torque de saída do controle,
- e é o erro entre a posição desejada e a real, ou entre as características de imagem
- K_P é o ganho proporcional,
- K_I é o ganho integral,
- K_D é o ganho derivativo.

O primeiro termo, proporcional, controla a rigidez das juntas. Atua basicamente como um amplificador, proporcional ao erro entre a posição desejada e a real.

O controle integral pode ser entendido como um acumulador. Durante todo o processo, o termo cresce com a curva erro *versus* tempo. Este só começa a influenciar significativamente o controle quando o termo proporcional não conseguir mais compensar os pequenos erros (também chamados de erros residuais).

Por último, o controle derivativo representa a taxa (ou velocidade) de variação do erro em relação ao tempo. Este controle pode ser entendido como um amortecedor do sistema, e por isso evita oscilações em torno da posição desejada.

Cada termo mencionado acima possui uma constante de controle (K_P, K_I, K_D). Ziegler e Nichols (1942) propuseram regras para a calibração destes parâmetros, utilizadas especialmente em casos onde não é conhecido o modelo matemático do sistema. Para os demais casos, a sintonia do controle pode ser obtida fazendo uso de técnicas experimentais (Ogata, 1997).

No presente trabalho, as constantes K_P, K_I e K_D foram determinadas através de procedimentos experimentais. A calibração é feita individualmente para cada elo, determinando por vez cada constante. Inicialmente todas as constantes são forçadas em zero, variando apenas o valor de K_P. Quando finalmente é escolhido o valor da constante baseado na rigidez desejada para a junta, varia-se o valor de K_I (mantendo a constante proporcional com o valor já escolhido). Por fim, quando determinado o valor desejado para a constante integral, varia-se o valor da constante derivativa até que este também atinja a calibração desejada. Este procedimento para sintonizar controles PID é bastante comum em procedimentos experimentais onde a movimentação de um elo não influencia em outro. Além disso é utilizado em casos onde não é necessária alta precisão nos valores das constantes.

3.4. Integração da Visão Computacional no Controle

Uma vez estabelecida a posição desejada para cada junta do manipulador, o controle PID (descrito na seção 3.3) é encarregado de levar as juntas a uma posição desejada. Uma vez que o controle PID fornece um valor de atuação, este deve ser processado por um sistema eletrônico capaz de transformar a informação vinda do computador em tensão elétrica para os atuadores. Para isso, essa eletrônica deve possuir um sistema de baixa potência, responsável pela interpretação da informação recebida, e um sistema de alta potência, que será responsável por fornecer a energia requerida pelos atuadores.

Uma vez enviada uma tensão aos atuadores, os sistemas de controle podem ser diferenciados pela forma de realimentação do sistema, ou seja, a maneira como fazem a verificação da posição final. Conforme visto na seção 3.1, as realimentações do sistema podem ser feitas através de sensores de posição acoplados aos motores (controle *look-and-move*), como também utilizando novas

imagens adquiridas ao longo da movimentação do manipulador (controle servo-visual).

Optando-se pelo controle *look-and-move*, uma imagem é extraída da câmera e processada. As posições relativas entre o objeto e a imagem são enviadas ao controle PID, que recebe também as informações vindas dos sensores de posição. O erro entre a posição desejada e a atual é realimentado a cada instante de tempo até que a posição desejada seja atingida.

Pelo controle servo-visual, várias imagens são sucessivamente extraídas da câmera. A cada instante de tempo uma imagem é processada e são determinadas as posições relativas entre o objeto e a imagem. O controle PID é processado utilizando como erro a distância relativa até o alvo. Uma tensão de saída é gerada e encaminhada aos motores. Feito isso, uma nova imagem é capturada pela câmera, e o ciclo se repete. São extraídas diversas imagens da câmera, até que o manipulador atinja a posição desejada.

4
Sistema Experimental

O desenvolvimento do sistema experimental requer a integração entre as áreas de mecânica, eletrônica, controle e visão computacional. Através da Figura 17 é possível compreender melhor todas as etapas do projeto do sistema e como ele está estruturado.

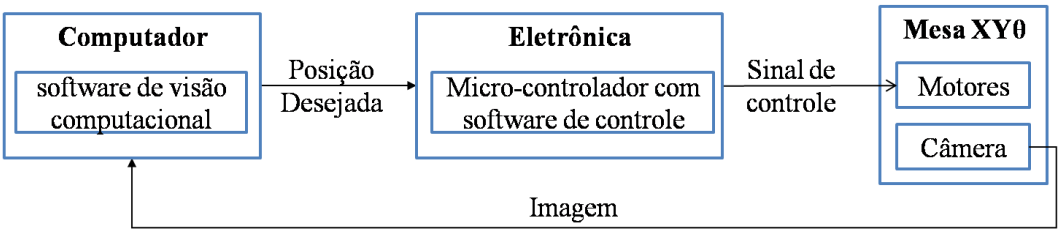


Figura 17 - Esquema do Sistema Experimental

O sistema utiliza uma câmera fixa a uma mesa coordenada XYθ, que irá ser controlada em direção a um objeto alvo. A imagem extraída da câmera é processada por um computador PC, e através de um software de visão computacional são determinadas as posições a serem atingidas pela mesa. Essa posição é enviada a uma placa eletrônica, onde um microcontrolador transforma esta informação em sinais elétricos para os motores/atuadores da mesa.

Este capítulo irá apresentar o procedimento experimental criado para testar e validar os resultados. A primeira seção descreve todo o desenvolvimento mecânico realizado, incluindo a automatização de uma mesa coordenada responsável pela locomoção do protótipo. Em seguida, é apresentado o modelo matemático da mesa, vista como um manipulador robótico. A terceira seção descreve o sistema eletrônico utilizado como interface entre o sistema experimental e o computador. Na quarta seção será apresentado o *software* criado para gerenciar todas as arquiteturas de controle. E por fim os equacionamentos obtidos através do processamento de imagem utilizados dentro do sistema de controle.

4.1.
Sistema mecânico

Conforme mencionado no Capítulo 1, o presente trabalho possui potenciais aplicações em tarefas de intervenção submarina. Em uma típica tarefa de interesse da Petrobras, o manipulador Slingsby TA-40 (Figura 18), acoplado a uma ROV, realiza as movimentações necessárias para que a ferramenta em sua extremidade atinja o objeto desejado, no caso válvulas de equipamentos (*manifolds*). Este manipulador hidráulico, de 6 graus de liberdade, é capaz de suportar uma carga de até 210kg.

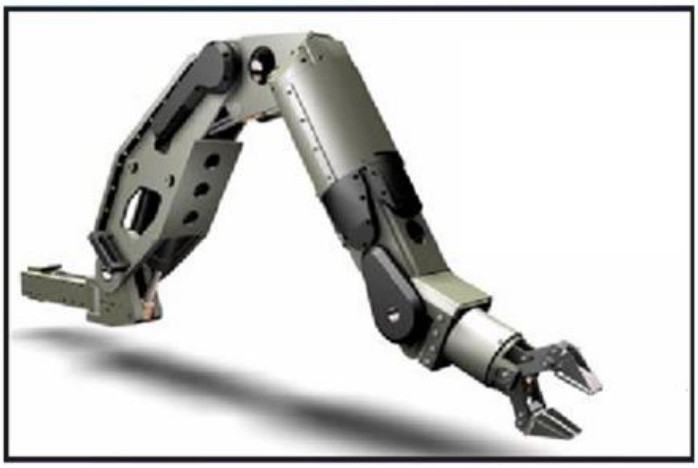


Figura 18 - Manipulador TA-40

A arquitetura fechada do controlador do manipulador TA-40 dificulta a implementação das técnicas de controle propostas sem o suporte do fabricante. Deste modo, optou-se por utilizar uma mesa XYθ, com 3 graus de liberdade, já existente no Laboratório de Robótica da PUC-Rio, para a comprovação experimental. A mesa, apesar de operar no seco, inclui todos os elementos de controle que poderiam ser implementados no manipulador TA-40.

Pela Figura 19, pode-se perceber os 3 motores responsáveis pela movimentação dos 3 elos da mesa. Os dois primeiros elos de translação, são responsáveis pela movimentação em x e y, e o terceiro é referente à rotação. Uma câmera fica acoplada a este último elo, podendo assim ser girada de -180° a 180°.

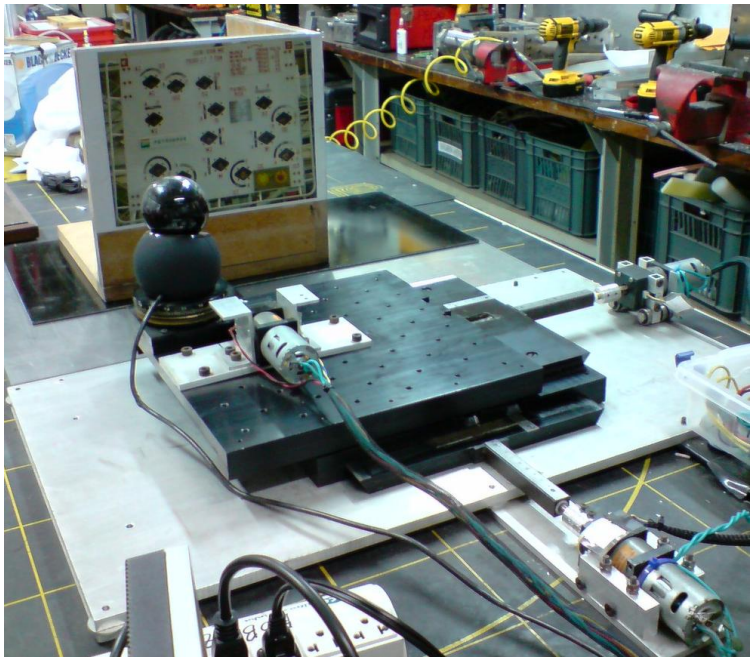


Figura 19 - Mesa Coordenada XYθ

A mesa era originalmente controlada manualmente através de três fusos, porém estes não permitiam uma movimentação da mesa de forma automática. Para a automatização, utilizaram-se 3 motoredutores de corrente contínua da marca *Banebots*, todos com tensão máxima de entrada igual a 24 volts, para que, acoplados aos eixos, pudessem ser controlados por uma eletrônica e posteriormente por um computador. As especificações técnicas e o *datasheet* dos motores encontram-se no Apêndice I.

Figura 20 – Motoredutor *Banebots*

Para as juntas referentes às coordenadas x e y, utilizaram-se motores com redução 16:1. Já para o terceiro elo, referente à angulação da câmera, optou-se pelo motor com redução de 125:1, uma vez que não se fazia necessária alta velocidade de rotação. Foram projetados acoplamentos mecânicos em forma

cilíndrica, conectando os eixos do motor e do fuso da mesa, os quais possuem diferentes diâmetros.

Para manter os motores fixos à mesa, foram realizadas algumas modificações na estrutura do sistema, de forma que não comprometesse a movimentação dos elos. É necessário destacar que o motor referente à movimentação na coordenada x não pode ser fixado à mesa. Isto acontece pois quando o motor referente à movimentação em y é acionado, o posicionamento do motor x muda (conforme a Figura 22). Logo, para que o motor x possa deslizar sobre a mesa, foram acoplados roletes a ele. Na Figura 21, pode-se perceber o acoplamento mecânico construído para unir os elos do motor e da mesa, assim como os roletes presos na base do motor, permitindo assim o seu deslocamento.

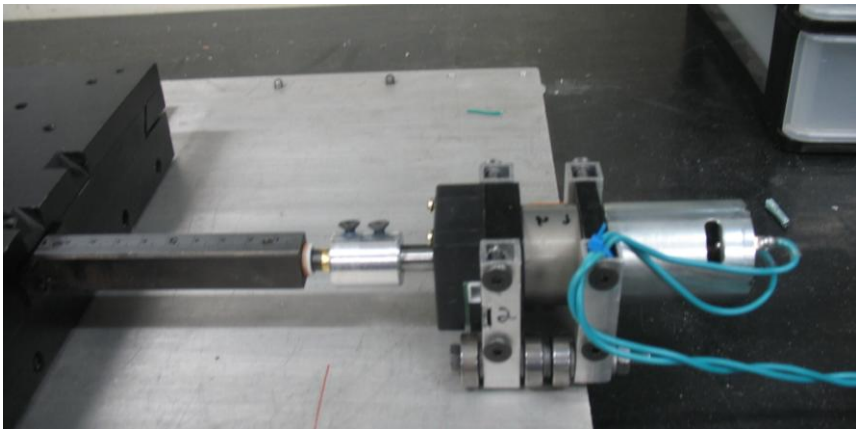


Figura 21 - Acoplamento mecânico entre os eixos do motor e da mesa

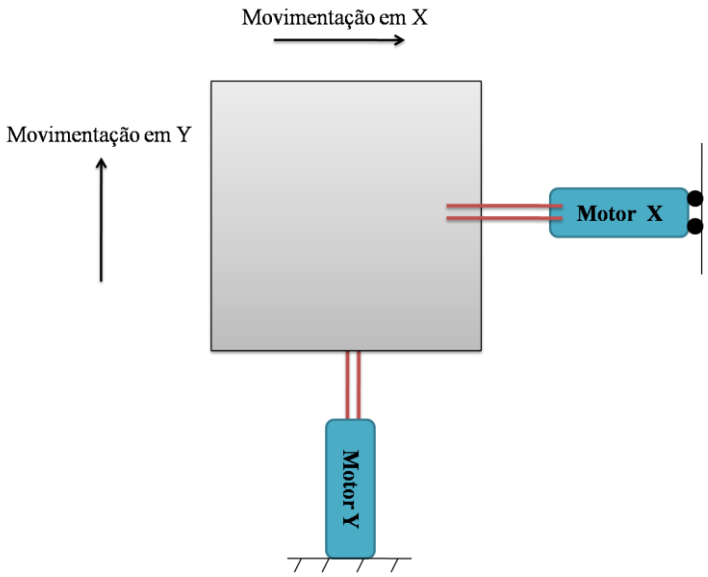


Figura 22 - Esquema da Mesa XYθ

Na extremidade da mesa, foi posicionada uma câmera digital para fazer a aquisição da imagem. Foi escolhida a câmera QuickCam® Orbit AF da marca Logitech, e esta foi acoplada a mesa juntamente com seu suporte próprio (Figura 23). Esta câmera foi escolhida para este projeto pela alta qualidade das imagens extraídas, podendo capturar vídeos de até 960x720 *pixels*, em um total de 30 frames/s. Além da alta qualidade, a câmera possui foco automático, altamente necessário devido à variação de sua posição em relação ao objeto. Por ser uma câmera do tipo *webcam*, sua conexão ao computador é feita através de porta USB, facilitando assim a comunicação.

A câmera possibilita extrair imagens em diversos padrões, porém neste trabalho optou-se por utilizar imagens coloridas, no formato RGB. O tamanho das imagens extraídas variou entre 352x288 e 960x720 *pixels* dependendo do controle utilizado.

Figura 23 - Câmera Logitech utilizada para aquisição das imagens

4.2. Modelo Matemático da Mesa XYθ

Para estabelecer o modelo matemático da mesa XYθ, ela é representada como um manipulador plano de 3 graus de liberdade. As movimentações em x e y podem ser comparadas a juntas prismáticas, sendo suas coordenadas representadas por distâncias d_1 e d_2 respectivamente (vide Figura 24). Já a rotação da câmera na extremidade pode ser interpretada como uma junta rotativa com ângulo representado por θ_3 em relação ao eixo x. São definidos d_1, d_2 e θ_3 como positivos no sentido em que crescem de valor em relação a x, y e no

sentido trigonométrico. A Figura 24 apresenta o Esquema do manipulador juntamente com os parâmetros de cada elo.

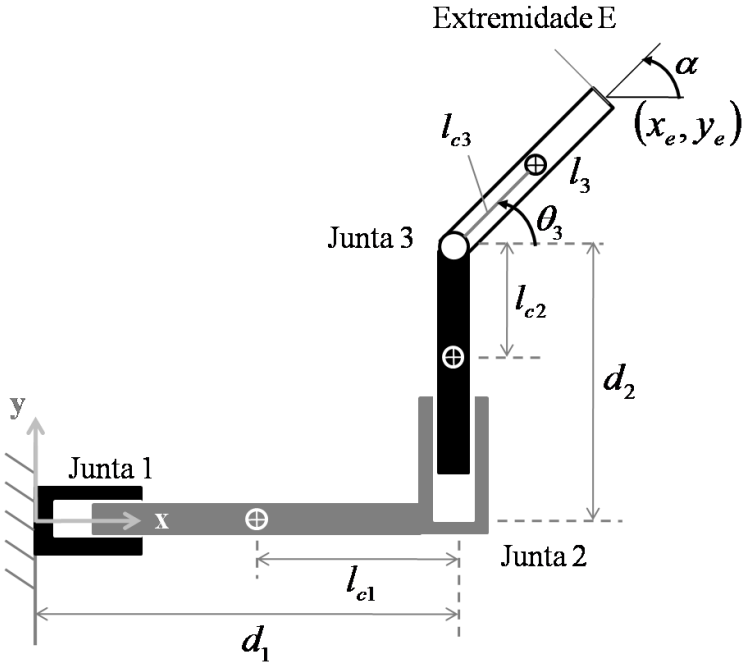


Figura 24 - Parâmetros do Manipulador

Assume-se que os elos 1, 2 e 3 possuem respectivamente massa m_1, m_2 e m_3, momento de inércia em relação ao centro de massa I_1, I_2 e I_3, e a distância entre seu centro de massa e as juntas 2, 3 e 3 são l_{c1}, l_{c2} e l_{c3}. Na mesa coordenada, como o centro de massa da câmera está sobre o centro de rotação da junta 3, $l_{c3} = 0$.

A extremidade do manipulador é definida pelas coordenadas (x_e, y_e, α) e, a partir da Figura 24, pode-se dizer que

$$x_e = d_1 + l_3 \cos(\theta_3) \tag{39}$$

$$y_e = d_2 + l_3 \sin(\theta_3) \tag{40}$$

$$\alpha = \theta_3 \tag{41}$$

A equação de movimento de cada elo é dada pelas seguintes equações:

$$\tau_i = \sum_j H_{ij}\ddot{q}_j + \sum_j \sum_k h_{ijk}\dot{q}_j\dot{q}_k + G_i, \qquad i = 1, 2 \tag{42}$$

sendo H a matriz de inércia do manipulador, h termos não-lineares responsáveis por efeitos como a força centrífuga e termo de Coriolis, G refere-se ao torque que

deve ser adicionado ao elo para compensar os efeitos gravitacionais, e q representa o vetor de posição dos elos $q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \theta_3 \end{bmatrix}$.

Para descobrir as equações do movimento dos elos, é necessário primeiro, calcular a matriz de inércia do manipulador. Como o sistema é plano e só possui uma junta rotatória, os jacobianos analítico e geométrico são iguais. Inicia-se então o cálculo das matrizes Jacobianas que relacionam a velocidade da extremidade com a velocidade das juntas do robô. Estas podem ser dividas em uma parte linear J_L:

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & -l_3 \sin(\theta_3) \\ 0 & 1 & l_3 \cos(\theta_3) \end{bmatrix}}_{J_L} \begin{bmatrix} \dot{d}_1 \\ \dot{d}_2 \\ \dot{\theta}_3 \end{bmatrix} \tag{43}$$

e angular J_A:

$$[\dot{\alpha}] = \underbrace{[0 \quad 0 \quad 1]}_{J_A} \begin{bmatrix} \dot{d}_1 \\ \dot{d}_2 \\ \dot{\theta}_3 \end{bmatrix} \tag{44}$$

É necessário também calcular todas as matrizes Jacobianas totais $J_L^{(i)}$ e $J_A^{(i)}$ (Asada e Slotine, 1985) que relacionam a velocidade linear e angular do centro de massa de cada elo às velocidades das juntas.

Para a junta 1 tem-se que:

$$\begin{pmatrix} x_{c1} \\ y_{c1} \end{pmatrix} = \begin{pmatrix} d_1 - l_{c1} \\ 0 \end{pmatrix} \rightarrow J_L^{(1)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{45}$$

$$\alpha_1 = 0° \rightarrow J_A^{(1)} = [0 \quad 0 \quad 0] \tag{46}$$

para a junta 2:

$$\begin{pmatrix} x_{c2} \\ y_{c2} \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 - l_{c2} \end{pmatrix} \rightarrow J_L^{(2)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{47}$$

$$\alpha_2 = 0° \rightarrow J_A^{(2)} = [0 \quad 0 \quad 0] \tag{48}$$

e por último os termos relacionados à junta 3:

$$\begin{pmatrix} x_{c3} \\ y_{c3} \end{pmatrix} = \begin{pmatrix} d_1 + l_{c3} \cos(\theta_3) \\ d_2 + l_{c3} \sin(\theta_3) \end{pmatrix} \rightarrow J_L^{(3)} = \begin{bmatrix} 1 & 0 & -l_{c3} \sin(\theta_3) \\ 0 & 1 & l_{c3} \cos(\theta_3) \end{bmatrix} \tag{49}$$

$$\alpha_3 = \theta_3 \rightarrow J_A^{(3)} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \tag{50}$$

Pode-se então determinar a matriz de inércia do manipulador, expressa de maneira geral como $H = \sum_{i=1}^{n} \left(J_L^{i\,T} \cdot J_L^i \cdot m_i + J_A^{i\,T} \cdot I_i \cdot J_A^i \right)$. Para o caso do manipulador, tem-se que

$$H = \begin{bmatrix} m_1 + m_2 + m_3 & 0 & -m_3 l_{c3} \sin(\theta_3) \\ 0 & m_2 + m_3 & m_3 l_{c3} \cos(\theta_3) \\ -m_3 l_{c3} \sin(\theta_3) & m_3 l_{c3} \cos(\theta_3) & m_3 l_{c3}^2 + I_3 \end{bmatrix} \tag{51}$$

Para o cálculo dos termos h, também presentes nas equações de movimento, sabe-se de maneira geral que

$$h_{ijk} = \frac{\partial H_{ij}}{\partial q_k} - \frac{1}{2} \frac{\partial H_{jk}}{\partial q_i} \tag{52}$$

Sendo assim, para o caso da mesa XYθ, os seis termos são expressados como:

$$\begin{aligned} h_{133} &= -m_3 l_{c3} \cos(\theta_3) & h_{233} &= -m_3 l_{c3} \sin(\theta_3) \\ h_{313} &= -m_3 l_{c3} \cos(\theta_3) / 2 & h_{323} &= -m_3 l_{c3} \sin(\theta_3) / 2 \\ h_{331} &= m_3 l_{c3} \cos(\theta_3) / 2 & h_{332} &= m_3 l_{c3} \sin(\theta_3) / 2 \end{aligned} \tag{53}$$

Sabendo que neste caso os elos encontram-se paralelos a mesa, supõem-se os efeitos gravitacionais (representados pelos termos G_i) desprezíveis. Sendo assim, todos os termos necessários para o cálculo das equações de movimento já foram determinados e, a partir da equação (46) são obtidas as equações de movimento:

$$F_1 = (m_1 + m_2 + m_3)\ddot{d}_1 - m_3 l_{c3} \sin(\theta_3)\ddot{\theta}_3 - m_3 l_{c3} \cos(\theta_3)\dot{\theta}_3^{\,2} \tag{54}$$

$$F_2 = (m_2 + m_3)\ddot{d}_2 + m_3 l_{c3} \cos(\theta_3)\ddot{\theta}_3 - m_3 l_{c3} \sin(\theta_3)\dot{\theta}_3^{\,2} \tag{55}$$

$$\tau_3 = -m_3 l_{c3} \sin(\theta_3)\ddot{d}_1 + m_3 l_{c3} \cos(\theta_3)\ddot{d}_2 + (I_3 + m_3 l_{c3}^2)\ddot{\theta}_3 \tag{56}$$

Uma vez determinadas as equações da dinâmica dos três elos, é possível simular a trajetória do manipulador. Neste trabalho, no entanto, não foram realizadas simulações, pois as equações referentes à parte de visão computacional não poderiam ser simuladas. Isso porque a simulação dependeria das imagens obtidas durante a execução do controle em tempo real.

Para aplicar acelerações constantes a_x (linear) na junta 1, a_y (linear) na junta 2, e α' (angular) na junta 3 é necessário utilizar um controlador não linear para cancelar os efeitos dinâmicos. Deseja-se controlar uma trajetória com valores desejados $d_{1d}(t) = a_x t^2 / 2$, $d_{2d}(t) = a_y t^2 / 2$ e $\theta_{3d}(t) = \alpha' t^2 / 2$ supondo sem perda de generalidade, a mesa inicialmente parada na posição $(d_1, d_2, \theta_3) = (0,0,0)$ no instante $t = 0$. Pode-se então determinar a lei não-linear de controle de torque computado combinado com um controle PID de ganhos K_{Pi}, K_{Ii} e K_{Di}. Sabendo que a lei de controle PID, obedece à formulação

$$u_i = K_{Pi}(d_{id} - d_i) + K_{Di}(\dot{d}_{id} - \dot{d}_i) + K_{Ii}\int (d_{id} - d_i)dt \tag{57}$$

tem-se para os três elos

$$u_1 = K_{P1}\left(\frac{a_x t^2}{2} - d_1\right) + K_{D1}(a_x t - \dot{d}_1) + K_{I1}\int (a_x - \ddot{d}_1)dt \tag{58}$$

$$u_2 = K_{P2}\left(\frac{a_y t^2}{2} - d_2\right) + K_{D2}(a_y t - \dot{d}_2) + K_{I2}\int (a_y - \ddot{d}_2)dt \tag{59}$$

$$u_3 = K_{P3}\left(\frac{\alpha' t^2}{2} - \theta_3\right) + K_{D3}(\alpha' t - \dot{\theta}_3) + K_{I3}\int (\alpha' - \ddot{\theta}_3)dt \tag{60}$$

Obtidas as leis de controle, pode-se então calcular o torque a ser enviado a cada junta:

$$F_1 = (m_1 + m_2 + m_3)u_1 - m_3 l_{c3}\sin(\theta_3)u_3 - m_3 l_{c3}\cos(\theta_3)\dot{\theta}_3^2 \tag{61}$$

$$F_2 = (m_2 + m_3)u_2 + m_3 l_{c3}\cos(\theta_3)u_3 - m_3 l_{c3}\sin(\theta_3)\dot{\theta}_3^2 \tag{62}$$

$$\tau_3 = -m_3 l_{c3}\sin(\theta_3)u_1 + m_3 l_{c3}\cos(\theta_3)u_2 + (I_3 + m_3 l_{c3}^2)u_3 \tag{63}$$

O modelo apresentado acima representa o controle dinâmico do manipulador, conhecido também como controle de Torque Computado. Através deste são considerados efeitos não-lineares da dinâmica do robô originários da velocidade dos elos ou da influência entre eles. No entanto estes efeitos não são significativos na maioria dos sistemas *eye-in-hand*, cujas velocidades são limitadas pelo tempo de processamento das imagens. Sendo assim, para este trabalho foi apenas considerado o controle PID, utilizando apenas o modelo cinemático do manipulador.

4.3.
Sistema eletrônico

O sistema eletrônico desenvolvido para este projeto encarrega-se de receber uma posição desejada do computador, processá-la, e enviar o torque necessário para os motores.

Duas maneiras de se fazer um controle eletrônico de atuadores pelo computador foram consideradas. Na primeira, uma porta paralela é utilizada para controlar conversores Analógicos/Digitais que fornecem uma tensão analógica de referência para que um circuito de alta potência faça a amplificação de corrente e/ou tensão.

Na segunda, uma porta de comunicação serial (USB, RS232, etc.) é utilizada para enviar valores que serão interpretados por uma eletrônica micro-controlada (provida de uma interface de comunicação serial) que comandará circuitos de alta potência para ativação dos atuadores. Neste caso, o microcontrolador utilizado pode inclusive ser programado para receber do computador somente as posições desejadas das juntas, executando o controle PID internamente.

Desenvolveu-se para este trabalho uma placa que se comunica com o computador via serial, contendo esta um microcontrolador (Apêndice II) responsável por calcular e enviar torques para os motores. A placa eletrônica desenvolvida funciona como uma interface entre o computador e os motores de ativação da mesa. A Figura 25 apresenta um Esquema da placa, e através dela nota-se a presença do conector serial (identificado pelo número 1) e do micro-controlador (identificado pelo número 2). Foram incluídos também botões de ajuste manual de posição. Cada par de botões é responsável pelo acionamento de um motor no sentido positivo e negativo. Os seis botões (dois para cada um dos três elos) também estão identificados na Figura 25 pelo número 3.

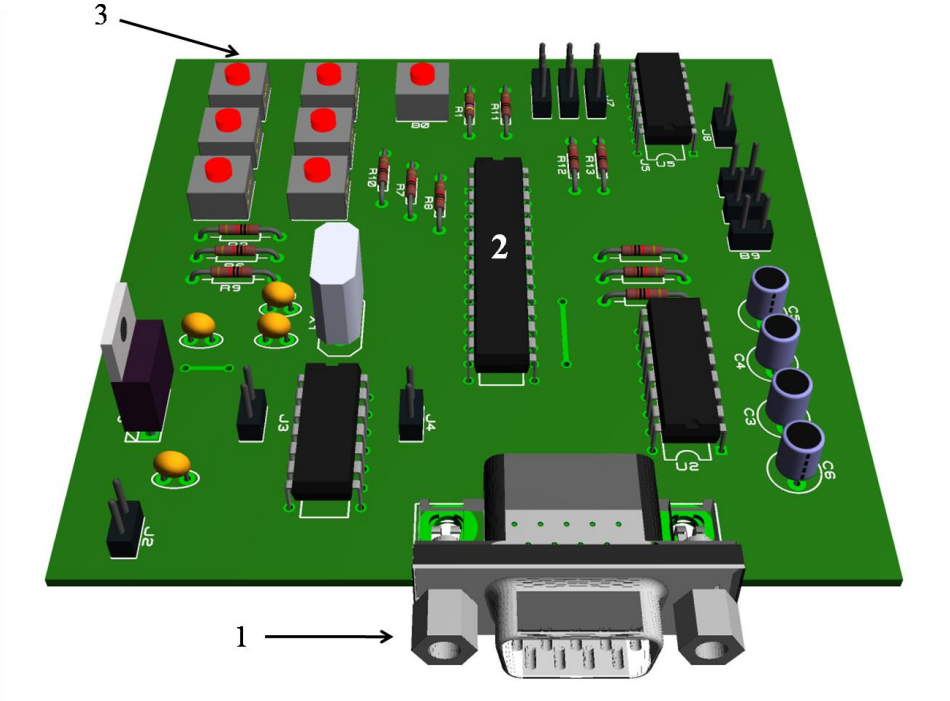


Figura 25 - Placa eletrônica

O sinal de saída da eletrônica é do tipo PPM (*Pulse Position Modulation*). Este sinal foi escolhido porque diversos sistemas de potência disponíveis no mercado utilizam-no como sinal de ativação. Desta forma, a mesma eletrônica pode ser utilizada em diferentes sistemas de potência, se adequando assim a corrente drenada por diferentes motores DC.

Para ativação dos motores DC, foram utilizados controladores de velocidade Banebots® BB-12-45, que suportam correntes contínuas de 12A, 45A de pico, e são ativados por sinais PPM vindos diretamente da placa eletrônica.

Figura 26 - Controlador de Velocidade *Banebots*

Neste projeto serão estudadas duas diferentes técnicas de controle (conforme visto no Capítulo 3). O controle *look-and-move* se utiliza de sensores de posição para realimentar o sistema. Já o controle servo-visual desconhece a posição da mesa no espaço e utiliza apenas o erro entre o objeto e a câmera (obtido através de imagens) para levar a mesa coordenada até a posição desejada. Para que a eletrônica desenvolvida pudesse se adequar às duas técnicas, foram implementadas pequenas mudanças no microcontrolador responsável pelo controle. A implementação no sistema eletrônico de ambas as técnicas serão descritas nas seções seguintes.

4.3.1. Sistema com realimentação por sensores de posição (Controle *Look-and-Move*)

Uma vez recebida do computador a informação da posição desejada, calculada a partir da imagem da câmera na posição inicial, o microcontrolador lê a posição atual dos motores através dos sensores de posição das juntas, e calcula através do controle PID o torque a ser enviado para cada motor. Este ciclo se repete a cada instante de tempo, até que a mesa atinja a posição desejada. Este processo é inteiramente realizado dentro do microcontrolador, sem que seja necessária qualquer outra intervenção do *software* do computador, e sem a captura de novas imagens. Se, durante a execução da tarefa, alguma outra informação for enviada à eletrônica, o controle automaticamente muda de trajetória, dirigindo-se então à nova posição desejada. Vale ressaltar que as constantes K_P, K_I e K_D, necessárias para o controle PID, são informadas continuamente à placa eletrônica juntamente com a informação da posição desejada, permitindo assim que os ganhos sejam modificados se necessário.

4.3.2. Sistema com realimentação por imagem (Controle Servo-Visual)

No controle servo-visual, o erro entre a posição desejada e a real não é obtido no microcontrolador, e sim no computador. Isso acontece pois o erro entre a posição desejada e a real é obtido através das imagens, e varia ao longo da

movimentação da mesa. Para este caso, o cálculo do controle PID é feito através do *software* do computador, e apenas o resultado é enviado à placa eletrônica.

Para comportar os dois sistemas de controle, a formulação do PID utilizada no microcontrolador foi acrescentada de um novo fator:

$$u_i = K_{Pi}(d_{id} - d_i) + K_{Di}(\dot{d}_{id} - \dot{d}_i) + K_{Ii}\int(d_{id} - d_i)dt + T_i \qquad (64)$$

onde T_i é um torque calculado pelo *software* e enviado à eletrônica. Para que este torque seja enviado diretamente aos motores sem ser alterado pela rotina PID residente no micro-controlador, é necessário que os valores das posições desejadas e as constantes K_P, K_I e K_D enviados ao sistema eletrônico sejam nulos, resultando em $u_i = T_i$.

No caso do controle *look-and-move*, os valores das posições desejadas e as constantes do controle PID são enviados normalmente, com a diferença que para este caso os valores de T_i devem ser iguais a zero. Na Tabela 1 estão listados, em ordem de envio, os parâmetros enviados à eletrônica, variando com as técnicas de controle.

Tabela 1 - Parâmetros enviados através da porta serial à eletrônica desenvolvida

Parâmetros enviados à eletrônica		**Controle *look-and-move***	**Controle Servo-visual**
Posição desejada	**Elo1**	$x_d \neq 0$	$x_d = 0$
	Elo2	$y_d \neq 0$	$y_d = 0$
	Elo3	$\theta_d \neq 0$	$\theta_d = 0$
Ganhos (iguais para os três elos)	**Proporcional**	$K_P \neq 0$	$K_P = 0$
	Integral	$K_I \neq 0$	$K_I = 0$
	Derivativo	$K_D \neq 0$	$K_D = 0$
Torque enviado aos motores	**Elo1**	$T_1 = 0$	$T_1 \neq 0$
	Elo 2	$T_2 = 0$	$T_2 \neq 0$
	Elo 3	$T_3 = 0$	$T_3 \neq 0$

Vale ressaltar que os ganhos utilizados no controle servo-visual não estão representados na tabela acima. Seus valores diferentes de zero, são calibrados dentro do software, para gerar os torques T_1, T_2 e T_3.

Tanto para o controle *look-and-move* quanto o servo-visual, o controle PID foi empregado em sua forma discretizada. O taxa de amostragem varia com a frequência ora do microcontrolador (controle *look-and-move*), ora do computador (controle servo-visual). A formulação geral do controle PID discreto é

$$u(t) = K_P\, e(t) + \frac{K_D}{\Delta t}\big(e(t) - e(t-1)\big) + K_I \Delta t \sum e(t) \qquad (65)$$

onde o erro $e(t)$ representa a diferença entre a posição desejada e a atual. Conforme a equação (65), o termo derivativo pode ser representado como a diferença entre o erro no instante atual e o instante anterior. Já o termo integral representa um acumulador, em que a cada instante de tempo é acrescido do valor do erro.

4.4. *Software* de controle

Para a implementação do *software* de controle no computador optou-se por utilizar o programa Matlab® da empresa *MathWorks,* devido à sua vasta biblioteca de visão computacional, e também pela facilidade de comunicação com a porta serial. O *software* criado poderá ser de grande ajuda na área de pesquisa, servindo como ferramenta de estudo.

A tela principal do programa, apresentada na Figura 27, contém todas as opções de controle a serem executadas. No canto direito da tela, uma chave indica se o alvo utilizado será um círculo vermelho (utilizado nos primeiros testes), ou se será utilizada uma imagem de um painel de válvulas típico de intervenções submarinas (simulando um painel real).

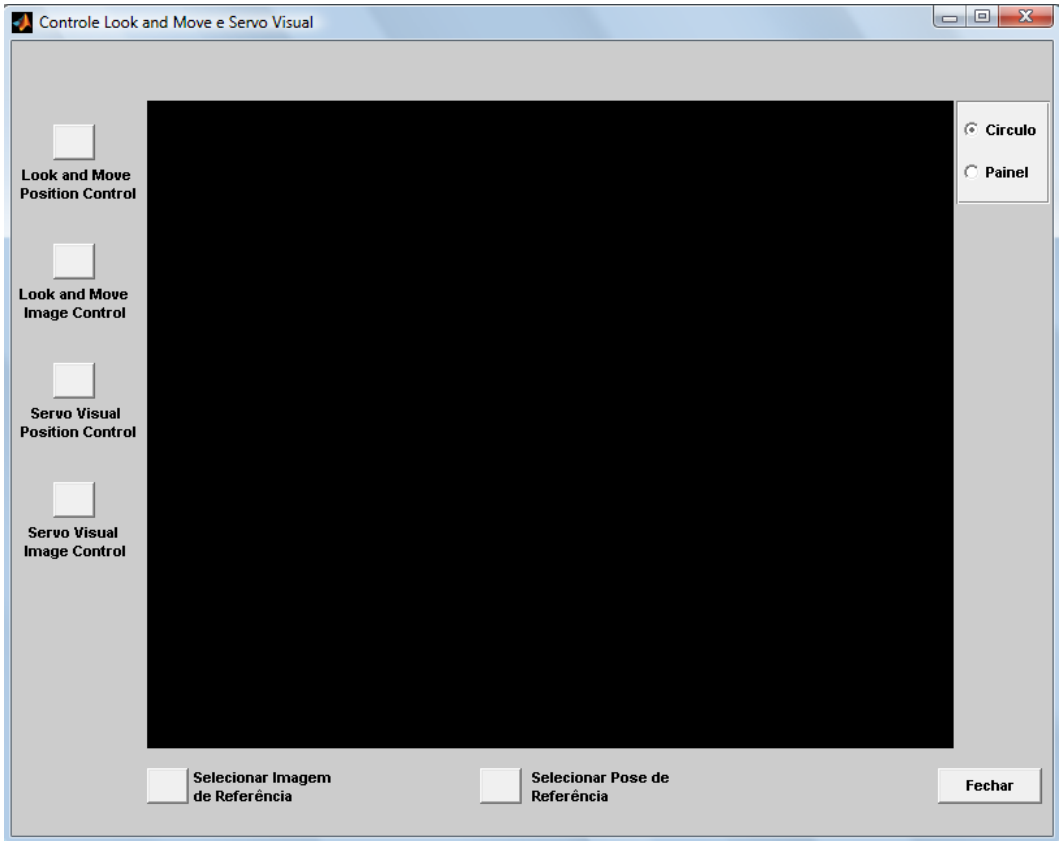


Figura 27 - Tela principal do *software*

Na parte inferior da tela, dois botões permitem ao usuário a troca das variáveis de referência. No caso do controle baseado em imagem, uma janela permite a escolha da imagem de referência (Figura 28). Ou seja, a imagem da câmera que se deseja observar quando o sistema atingir a posição final. No caso do controle baseado em pose, uma janela permite ao usuário que indique as posições desejadas em x, y e θ, representadas relativamente ao objeto de interesse (Figura 29).

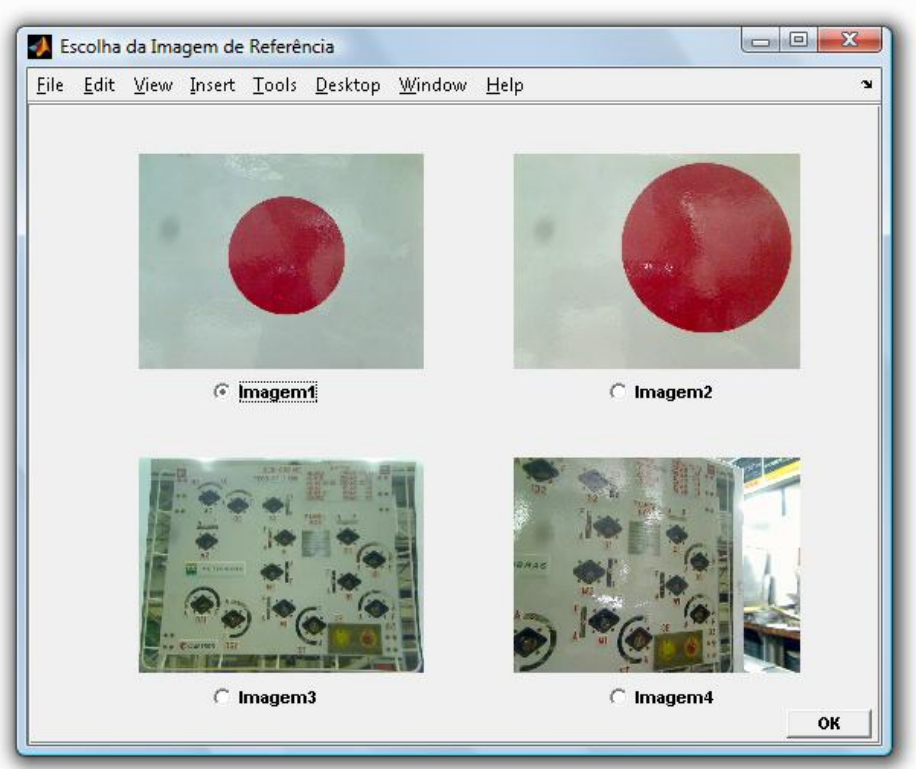


Figura 28 - Janela de seleção da imagem de referência (controle baseado em imagem)

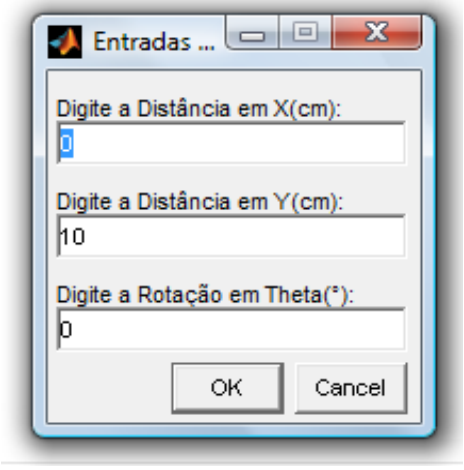


Figura 29 - Janela de distâncias desejadas do objeto de interesse (controle baseado em pose)

Uma vez escolhida a imagem ou pose de referência, e se o controle irá se aplicar ao círculo ou ao painel, o usuário deverá escolher na lateral esquerda da tela entre as opções de controle:

- Controle *look-and-move* baseado em pose

- Controle *look-and-move* baseado em imagem
- Controle servo-visual baseado em pose
- Controle servo-visual baseado em imagem

A seguir será apresentado um passo a passo do programa, descrevendo todas as principais etapas efetuadas pelo *software* de controle.

- Passo1: Captura uma imagem da câmera.
- Passo2: Controle baseado em pose ou em imagem.
    - Baseado em pose:
        - Processa imagem obtida no passo1, e obtém coordenadas relativas entre a câmera e o objeto: $x_{rel}, y_{rel}, \theta_{rel}$.
        - Usuário determina posição de referência desejada $x_{ref}, y_{ref}, \theta_{ref}$.
        - Se controle escolhido for ***look-and-move***:
            - ✓ Lê posição atual dos *encoders* x_0, y_0, θ_0.
            - ✓ Calcula posição desejada: $x_d = x_0 + (x_{rel} - x_{ref})$, $y_d = y_0 + (y_{rel} - y_{ref})$, $\theta_d = \theta_0 + (\theta_{rel} - \theta_{ref})$.
            - ✓ Envia parâmetros para eletrônica.
            - ✓ Fim.
        - Se controle escolhido for **servo-visual**:
            - ✓ Obtém-se o erro entre a câmera e o objeto: $e_x = x_{rel} - x_{ref}$, $e_y = y_{rel} - y_{ref}$, $e_\theta = \theta_{rel} - \theta_{ref}$.
            - ✓ Cálculo do controle PID, obtendo torques para as 3 juntas.
            - ✓ Envia parâmetros para eletrônica.
            - ✓ Se erro em alguma das 3 juntas for maior que um limiar, retorna ao passo1 para capturar uma nova imagem, senão, fim.
    - Baseado em imagem
        - Determinam-se as características da imagem obtida no Passo1: $\$_1, \$_2, \$_3$.

- Compara características obtidas com as características da imagem de referência $\$_{1ref}, \$_{2ref}, \$_{3ref}$.
- A partir da diferença entre as características desejadas e as reais, utiliza-se a matriz Jacobiana de imagem, definida na seção 4.6, para obter as coordenadas relativas entre a câmera e o objeto $\begin{pmatrix} x_{rel} \\ y_{rel} \\ \theta_{rel} \end{pmatrix} = J^* \cdot \begin{pmatrix} \$_1 - \$_{1ref} \\ \$_2 - \$_{2ref} \\ \$_3 - \$_{3ref} \end{pmatrix}$.
- Se controle escolhido for ***look-and-move***:
  - ✓ Lê posição atual dos *encoders* x_0, y_0, θ_0.
  - ✓ Calcula posição desejada: $x_d = x_0 + x_{rel}$, $y_d = y_0 + y_{rel}$, $\theta_d = \theta_0 + \theta_{rel}$.
  - ✓ Envia parâmetros para eletrônica.
  - ✓ Fim.
- Se controle escolhido for **servo-visual**:
  - ✓ Obtém-se erro entre a câmera e o objeto $e_x = x_{rel}$, $e_y = y_{rel}$, $e_\theta = \theta_{rel}$.
  - ✓ Inicia cálculo do controle PID, e obtém torques para as 3 juntas.
  - ✓ Envia parâmetros para eletrônica.
  - ✓ Se erro em alguma das 3 características $\$_1$, $\$_2$ ou $\$_3$ em relação a $\$_{1ref}$, $\$_{2ref}$ ou $\$_{3ref}$ for maior que um limiar, retorna ao passo1, senão, fim.

Vale ressaltar que durante o controle *look-and-move* a posição inicial (x_0, y_0, θ_0) medida pelos sensores de posição das juntas não precisa ser utilizada no *software* do computador, pois ela é computada diretamente na placa eletrônica.

4.5. Processamento da Imagem

Durante os testes realizados neste trabalho, foram utilizados dois objetos alvos distintos. O primeiro objeto utilizado foi um círculo (vermelho) impresso sobre uma superfície vertical, a fim de facilitar o processamento da visão computacional. Em seguida, foram implementadas novas técnicas que permitiram ampliar o programa, de forma que este funcionasse também utilizando qualquer outro objeto plano como referência. Para validar os experimentos com um alvo genérico, utilizou-se uma imagem de um painel de válvulas submarino, simulando assim uma aplicação real para o projeto. A seguir serão descritos os processamentos das imagens para os dois casos. Primeiro serão apresentados os equacionamentos utilizados no caso do círculo e em seguida para o caso genérico de um objeto plano.

4.6. Equacionamento do sistema com alvo circular

A fim de minimizar o esforço computacional nesta etapa do projeto, foi escolhido um disco circular vermelho, liso, em um plano vertical, e que pudesse ser de fácil reconhecimento na imagem. A cor vermelha foi escolhida aleatoriamente por se tratar de uma das cores primárias, uma vez utilizando o padrão RGB. A partir então da relação entre semi-eixos da elipse resultante na imagem, as distâncias em x, y, θ podem ser conhecidas. Na Figura 30 pode-se observar os parâmetros relevantes neste experimento.

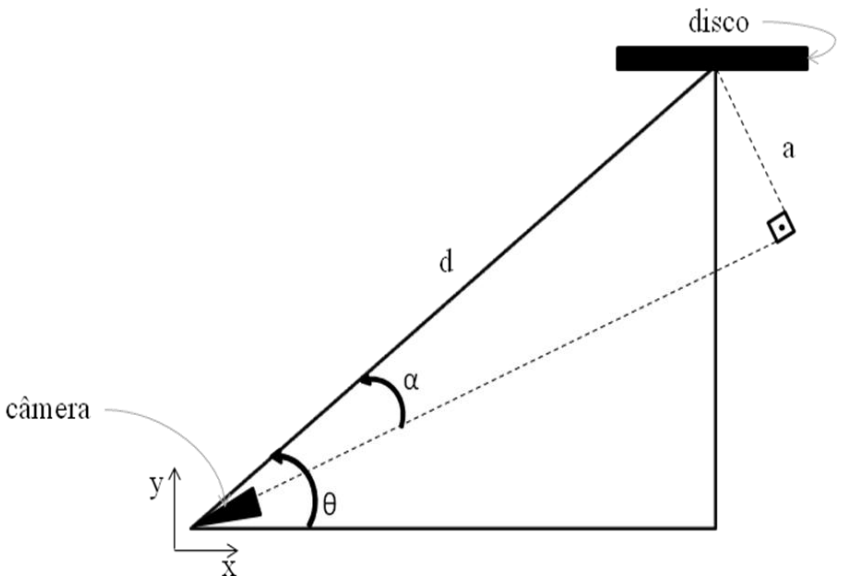


Figura 30 - Esquema dos testes com disco

A Figura 30 mostra a vista superior do disco e da mesa, e admite-se o disco com eixo axial na direção y; x e y representam os eixos de movimentação da mesa; d é a distância entre o centro da câmera e o centro do disco; a é a distância entre o centro do disco e o eixo óptico da câmera; θ é o ângulo entre o eixo x e o segmento que contém d; e α é o ângulo entre o eixo óptico e o segmento que contém d. Note que quando o disco encontra-se centralizado na imagem, $a = 0$.

Pela Figura 30, podem ser deduzidas as seguintes equações:

$$x = d\cos\theta \tag{66}$$

$$y = d\,\mathrm{sen}\,\theta \tag{67}$$

$$\mathrm{sen}\,\alpha = \frac{a}{d} \tag{68}$$

Para simplificar as equações, define-se também:

$$\theta' = \theta - \alpha \tag{69}$$

Quando o eixo óptico da câmera atravessa o centro do disco, pode-se perceber uma distância i_a entre o centro do disco e o centro da imagem. Pela Figura 31 é possível perceber também que quando a câmera não se encontra perpendicular ao disco, este pode ser representado na imagem por uma elipse com semi-eixos menor i_r e maior i_R.

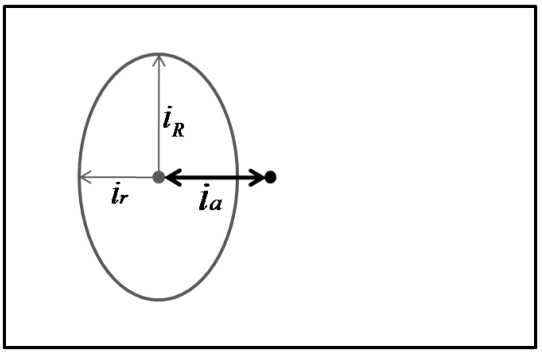


Figura 31 - Imagem obtida da câmera com disco deslocado do centro

Assumindo que o centro da câmera estará sempre na mesma altura que o centro do disco, uma vez que a mesa não se desloca na vertical, pode-se afirmar que o semi-eixo maior só irá variar de tamanho na imagem quando houver deslocamento em y. Ou seja, quando a câmera se aproximar do disco i_R aumenta, e quando a câmera se afastar este diminui. Essa afirmativa só é válida, no entanto, se $a << d$, resultando em

$$d = \frac{K}{i_R} \tag{70}$$

sendo K uma constante. Caso a não seja muito menor que d, a expressão acima se torna uma aproximação, adotada aqui por simplicidade, pois nos experimentos conduzidos o valor desejado de a é zero, satisfazendo $a = 0 << d$.

Admitindo-se o disco descentralizado da imagem, estabelece-se que:

$$\frac{a}{r} = \frac{i_a}{i_R} \tag{71}$$

onde r é o raio (real) do disco. Para uma imagem centralizada, $i_a = 0$, logo $a = 0$.

Para determinar o ângulo de inclinação do disco em relação à câmera, utiliza-se a razão entre seus semi-eixos na imagem. Observando a Figura 32, onde a imagem da esquerda representa o disco visto de frente, e a imagem da direita a visão superior, pode-se estabelecer que

$$i_r = i_R \operatorname{sen}\theta \Rightarrow \theta = \operatorname{sen}^{-1}\left(\frac{i_r}{i_R}\right) \tag{72}$$

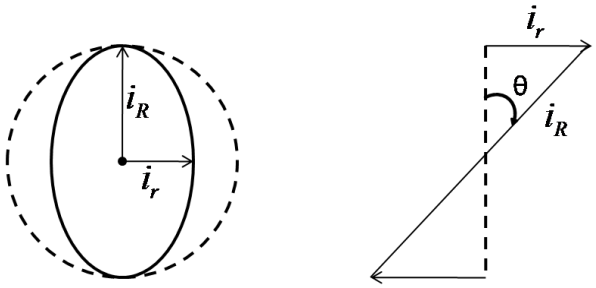


Figura 32 - Vista frontal e superior do disco

Substituindo as equações (68) e (72) em (69), é possível determinar o valor de θ', obtendo assim os valores necessários para a atuação dos 3 elos x, y e θ, onde $\theta = \theta' + \alpha$. Estes valores, que representam os deslocamentos e a rotação da câmera em relação ao disco, são suficientes para permitir que um controle baseado em pose seja executado.

Para as técnicas de controle baseadas em imagem, as variáveis desejadas são características a serem definidas pelas variáveis $\$_1$, $\$_2$ e $\$_3$. Tais características são escritas, segundo quaisquer características providas da imagem. Para este objeto alvo circular, as variáveis serão escolhidas em função de i_a, i_r e i_R.

A fim de facilitar os cálculos, as variáveis $\$_1$, $\$_2$ e $\$_3$ serão definidas como $\$_1 = \frac{1}{i_R}$, $\$_2 = \frac{i_r}{i_R}$, $\$_3 = \frac{i_a}{i_R}$.

Reescrevendo as equações já obtidas em função das novas variáveis, tem-se que a equação (70) passa a ser escrita como $d = \frac{K}{i_R} = K\$_1$ e a equação (72) por $\sin(\theta) = \left(\frac{i_r}{i_R}\right) = \$_2$. Sendo assim, os valores de x e y, descritos anteriormente nas equações (66) e (67), passam a ser escritos como:

$$x = K\,\$_1\sqrt{1-\$_2^2} \tag{73}$$

$$y = K\,\$_1\,\$_2 \tag{74}$$

A equação (71), $\frac{a}{r} = \frac{i_a}{i_R}$,pode ser reescrita como $\$_3 = \frac{a}{r}$; a equação (68), $\text{sen}\,\alpha = \frac{a}{d}$, como $\text{sen}\,\alpha = \frac{r\,\$_3}{K\,\$_1}$; e, por último, a equação (69), $\theta' = \theta - \alpha$, como $\theta' = \text{sen}^{-1}(\$_2) - \text{sen}^{-1}\left(\frac{r\,\$_3}{K\,\$_1}\right)$.

Sendo $J_{\$p}$ a matriz jacobiana de imagem responsável pela transformação entre o vetor de características $\$ = (\$1 \quad \$_2 \quad \$_3)^T$ e o vetor de parâmetros desejados $p = (d \quad \theta \quad \alpha)^T$, e J_{qp} a matriz Jacobiana de transformação entre o vetor de coordenadas desejadas q e o vetor de parâmetros desejados p, tem-se que

$$\underbrace{\begin{pmatrix} \delta\$_1 \\ \delta\$_2 \\ \delta\$_3 \end{pmatrix}}_{\delta\$} = J_{\$p} \underbrace{\begin{pmatrix} \delta d \\ \delta\theta \\ \delta a \end{pmatrix}}_{\delta p} \tag{75}$$

$$\underbrace{\begin{pmatrix} \delta x \\ \delta y \\ \delta\theta' \end{pmatrix}}_{\delta q} = J_{qp} \underbrace{\begin{pmatrix} \delta d \\ \delta\theta \\ \delta a \end{pmatrix}}_{\delta p} \tag{76}$$

Com isso é possível formular a matriz que relaciona variações das coordenadas desejadas com variações do vetor de características do objeto

$$\begin{pmatrix} \delta x \\ \delta y \\ \delta\theta' \end{pmatrix} = J_{qp}\, J_{\$p}^{-1} \begin{pmatrix} \delta\$_1 \\ \delta\$_2 \\ \delta\$_3 \end{pmatrix} \tag{77}$$

sendo

$$J_{\$p} = \begin{bmatrix} 1/K & 0 & 0 \\ 0 & \cos\theta & 0 \\ 0 & 0 & 1/r \end{bmatrix} \rightarrow J_{\$p}^{-1} = \begin{bmatrix} K & 0 & 0 \\ 0 & \sec\theta & 0 \\ 0 & 0 & r \end{bmatrix} \tag{78}$$

$$J_{qp} = \begin{bmatrix} \text{sen}\,\theta & d\cos\theta & 0 \\ \cos\theta & -d\,\text{sen}\,\theta & 0 \\ \frac{1}{d}\tan(\theta - \theta') & 1 & -\frac{1}{d}\sec(\theta - \theta') \end{bmatrix} \tag{79}$$

Escrevendo as matrizes acima em relação aos novos parâmetros $\$_1$, $\$_2$ e $\$_3$, tem-se que:

$$J_{\$p}^{-1} = \begin{bmatrix} K & 0 & 0 \\ 0 & \dfrac{1}{\sqrt{1-\$_2^2}} & 0 \\ 0 & 0 & r \end{bmatrix} \tag{80}$$

$$J_{qp} = \begin{bmatrix} \sqrt{1-\$_2^2} & -K\$_1\$_2 & 0 \\ \$_2 & K\$_1\sqrt{1-\$_2^2} & 0 \\ \dfrac{r\$_3}{K\$_1\sqrt{K^2\$_1^2 - r^2\$_3^2}} & 1 & -\dfrac{1}{\sqrt{K^2\$_1^2 - r^2\$_3^2}} \end{bmatrix} \tag{81}$$

Sendo assim, conhecidos os valores reais e desejados de $\$_1$, $\$_2$ e $\$_3$, pode-se determinar o vetor $\delta\$ = \begin{pmatrix} \$_1 - \$_{1d} \\ \$_2 - \$_{2d} \\ \$_3 - \$_{3d} \end{pmatrix}$ e consequentemente, a partir da equação (77), determinar os erros referentes às 3 juntas desejadas $\delta x, \delta y, \delta\theta'$. Note que a equação (77) só é exata para erros infinitesimais, portanto a sua aplicação a erros finitos é uma aproximação. Esta aproximação melhora à medida que a câmera se aproxima de seu objetivo quando os erros $\delta\$_1$, $\delta\$_2$ e $\delta\$_3$ tendem para zero.

Vale ressaltar que neste caso a matriz Jacobiana de imagem consegue determinar o vetor de características independente das posições atuais dos motores, sendo assim o sistema não precisa de realimentação dos sensores de posição. Para um sistema genérico é necessário uma realimentação dos motores.

4.7. Equacionamento do sistema com alvo 2D genérico

Para a segunda parte do experimento, o sistema será modificado de forma a operar utilizando como referência alvos planos genéricos. Para fazer o reconhecimento das imagens obtidas pela câmera, será utilizado o algoritmo SIFT (descrito anteriormente na seção 2.2) responsável por determinar pontos chaves na imagem. Através deste algoritmo, também é possível criar pares de pontos correspondentes entre duas imagens de vistas diferentes de um mesmo objeto. A partir destas correspondências, poderão ser obtidas as distâncias relativas entre a

câmera e o objeto. Executando o algoritmo sobre uma imagem de referência, cujas coordenadas dos pontos no espaço são conhecidas, é possível determinar para cada ponto encontrado a relação entre as coordenadas deste na imagem (em *pixels*) e as coordenadas no espaço. Na Figura 33 é representado um objeto plano utilizado como alvo no espaço, onde os pontos indicados representam os pontos chaves encontrados pelo algoritmo. Um sistema de referência (x_o, y_o) é fixado ao objeto, com origem definida em seu centro geométrico.

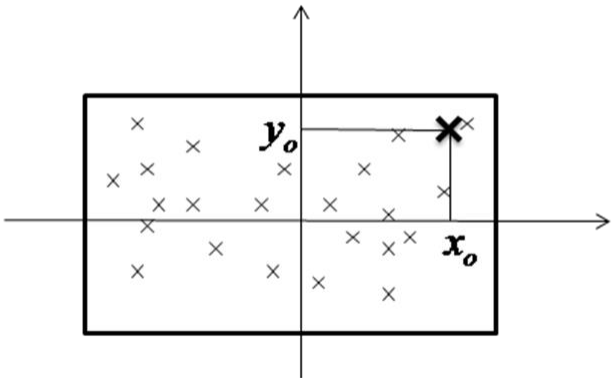


Figura 33 - Pontos chaves (x_o, y_o) encontrados pelo algoritmo SIFT sobre o objeto alvo

A Figura 34 mostra a projeção de um ponto do objeto, a uma distância x_o de seu centro na horizontal, considerando que a câmera se encontra a uma distância (x_c, y_c) do centro do objeto, e formando um ângulo θ entre seu eixo óptico e a direção de y_c.

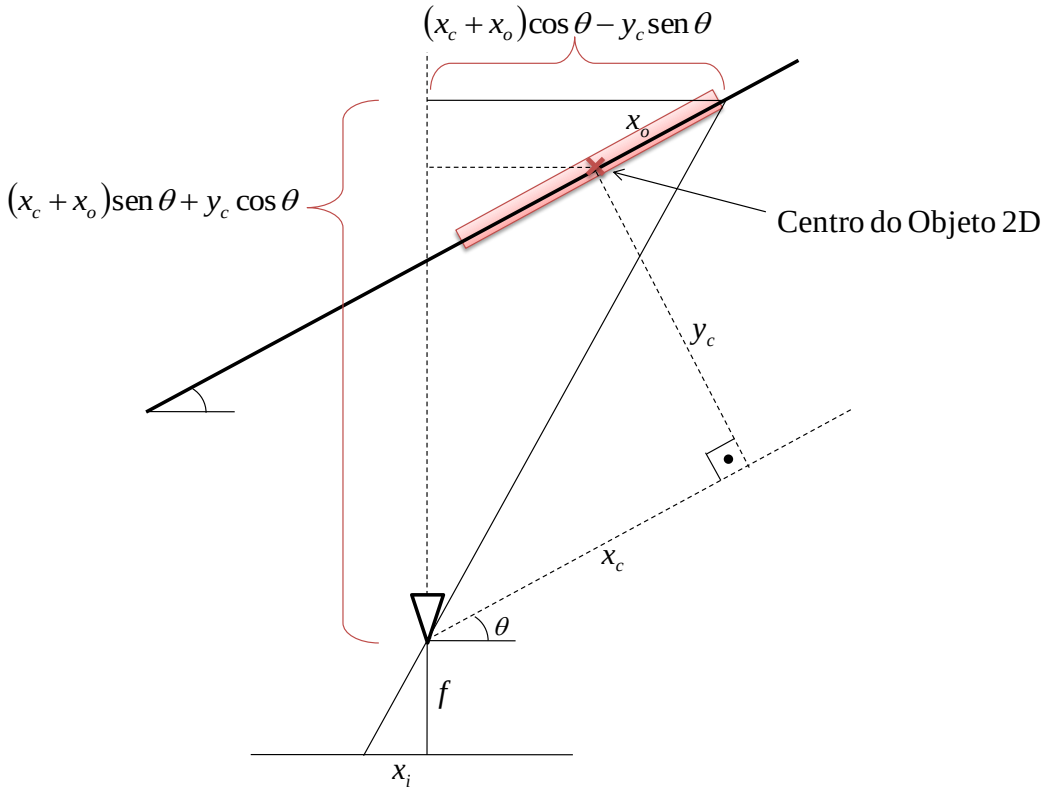


Figura 34 – Esquema do experimento utilizando objetos 2D

Na Figura 34, x_o representa a distância entre um dos pontos chaves e o centro do objeto alvo medida na horizontal. Já x_i representa a mesma distância nas coordenadas da imagem, em *pixels*. Sendo f a distância focal da câmera, expressa em pixels, tem-se que:

$$\frac{(x_c + x_o)\operatorname{sen}\theta + y_c \cos\theta}{f} = \frac{(x_c + x_o)\cos\theta - y_c \operatorname{sen}\theta}{x_i} \tag{82}$$

e consequentemente monta-se o sistema

$$\begin{bmatrix} f & x_i & x_o \cdot x_i \end{bmatrix} \cdot \underbrace{\begin{pmatrix} y_c \tan\theta - x_c \\ x_c \tan\theta + y_c \\ \tan\theta \end{pmatrix}}_{X} = f \cdot x_o \tag{83}$$

Para M pares de pontos (x_o, x_i), o vetor X pode ser determinado por um ajuste de mínimo quadrados

$$\underbrace{\begin{bmatrix} f & x_{i1} & x_{o1}x_{i1} \\ f & x_{i2} & x_{o2}x_{i2} \\ \vdots & \vdots & \vdots \\ f & x_{im} & x_{om}x_{im} \end{bmatrix}}_{A} \cdot X = f \cdot \underbrace{\begin{bmatrix} x_{o1} \\ x_{o2} \\ \vdots \\ x_{om} \end{bmatrix}}_{B} \Rightarrow A \cdot X = B \Rightarrow X = pinv(A) \cdot B \tag{84}$$

onde $pinv(A)$ é a pseudo-inversa da matriz A.

Uma vez determinado o vetor X, podem ser determinadas as distâncias relativas desejadas (x_c, y_c, θ). Sendo $X = [X_1 \quad X_2 \quad X_3]^T$, obtém-se a partir da definição de X

$$\begin{aligned} \theta &= \text{atan}(X_3) + k\pi \\ x_c &= \frac{X_2 \tan\theta - X_1}{1 + \tan^2\theta} \\ y_c &= \frac{X_1 \tan\theta - X_2}{1 + \tan^2\theta} \end{aligned} \tag{85}$$

Note que as equações (84) e (85) dependem dos valores de $x_{o1},...,x_{om}$ para obter x_c, y_c e θ. Esses valores foram obtidos previamente a partir da imagem de referência, capturada em uma posição conhecida da câmera e aplicando a equação (82) para cada um dos M pontos SIFT da imagem de referência.

Os equacionamentos descritos acima foram realizados em função da coordenada x_o dos pontos do objeto. De forma análoga, as mesmas distâncias poderiam ter sido obtidas utilizando as coordenadas em y_o. A partir do sistema:

$$[y_i \quad x_o \cdot y_i] \cdot \underbrace{\begin{pmatrix} x_c \,\text{sen}\,\alpha + y_c \cos\alpha \\ \text{sen}\,\alpha \end{pmatrix}}_{X^*} = F \cdot y_o \tag{86}$$

e utilizando a formulação da pseudo-inversa para M pares de pontos (y_o, y_i):

$$\underbrace{\begin{bmatrix} y_{i1} & x_1 y_{i1} \\ y_{i2} & x_2 y_{i2} \\ \vdots & \vdots \\ y_{im} & x_m y_{im} \end{bmatrix}}_{A^*} \cdot X^* = f \underbrace{\begin{bmatrix} y_{o1} \\ y_{o2} \\ \vdots \\ y_{om} \end{bmatrix}}_{B^*} \Rightarrow X^* = pinv(A^*) \cdot B^* \tag{87}$$

o vetor X^* pode ser determinado. A partir de $X^* = [X_1^* \quad X_2^*]^T$, é possível apenas determinar o par de equações

$$\begin{aligned} \alpha &= \text{asen}\left(\text{X}_2^*\right) + k\pi \\ x_c \,\text{sen}\,\alpha + y_c \cos\alpha &= \text{X}_1^* \end{aligned} \tag{88}$$

Note que ao usar apenas (y_o, y_i) para estimar a pose da câmera em relação ao objeto, obtém-se um sistema indeterminado, com apenas 2 equações para as 3 incógnitas x_c, y_c e θ. Isto ocorre porque o eixo óptico da câmera está sempre em um plano horizontal perpendicular à direção vertical y_o. Assim, na prática apenas os pares (x_o, x_i) são usados na obtenção de (x_c, y_c, θ). As equações (86-88) são usadas apenas para conferir os resultados.

No capítulo seguinte, serão descritos os testes realizados tanto com o alvo em forma de círculo quanto com a imagem do painel, e serão apresentados gráficos, quantificando os resultados. Por fim, será apresentada uma comparação entre as técnicas *look-and-move* e servo-visual mostrando as vantagens e desvantagens de cada uma.

5
Resultados

Este capítulo irá apresentar os testes realizados para validar o trabalho. Os primeiros testes serão realizados utilizando um círculo vermelho como objeto alvo. Para os testes seguintes, será utilizada uma imagem de um painel de válvulas típico de intervenções submarinas.

Devido à presença de *encoders* acoplados aos motores, é possível determinar a posição de cada elo em tempo real, mesmo que estes valores não tenham sido utilizados no algoritmo de controle. Assim, para cada experimento realizado, será traçado um gráfico da posição das juntas em relação ao tempo.

Os testes são divididos em quatro etapas: as duas primeiras utilizando o círculo vermelho e as duas últimas utilizando o painel. Para ambos os casos será avaliado o "posicionamento frontal" (onde a câmera visualiza o alvo com o eixo óptico da câmera alinhado ao centro do objeto), e o "posicionamento lateral" em relação ao alvo (onde a câmera avista ou uma parte do objeto, ou a lateralmente utilizando a rotação da câmera).

Durante os testes com o painel, percebeu-se que o tempo de processamento do algoritmo SIFT crescia muito com o tamanho da imagem. O controle servo-visual depende da freqüência do processamento da imagem para enviar os torques aos motores. Inicialmente foram utilizadas imagens de 960 x 720 pixels, porém nesta resolução o tempo de processamento do algoritmo SIFT era em média 5 segundos, gerado a partir do software *Matlab* em um notebook (processador Intel® Core™2 Duo de 2.0 GHz, 2048MB de memória RAM e placa de vídeo Mobile Intel® 965 Express Chipset Family). Desta forma, o sistema não conseguia corrigir adequadamente pequenos erros de posição e assim não convergia até a posição desejada. A solução encontrada foi diminuir a resolução das imagens obtidas pela câmera para 352 x 288 pixels, somente nos casos de controle servo-visual. Para os testes do controle *look-and-move*, foram mantidas as imagens de 960 x 720 pixels, uma vez que esse controle se baseia apenas em

uma imagem (e quanto maior a qualidade da imagem, mais precisos serão os resultados).

Na Figura 35, um gráfico apresenta a resolução em *pixels* de uma imagem e seu tempo de processamento pelo algoritmo SIFT. Vale ressaltar que a implementação do algoritmo utilizado foi obtida no site oficial do criador David Lowe e, após análise entre diferentes implementações disponíveis, constatou-se que esta, além de mais fiel ao algoritmo original, era também uma das mais rápidas.

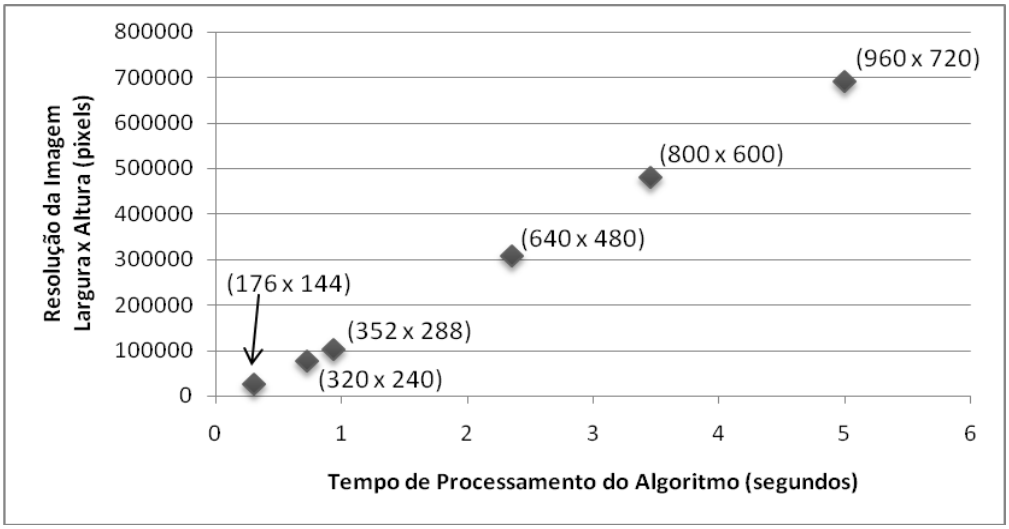


Figura 35 – Resolução em *pixels versus* tempo de processamento do algoritmo SIFT

Vale ressaltar também que o tempo necessário para a mesa atingir uma dada posição depende diretamente da velocidade máxima dos motores. Durante o controle *look-and-move* a velocidade utilizada foi máxima, porém no controle servo-visual a velocidade foi ajustada de acordo com a proximidade do alvo. Fazendo uma estimativa de sua velocidade, conclui-se que a velocidade máxima da mesa (nas coordenadas x e y) é de 13.3 mm/s, ou 1.33 cm/s.

5.1. Experimento 1: Teste frontal utilizando como alvo círculo

Para iniciar os testes, montou-se o experimento apresentado na Figura 36, onde a posição inicial e final são medidas em relação à posição inicial da câmera. Conforme mencionado no Capítulo 4, o equacionamento da pose do círculo foi realizado admitindo-se que a distância entre a câmera e o círculo era muito maior

que a distância entre seu centro e o eixo óptico da câmera. Para evitar a inclusão de erros associados a essa aproximação, os experimentos com o círculo para os controles baseados em pose foram realizadas com o grau de liberdade de rotação da mesa travado, fixado em $\theta = 0°$.

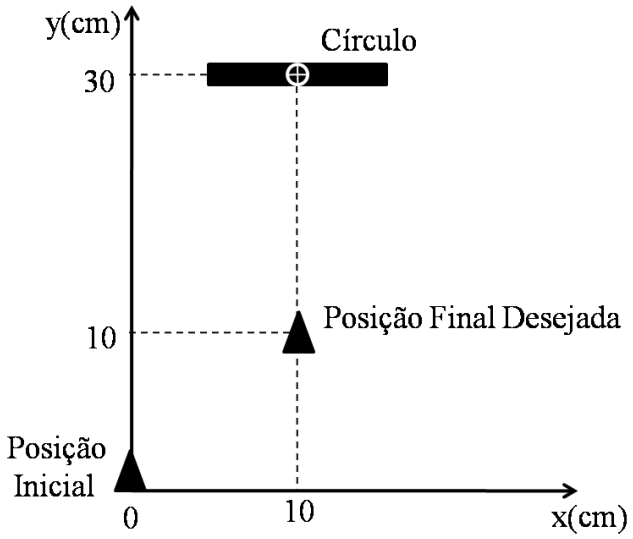


Figura 36 - Vista superior do experimento 1, para teste frontal utilizando círculo vermelho como alvo

A posição final desejada representa a posição onde é possível observar o círculo de frente, ou seja, com o eixo óptico da câmera alinhado com o centro do círculo. Nesta pose, o círculo é avistado com semi-eixos iguais na imagem $(i_r = i_R = 195\,\text{pixels})$, e a distância entre o centro da imagem e do círculo é $i_a = 10\,\text{pixels}$.

Na Figura 37 encontra-se a imagem inicial do experimento, já na Figura 38 encontra-se a imagem que se deseja obter com o experimento (utilizada como imagem de referência no controle baseado em imagem). Sendo assim a posição ideal a ser encontrada pelo controle a fim de atingir a posição final desejada é $x_d = 10\,\text{cm}$, $y_d = 10\,\text{cm}$ e $\theta_d = 0°$.

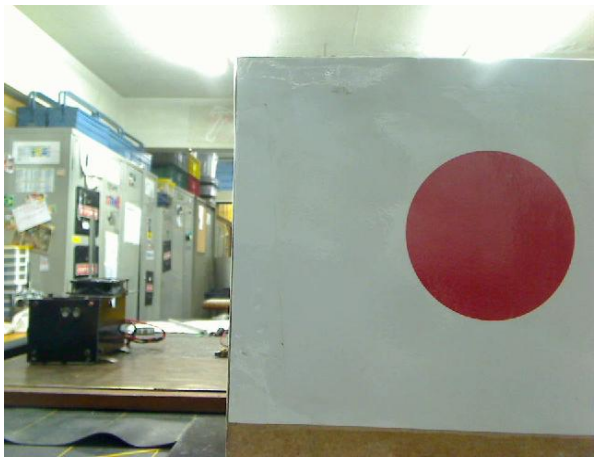


Figura 37 - Vista inicial do experimento 1 e 2

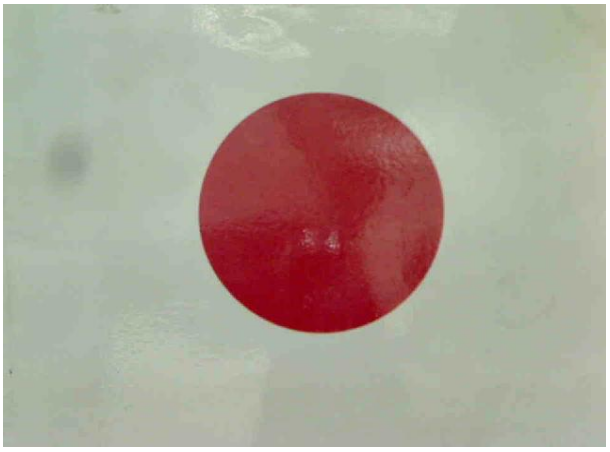


Figura 38 - Vista final do experimento 1

Foram testadas as quatro técnicas de controle: *look-and-move* baseado em pose e em imagem, e servo-visual também baseado em pose e em imagem. Os resultados encontram-se na Tabela 2, onde x_{rel}, y_{rel} e θ_{rel} são as coordenadas encontradas pelo software de controle a partir da imagem capturada na posição inicial, e x_f, y_f e θ_f são as coordenadas reais atingidas pela mesa, inferidas pelas leituras dos *encoders*. As últimas três linhas referem-se às características da imagem, que também poder ser utilizadas para avaliação do resultado; i_r, i_R e i_a correspondem respectivamente aos semi-eixos menor e maior da imagem do círculo e a distância entre o centro da imagem e do círculo. Para o controle servo-visual, as posições relativas x_{rel}, y_{rel} e θ_{rel} variam a cada interação do controle, quando novas imagens são capturadas, e por isso não foram apresentadas na Tabela 2.

Tabela 2 - Posições reais e relativas obtidas no experimento 1, para as quatro técnicas de controle, associada a valores desejadas $x_d = 10cm$, $y_d = 10cm$, $\theta_d = 0°$, $i_r = i_R = 195 pixels$ e $i_a = 10 pixels$

	Controle *look-and-move* baseado em pose	**Controle *look-and-move* baseado em imagem**	**Controle servo-visual baseado em pose**	**Controle servo-visual baseado em imagem**
x_{rel}	10,1 cm	7,9 cm	48 interações	20 interações
y_{rel}	8,0 cm	8,3 cm		
θ_{rel}	0°	3°		
x_f	10,7 cm	9,0 cm	10,0 cm	5,4 cm
y_f	8,1 cm	8,4 cm	10,3 cm	10,5 cm
θ_f	0°	3°	0°	10°
i_r	179 pixels	183 pixels	200 pixels	189 pixels
i_R	180 pixels	183 pixels	202 pixels	195 pixels
i_a	15 pixels	-37 pixels	-20 pixels	-37 pixels

Exceto o controle servo-visual baseado em imagem, todos os controles apresentaram resultados próximos do esperado. No controle *look-and-move*, os erros em (x, y, θ) foram provenientes do cálculo da posição $(x_{rel}, y_{rel}, \theta_{rel})$ a partir da imagem. Vale lembrar que, nos controles baseados em pose foi fixado o ângulo $\theta = 0°$. Porém, no caso do controle servo-visual baseado em imagem, a posição final é obtida utilizando as matrizes Jacobianas de imagem, e por isso não era possível fixar a rotação da câmera, o que gerou erros na orientação final de 3° e 10° para o *look-and-move* e servo-visual respectivamente, assim como erros em x e y. No entanto, ao observar a Figura 44 com as imagens resultantes, percebe-se que a imagem final ficou muito próxima da desejada, mesmo considerando os erros na posição final. Isto indica que a Jacobiana de imagem está próxima de uma singularidade na configuração de referência escolhida, ou seja, pequenos erros em i_r, i_R e i_a resultam em grandes erros em x, y e θ.

Durante a execução do programa, são apresentados ao usuário diversos gráficos para o acompanhamento da tarefa. Para todos os tipos de controle, são apresentados gráficos da posição *versus* tempo. No controle *look-and-move,* a posição real dos sensores vem acompanhada da posição desejada, que permanece constante durante todo o processo. Já no controle servo-visual, a posição desejada varia durante a execução da tarefa, e por isso, juntamente com a posição real é apresentado o erro entre a posição desejada e a real (chamado anteriormente também de posição relativa). Nas Figuras 39 e 40 são apresentados os resultados obtidos nos controles baseados em pose.

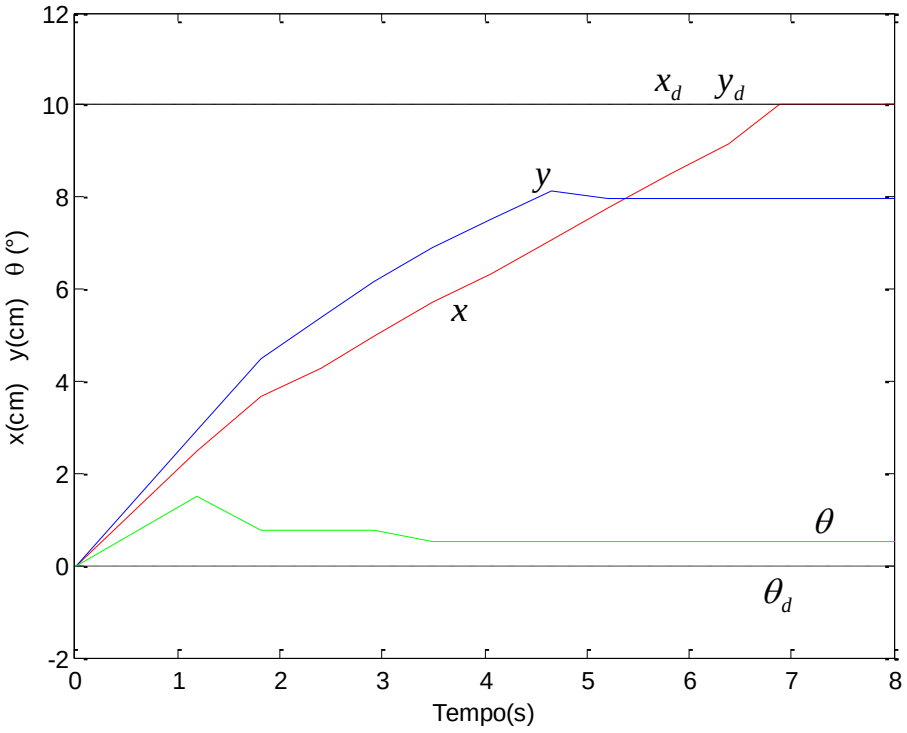


Figura 39 - Gráfico da posição real e desejada *versus* tempo (controle *look-and-move* baseado em pose), experimento 1

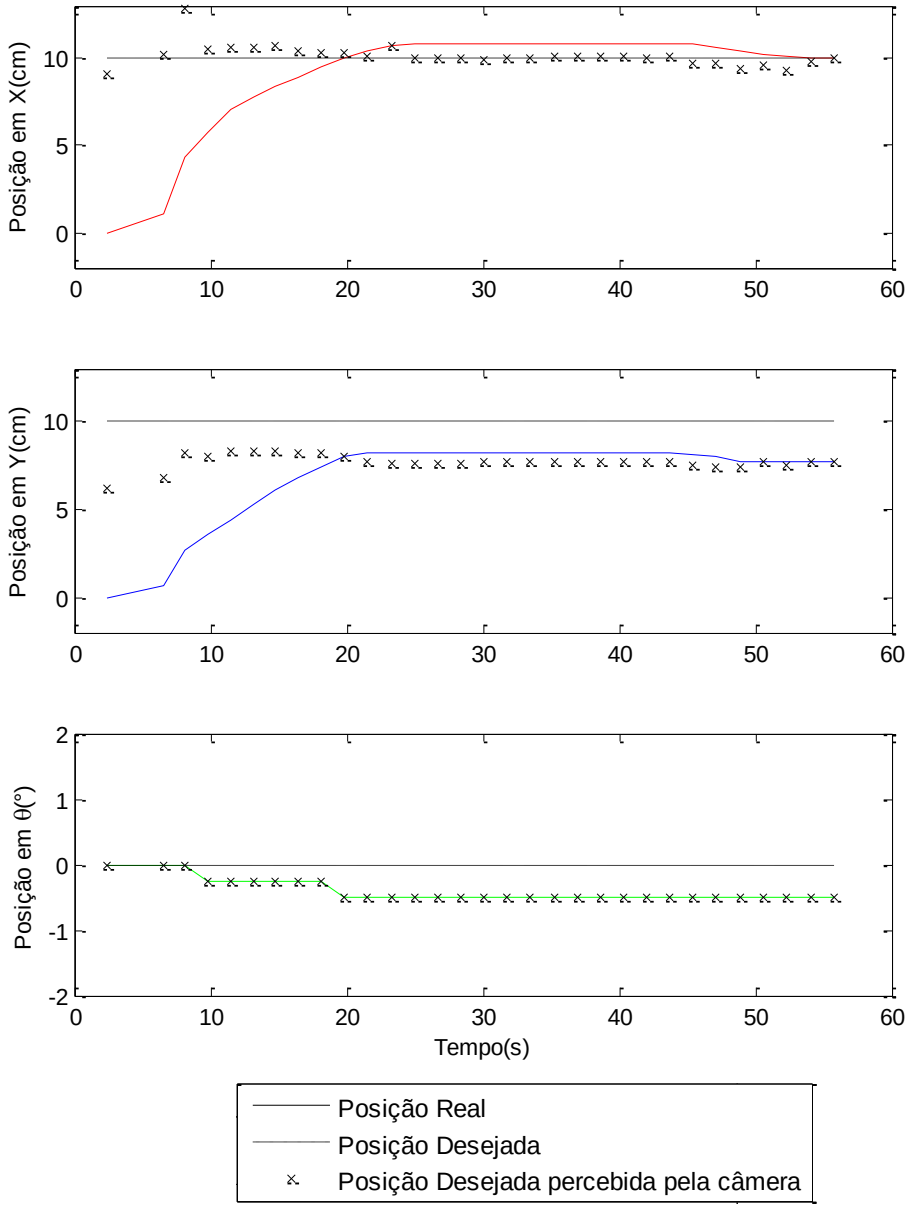


Figura 40 - Gráfico da posição real, desejada e percebida pela câmera através do software, *versus* tempo (controle servo-visual baseado em pose), experimento 1

Para o controle baseado em imagem, também são apresentados gráficos das características *versus* tempo, uma vez que a variável desejada não é mais a posição, e sim as características de imagem $\$_1$, $\$_2$ e $\$_3$. Nas Figuras 41 e 42 são apresentados os resultados obtidos nos controles baseados em imagem.

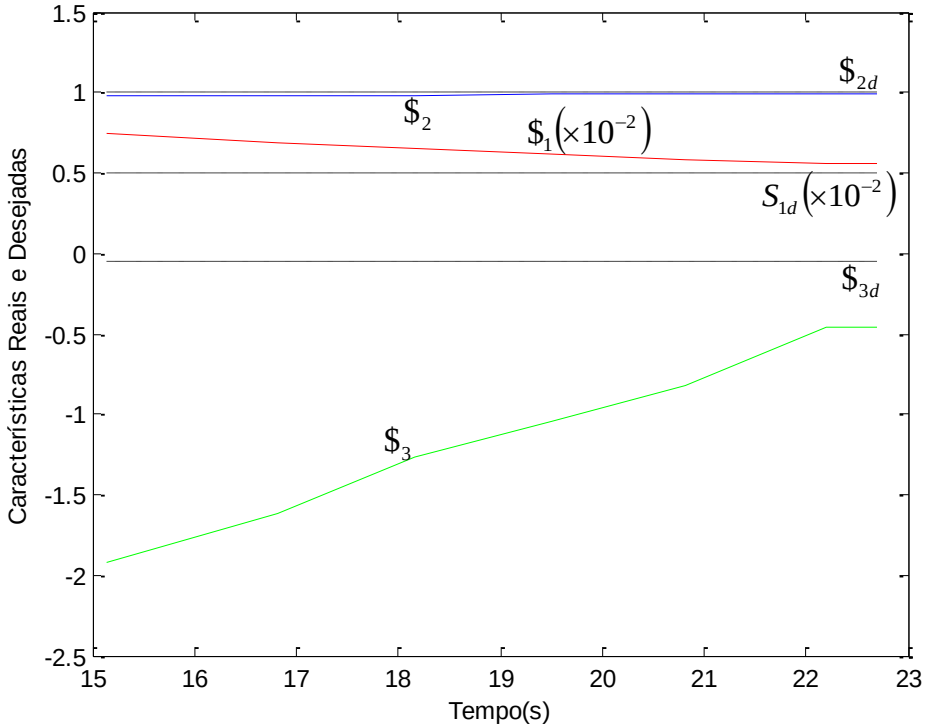


Figura 41 - Gráfico de características *versus* tempo (controle *look-and-move* baseado em imagem), experimento 1

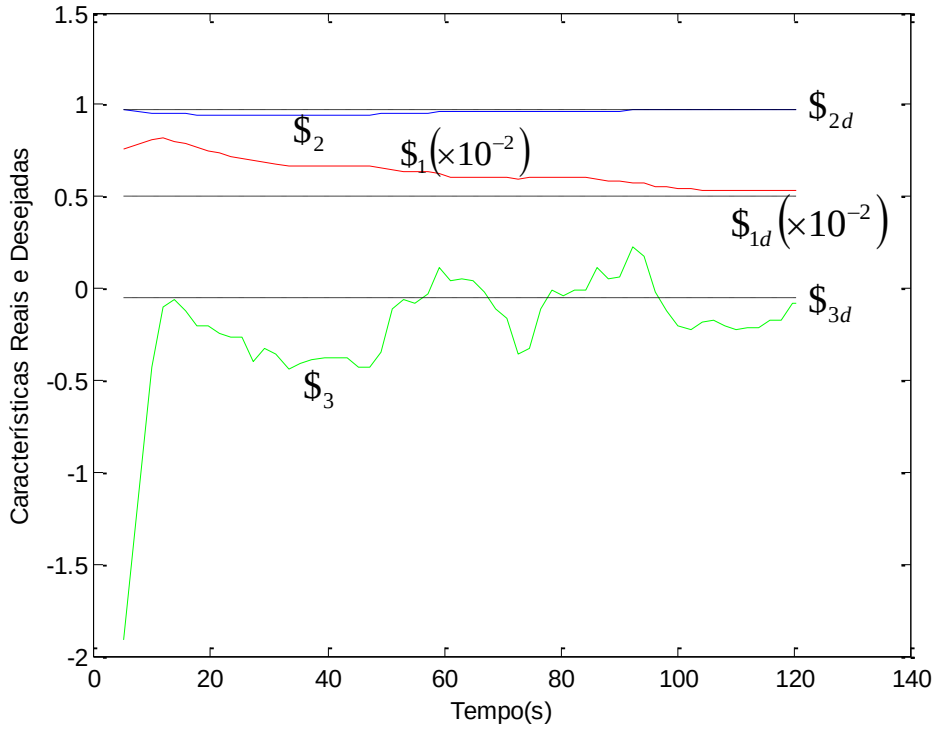


Figura 42 - Gráfico de características *versus* tempo (controle servo-visual baseado em imagem), experimento 1

Para estabelecer uma comparação entre as técnicas propostas, foi construído um gráfico com as posições reais, medidas a partir dos sensores de posição ao longo da movimentação da mesa, para todas as técnicas de controle.

Pela Figura 43, percebe-se que o controle *look-and-move* possui um tempo de acomodação, definido como tempo necessário para a curva atingir a posição desejada com um erro de $\pm 5\%$ (Ogata, 1997), menor que o do controle servo-visual. No entanto, apesar de lento o controle servo-visual atingi a posição desejada com um tempo de acomodação em torno de 25 segundos. O que torna o tempo final do controle igual a 90 segundos são as correções efetuadas através da imagem. A Figura 44 apresenta as imagens obtidas pelas quatro técnicas de controle.

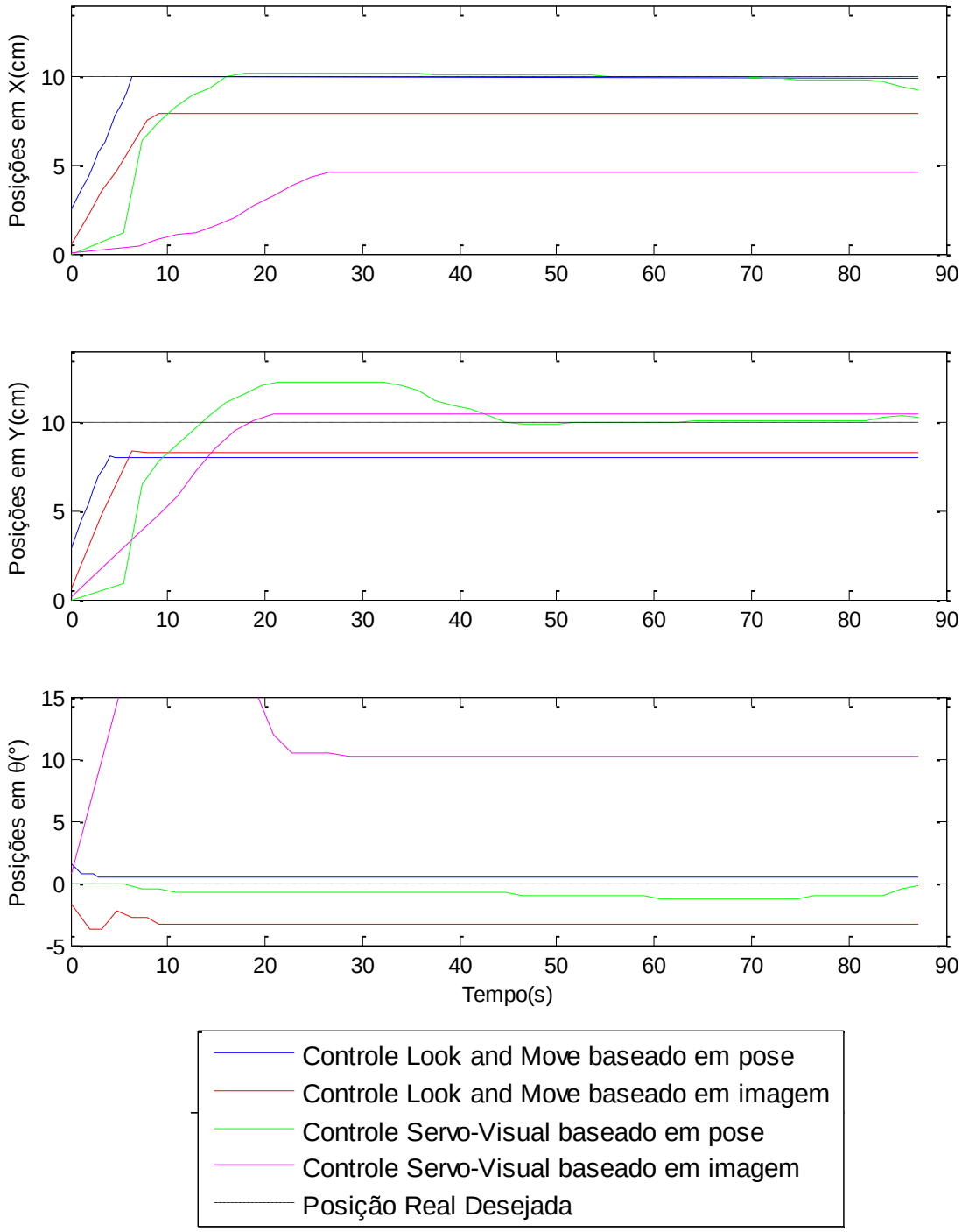
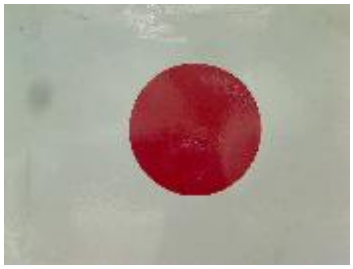


Figura 43 - Comparação entre as técnicas de controle pelo gráfico posição *versus* tempo, experimento 1

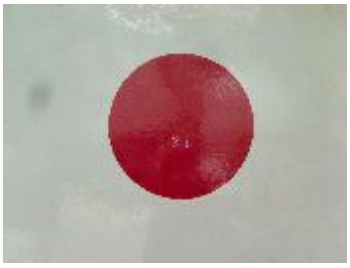
Imagem Inicial

Imagem Desejada

Controle Look and Move
baseado em pose



Controle Look and Move
baseado em imagem



Controle Servo Visual
baseado em pose

Controle Servo Visual
baseado em imagem

Figura 44 - Imagem inicial, deseja e resultantes nas quatro técnicas de controle, experimento 1

Pelas imagens geradas, percebe-se também que o resultado do controle servo-visual baseado em imagem, apesar do posicionamento diferenciado, realmente está de acordo com o esperado. Conforme visto na Tabela 2, ambos os resultados do controle *look-and-move* apresentaram um deslocamento em y menor do que o esperado. Sendo assim, o círculo avistado ao final do experimento também obteve raios menores que os da imagem de referência. Por fim, além de apresentar o melhor posicionamento, o controle servo-visual de fato apresentou as imagens mais semelhantes à desejada. Além de posicionar a câmera bem ao centro

da imagem, o deslocamento mais preciso em y fez com que o círculo fosse avistado com os raios iguais ao da imagem de referência.

5.2. Experimento 2: Teste lateral utilizando como alvo círculo

Para este experimento, a mesa foi posicionada conforme o esquema da Figura 45.

A posição final desejada representa a posição onde é possível observar o círculo descentralizado, porém ainda assim com semi-eixos iguais. Nesta pose, o círculo é avistado com $i_r = i_R = 285\,\text{pixels}$, e a distância entre o centro da imagem e do círculo (i_a) igual a -161 pixels. Essa combinação de i_r, i_R e i_a encontra-se longe das configurações de singularidade da matriz Jacobiana de imagem. Desse modo, espera-se obter um menor erro na pose final (x, y, θ). O experimento se inicia na mesma posição que no teste anterior, por isso a imagem inicial é a mesma da Figura 37. Já a imagem a ser obtida pelo controle encontra-se na Figura 46.

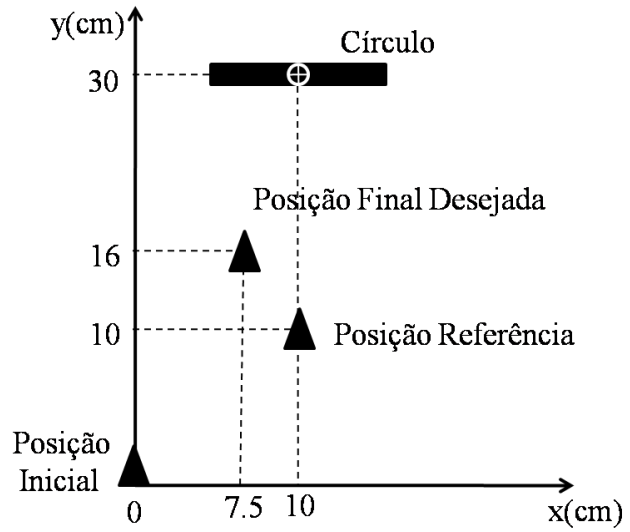


Figura 45 - Vista superior do experimento 2 para teste lateral utilizando círculo vermelho como alvo

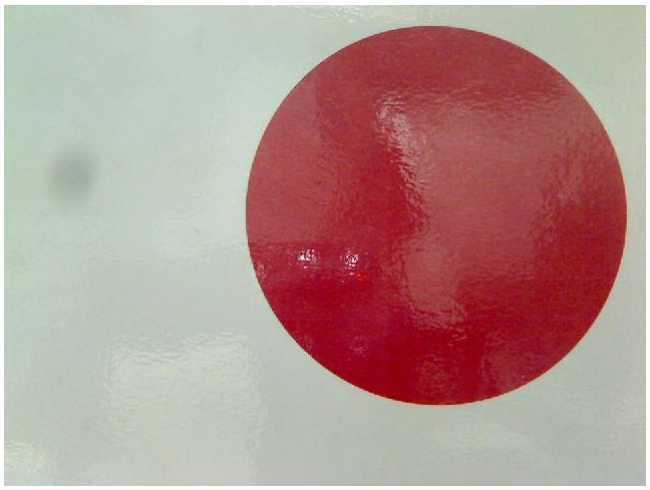


Figura 46 - Imagem desejada do experimento 2

A fim de atingir a posição final desejada, o controle deverá encontrar como coordenadas $x = 7.5\,\text{cm}$, $y = 16\,\text{cm}$ e $\theta = 0\,°$. Novamente foram testadas as quatro técnicas de controle e os resultados encontram-se na Tabela 3.

Tabela 3 - Posições reais e relativas obtidas no experimento 2, para as quatro técnicas de controle, associada a valores desejados $x_d = 7{,}5cm$, $y_d = 16cm$, $\theta_d = 0°$, $i_r = i_R = 285 pixels$ e $i_a = -161 pixels$

	Controle *look-and-move* baseado em pose	**Controle *look-and-move* baseado em imagem**	**Controle servo-visual baseado em pose**	**Controle servo-visual baseado em Imagem**
x_{rel}	7,5 cm	7,0 cm	36 interações	17 interações
y_{rel}	14,4 cm	14,9 cm		
θ_{rel}	0°	0°		
x_f	8,0 cm	6,9 cm	7,3 cm	7,2 cm
y_f	14,6 cm	15,0 cm	14,2 cm	14,3 cm
θ_f	0°	0°	0°	0°
i_r	257 pixels	264 pixels	281 pixels	278 pixels
i_R	260 pixels	268 pixels	284 pixels	282 pixels
i_a	-155 pixels	-187 pixels	-171 pixels	-166 pixels

Para estabelecer a comparação entre as técnicas propostas, será apresentado na Figura 47, o gráfico com as posições reais obtidas nos quatro controles.

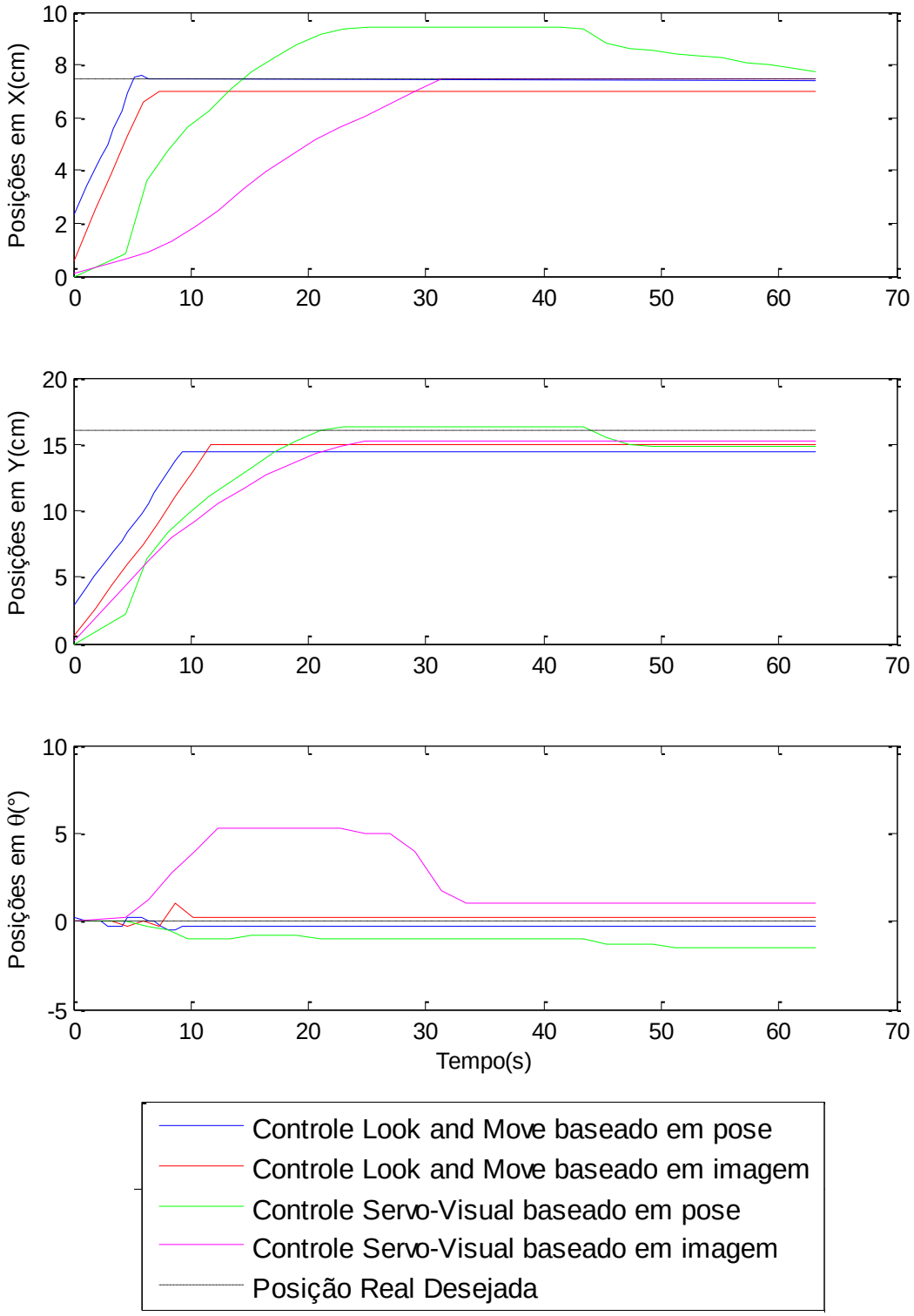


Figura 47 - Comparação entre as técnicas de controle pelo gráfico posição *versus* tempo, experimento 2

Assim como no primeiro experimento, é possível notar que devido à falta de realimentação por imagem do controle *look-and-move*, os deslocamentos em x e y não foram precisos. Isto pode ser observado não só pelos valores apresentados na Tabela 3, como também pelas imagens geradas pelos controles, onde o erro no centro do círculo e o tamanho dos raios gerados demonstram o mau posicionamento (Figura 48). Já o controle servo-visual, apesar de mais lento, conseguiu posicionar a câmera de forma a apresentar um resultado muito próximo do esperado.

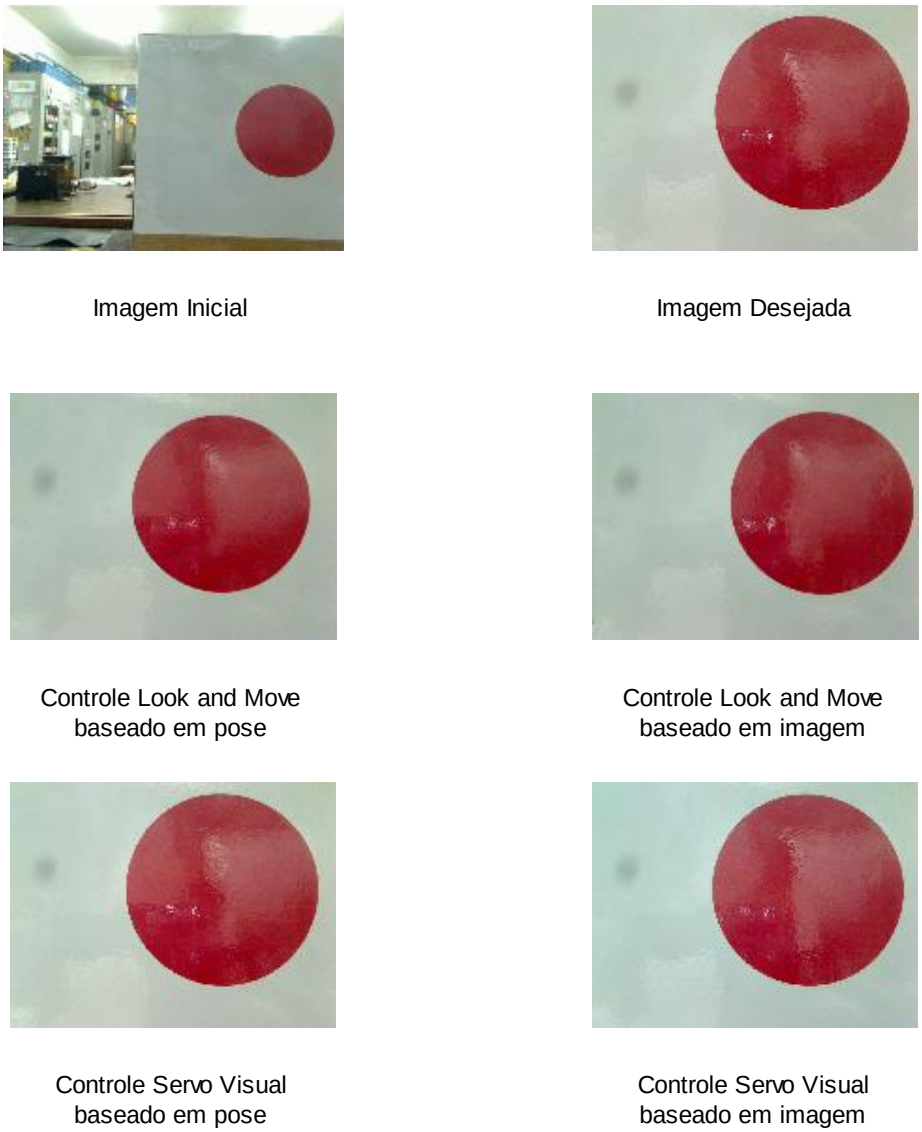


Figura 48 - Imagem inicial, desejada, e resultantes nas quatro técnicas de controle, experimento 2

5.3.
Experimento 3: Teste frontal utilizando como alvo painel

Para os testes com o painel frontal, a câmera foi posicionada inicialmente conforme a Figura 49. Para estes testes, o posicionamento foi obtido através do algoritmo SIFT, e por isso além das posições em x e y também pode ser determinada a rotação da câmera sem a necessidade de usar aproximação no modelo.

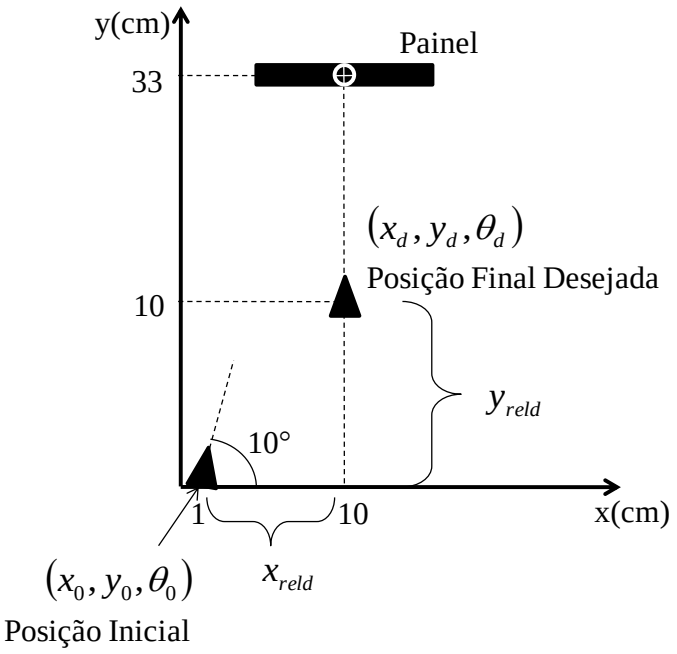


Figura 49 - Vista superior do experimento 3 para teste frontal utilizando como alvo o painel 2D

Segundo o Esquema da Figura 49, é possível verificar as seguintes relações

$$\begin{aligned} x_d &= x_0 + x_{reld} \\ y_d &= y_0 + y_{reld} \\ \theta_d &= \theta_0 + \theta_{reld} \end{aligned} \tag{89}$$

Para atingir a posição desejada, o software de controle deveria então encontrar como coordenadas os parâmetros $x_{reld} = 9\,\text{cm}$, $y_{reld} = 10\,\text{cm}$ e $\theta_{reld} = 0°$, e fisicamente deseja-se que a mesa coordenada atinja $x_d = 10\,\text{cm}$, $y_d = 10\,\text{cm}$ e $\theta_d = 0°$. Pode-se afirmar também que

$$\begin{aligned} x_f &\neq x_0 + x_{rel} \\ y_f &\neq y_0 + y_{rel} \\ \theta_f &\neq \theta_0 + \theta_{rel} \end{aligned} \tag{90}$$

onde (x_f, y_f, θ_f) são as coordenadas reais atingidas pela mesa e $(x_{rel}, y_{rel}, \theta_{rel})$ representam os valores encontrados pelo software de controle. A equação (90) pode ser confirmada, uma vez que as coordenadas relativas desejadas são enviadas à mesa porém dependem de um controle exato para serem atingidas.

Nestas configurações desejadas o painel é visto conforme a Figura 50, e partindo-se da posição inicial apresentada no Esquema da Figura 49, é possível observar a Figura 51.

Figura 50 - Imagem frontal do painel para o experimento 3

Figura 51 - Imagem inicial do experimento 3 utilizando como alvo o painel 2D

Nos experimentos utilizando SIFT, foram utilizados apenas os controles baseados em pose. Os controles baseados em imagem dependeriam de uma matriz Jacobiana de imagem que fosse obtida a partir de parâmetros geométricos do painel $\$_1$, $\$_2$, etc., como por exemplo, sua largura e altura na imagem, e a posição de seu centro geométrico. Porém, não foram implementados neste trabalho algoritmos que possibilitassem obter estes parâmetros para a imagem de um painel genérico. Estes parâmetros poderiam ser obtidos, por exemplo, através de um algoritmo de detecção de arestas. No entanto, implementou-se duas formas de controle baseado em pose, tanto para o *look-and-move* quanto para o servo-visual. As duas formas consistem em informar a pose desejada (Caso A) através de coordenadas (x_d, y_d, θ_d), ou (Caso B) através de uma imagem previamente obtida na posição desejada.

Utilizou-se como imagem de referência, uma imagem frontal do painel. Para os casos onde a imagem desejada era diferente da imagem de referência, eram feitas duas correlações: a primeira entre a imagem desejada e a de referência, e a segunda entre a imagem atual e a de referência (conforme indicado na Figura 52). Sendo assim era possível obter a relação entre a imagem atual e a desejada.

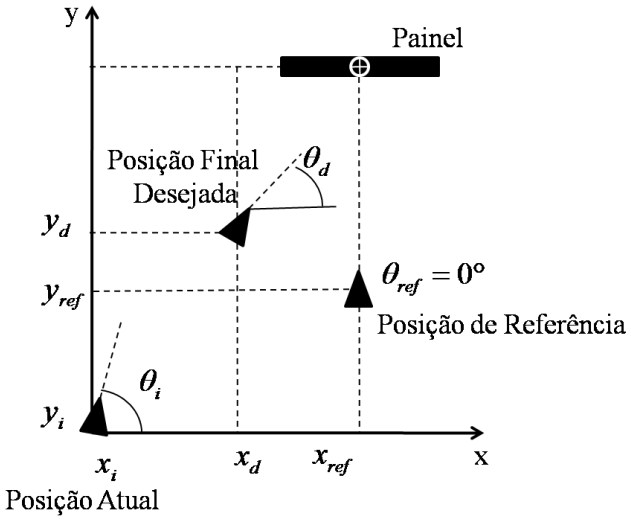


Figura 52 - Esquema entre a posição atual, desejada e de referência

Abaixo a Tabela 4, apresenta os resultados obtidos nos quatro controles. As 3 primeiras linhas indicam as distâncias relativas encontradas pelo software, e as 3 últimas linhas as posições reais atingidas pela mesa x_f, y_f, θ_f.

Tabela 4 - Posições reais e relativas obtidas no experimento 3 para quarto técnicas de controle, associada a valores desejados $x_d = 10cm$**,** $y_d = 10cm$**,** $\theta_d = 0°$**,** $x_{reld} = 9cm$**,** $y_{reld} = 10cm$ **e** $\theta_{reld} = 0°$

	Controle *look-and-move* baseado em pose Caso (A)	**Controle *look-and-move* baseado em pose Caso (B)**	**Controle servo-visual baseado em pose Caso (A)**	**Controle servo-visual baseado em pose Caso (B)**
x_{rel}	7,3 cm	7,5 cm	55 interações	27 interações
y_{rel}	9 cm	9,1 cm		
θ_{rel}	7,6°	7,7°		
x_f	8,5 cm	8,4 cm	9,5 cm	9,5 cm
y_f	9,3 cm	9,3 cm	9,8 cm	9,5 cm
θ_f	1°	1°	1°	1°

Mais uma vez, foram traçados todos os valores de posição em um único gráfico, apresentado na Figura 53.

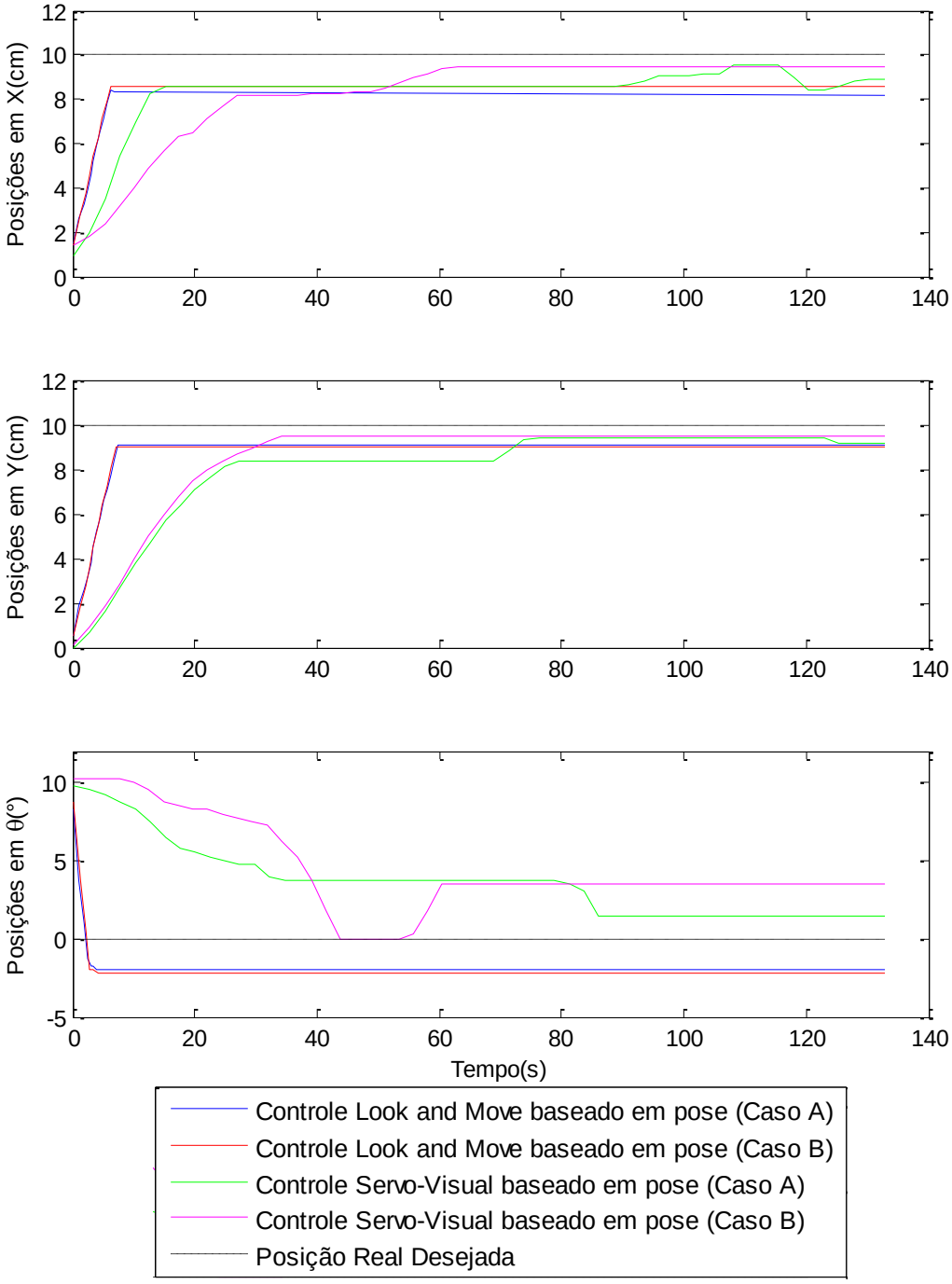


Figura 53 - Comparação entre as técnicas de controle pelo gráfico posição *versus* tempo, experimento 3

Para todos os casos de controle *look-and-move*, o tempo final de processamento foi relativamente o mesmo tanto para o círculo quanto para o painel, já que apenas uma imagem era analisada (e uma vez determinada a posição, o controle não precisa capturar ou processar novas imagens). No entanto, para o controle servo-visual é interessante reparar que o tempo de acomodação

aumentou consideravelmente, uma vez que a cada interação do programa um novo processamento do algoritmo SIFT é realizado. Para o caso do círculo o controle servo-visual apresentou um tempo de acomodação em torno de 25 segundos, e um tempo final de 90 segundos. Para este caso com o painel, o tempo de acomodação aumentou para 90 segundos, e o tempo final para 120 segundos.

Pelas imagens obtidas ao final do controle (Figura 54), percebe-se a influência da realimentação visual na precisão do experimento. As imagens obtidas pelo controle *look-and-move* não conseguiram atingir a imagem esperada com tanta precisão quanto o servo visual.

Imagem Inicial

Imagem Desejada

Controle Look and Move
baseado em pose - Caso (A)

Controle Look and Move
baseado em pose - Caso (B)

Controle Servo Visual
baseado em pose - Caso (A)

Controle Servo Visual
baseado em pose - Caso (B)

Figura 54 - Imagem inicial, desejada e resultantes nas quatro técnicas de controle, experimento 3

5.4. Experimento 4: Teste lateral utilizando como alvo painel

Para os testes com o painel lateral, o experimento foi montado conforme a Figura 55.

Para atingir a posição desejada, o software de controle deveria então encontrar como parâmetros $x_d = 8\,\text{cm}$, $y_d = 15\,\text{cm}$ e $\theta_d = 30°$; e $x_{reld} = 8\,\text{cm}$, $y_{reld} = 15\,\text{cm}$ e $\theta_{reld} = 20°$. Nesta configuração, o painel é visto conforme a Figura 56, e esta é a imagem informada ao *software* nos controles do caso (B). Já a imagem inicial do experimento pode ser observada na Figura 57.

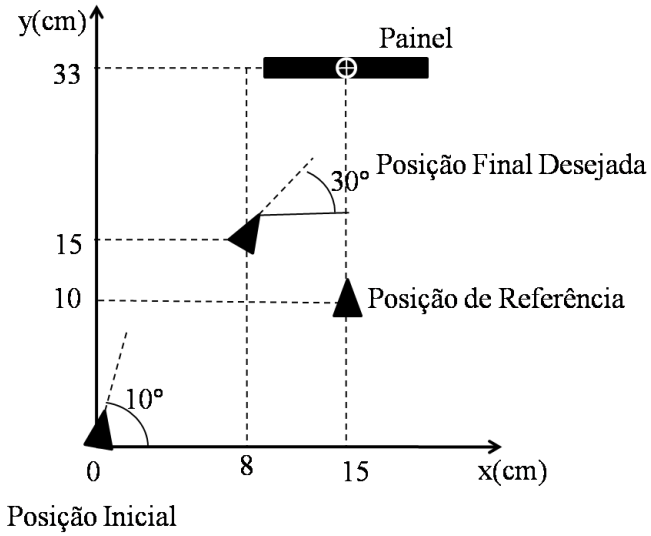


Figura 55 - Vista superior do experimento 4 para teste lateral utilizando painel 2D como alvo

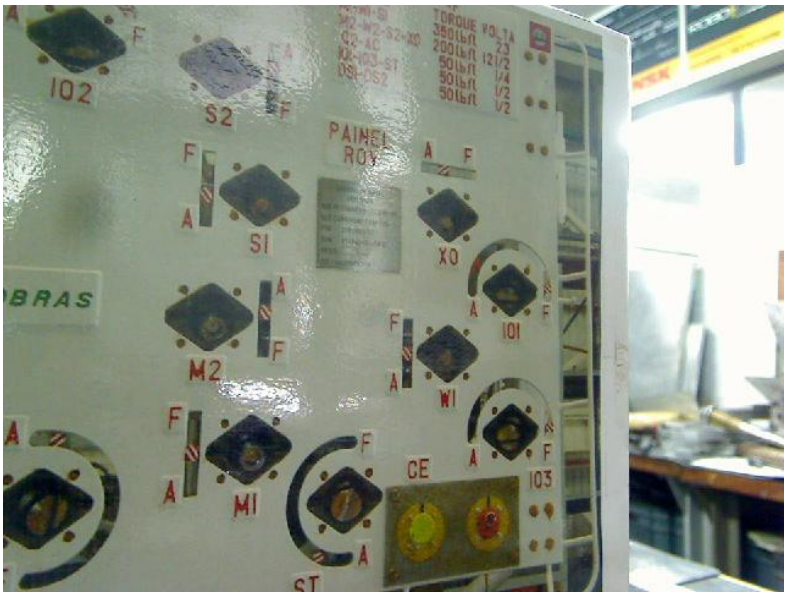


Figura 56 - Imagem lateral do painel

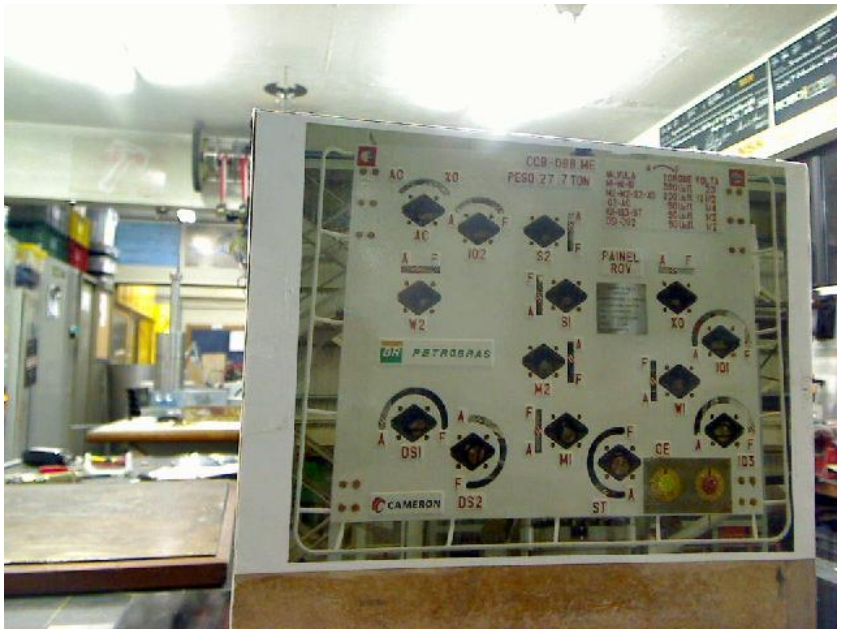


Figura 57 - Imagem inicial do experimento 4 utilizando como alvo o painel 2D

Novamente foram testadas as quatro técnicas de controle, e os resultados encontram-se na Tabela 5.

Tabela 5 - Posições reais e relativas obtidas no experimento 4 para quatro técnicas de controle, associada a valores desejados $x_d = 8cm$, $y_d = 15cm$, $\theta_d = 30°$ **e** $x_{reld} = 8$ **cm,** $y_{reld} = 15$ **cm e** $\theta_{reld} = 20°$

	Controle *look-and-move* baseado em pose Caso (A)	**Controle *look-and-move* baseado em pose Caso (B)**	**Controle servo-visual baseado em pose Caso (A)**	**Controle servo-visual baseado em pose Caso (B)**
x_{rel}	6,6 cm	6,6 cm	13 interações	50 interações
y_{rel}	15,9 cm	19 cm		
θ_{rel}	16,5°	28°		
x_f	6,8 cm	6,8 cm	6,5 cm	7,0 cm
y_f	16,3 cm	19,4 cm	17 cm	16,6 cm
θ_f	27°	38°	34°	33°

Pelos resultados apresentados na Tabela 5, percebe-se que a posição desejada foi atingida com maiores erros para o controle *look-and-move*. Ao analisar as imagens finais geradas por estes, pode-se perceber que todos conseguiram convergir para imagens bem próximas da desejada.

Durante a execução do controle servo-visual, percebeu-se no entanto que quanto mais próximo do objeto alvo, maior era o erro nas posições obtidas pelo *software*. Isso pode ser explicado pela baixa resolução das imagens. Além disso, muitas vezes a rotação errada da câmera implicava na perda de grande parte do objeto alvo no seu campo de visão. Além disso, grande parte do objeto alvo se encontrava na região de distorção da câmera, cuja calibragem foi ignorada neste projeto. Com estes problemas, o algoritmo SIFT detectava um número muito baixo de correlações, que em alguns casos podiam até não ser verdadeiras. A fim de minimizar os resultados de posicionamento falso, foram feitas verificações ao longo do processo de forma que um posicionamento muito distante do posicionamento obtido na interação anterior fosse descartado. A princípio, optou-se por manter o resultado anterior, assumido como correto, e continuar a movimentação da mesa. Porém, se fossem obtidas sucessivas posições erradas, a mesa poderia chegar ao fim do curso e danificar o experimento. Por isso optou-se por automaticamente parar a mesa nestes casos até que um novo resultado correto fosse obtido. Foram considerados posicionamentos falsos, aqueles que obtivessem uma diferença na angulação θ de 15° a mais que na interação anterior.

No caso do controle servo-visual caso (B), que obteve um total de 50 interações, cerca de 23 interações foram descartadas por apresentarem resultados considerados falsos. Ao analisar o gráfico obtido na Figura 59, é possível observar os erros de posicionamento obtidos. Na curva que indica a posição desejada obtida pelo *software* observa-se que diversos pontos encontram-se muito afastados da interação anterior.

Para o caso do controle servo-visual caso (A), também foi implementado o mesmo mecanismo de descartar interações falsas. Ao executar o mesmo teste algumas vezes, percebeu-se que este também apresentava o mesmo problema. No entanto, o experimento apresentado na Tabela 5, conseguiu convergir rapidamente sem contar com nenhum posicionamento falso (vide Figura 58). É importante ressaltar que apesar de a programação, do posicionamento do objeto alvo, e da posição inicial da câmera ser o mesmo, as trajetórias obtidas ao longo de diversos

experimentos dificilmente serão iguais. Isso porque os resultados dependem das imagens obtidas, e durante a movimentação da mesa as imagens não são obtidas exatamente no mesmo instante. Ao final do experimento, no entanto, espera-se que todos os experimentos atinjam uma posição final similar.

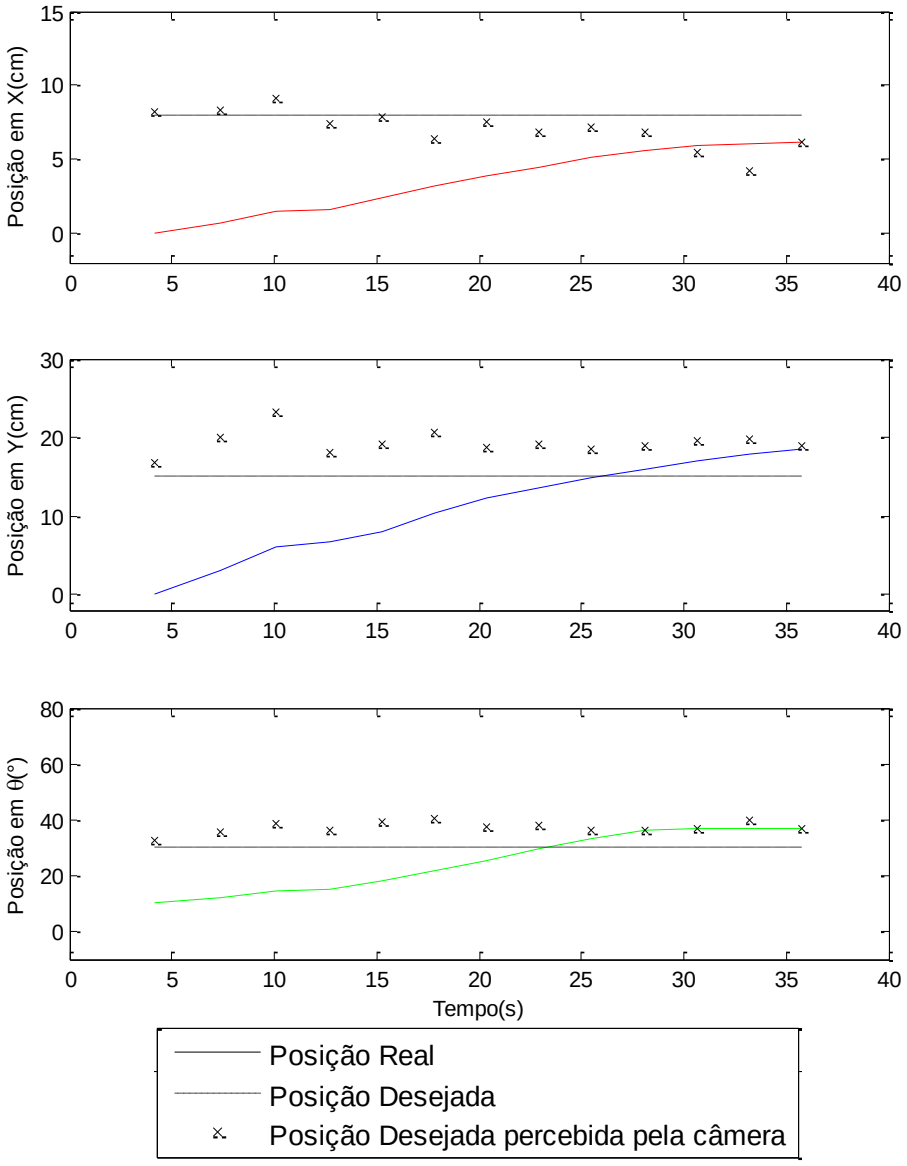


Figura 58 - Gráfico da posição real, desejada e percebida pela câmera através do software *versus* tempo para o controle servo-visual caso (A), experimento 4

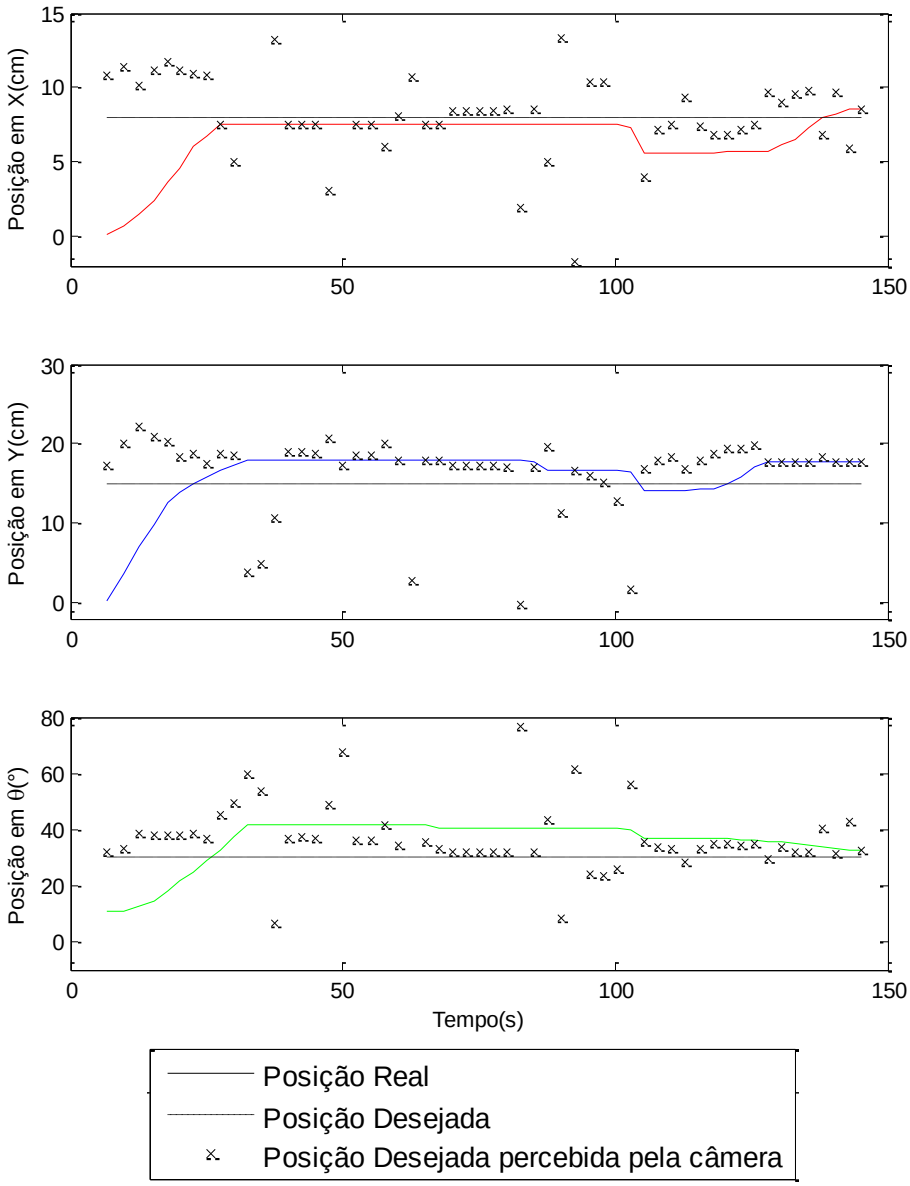


Figura 59 - Gráfico da posição real, desejada e percebida pela câmera através do software versus tempo para o controle servo-visual caso (B), experimento 4

Na Figura 60, é apresentado um gráfico com o resultado dos quatro controles. Percebe-se que nenhum deles atingiu a posição desejada, porém pela Figura 61, observa-se que todas as imagens atingidas assemelham-se da desejada.

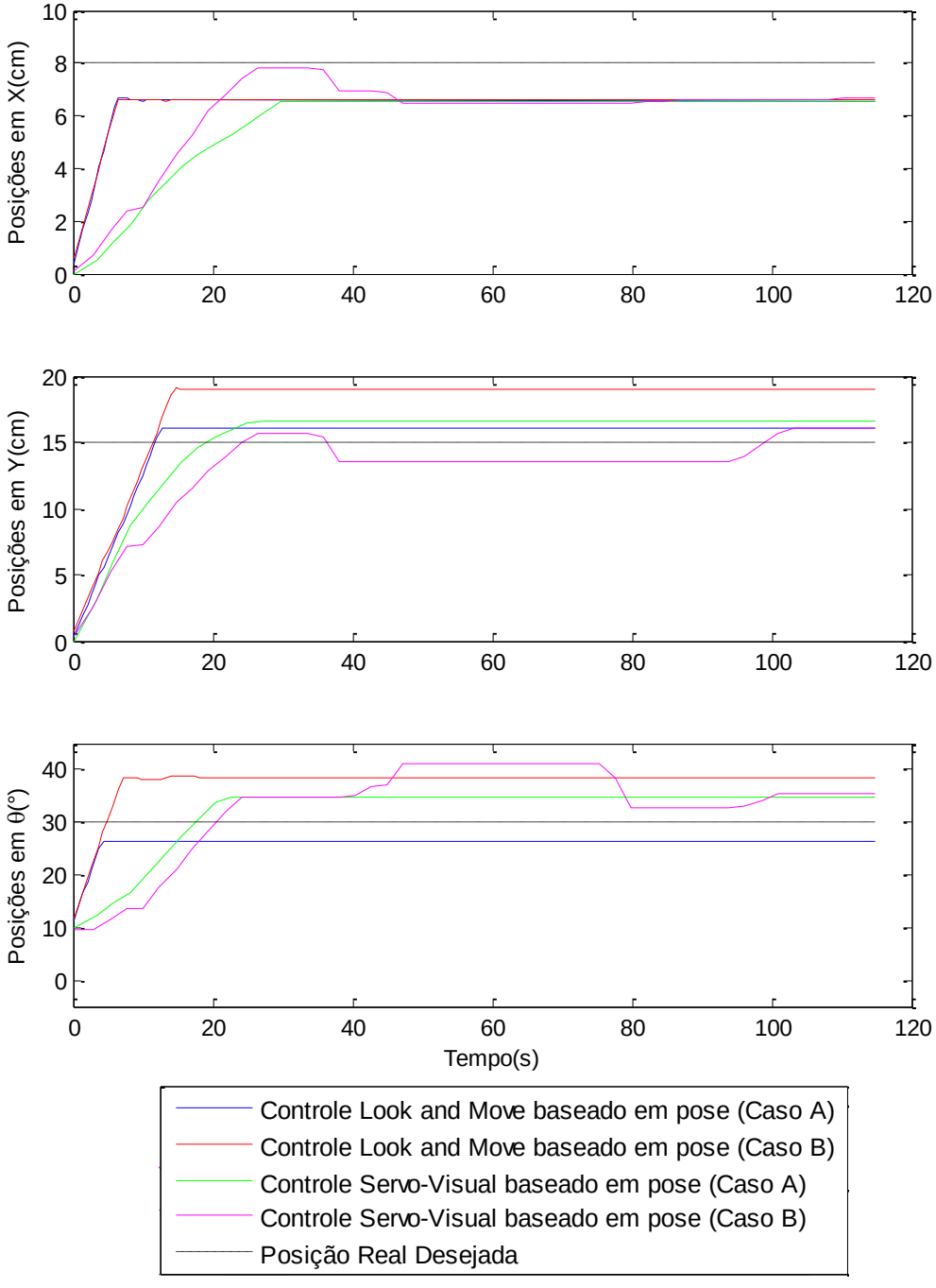


Figura 60 - Comparação entre as técnicas de controle pelo gráfico posição *versus* tempo, experimento 4

Imagem Inicial

Imagem Desejada

Controle Look and Move
baseado em pose - Caso (A)

Controle Look and Move
baseado em pose - Caso (B)

Controle Servo Visual
baseado em pose - Caso (A)

Controle Servo Visual
baseado em pose - Caso (B)

Figura 61 - Imagem inicial, desejada e resultantes nas quatro técnicas de controle, experimento 4

6
Conclusões

Nesta dissertação foram estudados os controles servo-visual e *look-and-move*, buscando determinar suas vantagens e desvantagens. Estes controles foram validados através de um sistema experimental, construído especialmente para este trabalho. Uma mesa coordenada XYθ já existente foi automatizada para posteriormente ser controlada por computador; um módulo eletrônico também foi construído de forma a funcionar com todos os controles estudados; e um *software* de controle foi desenvolvido de forma a utilizar diferentes objetos alvos.

Após os experimentos, concluiu-se que a realimentação utilizando imagens presente no controle servo-visual, garante um posicionamento preciso do experimento, diferente do controle *look-and-move*, que utiliza a imagem desejada apenas uma vez. Apesar de mais preciso, o controle servo-visual leva mais tempo para atingir o seu objetivo. Além disso, sua frequência depende inteiramente da velocidade do processamento da imagem. Apesar de muito utilizado na literatura de visão computacional, o algoritmo SIFT ainda é muito lento quando utilizado em tempo real, e por isso atrasa o posicionamento do controle servo-visual. Percebeu-se nos testes utilizando o círculo vermelho para o controle servo-visual, que é possível obter um resultado de qualidade e em tempo curto quando o processamento da imagem é feito de forma rápida. Já o controle *look-and-move*, apesar de impreciso, pode ser utilizado em casos onde a velocidade é mais importante que a precisão. Uma alternativa para casos onde o objeto alvo encontra-se muito afastado da câmera é utilizar o controle *look-and-move* a fim de aproximar rapidamente o objeto da câmera, e por fim utilizar o controle servo-visual para refinar o posicionamento.

Um dos problemas encontrados ao longo do experimento diz respeito às falsas correlações encontradas pelo algoritmo SIFT. No entanto, não era objetivo deste trabalho se aprofundar na implementação do algoritmo. Sabe-se que na literatura de visão computacional existem diversas maneiras de evitar falsas

correlações, e que podem ser incorporadas a este trabalho para futuramente aperfeiçoar os resultados.

Para os controles baseado em pose e imagem, os resultados apresentados mostraram que as técnicas são bastante parecidas, diferenciando-se apenas na variável de estado desejada. Esta escolha, no entanto, influenciou no resultado em configurações próximas de singularidade na matriz jacobiana de imagem $J_{\$p}$. Sabendo que nestas configurações grandes variações na posição do objeto implicavam em pequenas variações na imagem, o controle baseado em imagem apresentou resultados diferentes do esperado. Experimentalmente, no entanto, percebe-se que o controle baseado em imagem é mais intuitivo para o usuário, uma vez que, para alcançar uma posição basta enviar ao sistema uma imagem referente a posição a ser alcançada, ao invés de ter que computar e enviar coordenadas específicas.

Além de aumentar a velocidade do algoritmo SIFT e remover suas falsas correlações, sugere-se para trabalhos futuros testes com objetos 3D genéricos (não necessariamente com faces planas, como no painel estudado), utilizando técnicas mais abrangentes de detecção do posicionamento de objetos no espaço. Além disso, seria de bastante utilidade testes do controle servo-visual com objetos em movimento, naturalmente possível, uma vez que a aquisição de imagens é feita continuamente durante a movimentação.

Outro trabalho proposto consiste em utilizar manipuladores industriais robóticos que possuem mais graus de liberdade que uma mesa coordenada, para seguir os objetos. Além disso, utilizando garras em sua extremidade, estes poderiam não só alcançar o objeto alvo como também manipulá-lo.

7
Referências Bibliográficas

Allota, B., e Colombo, C. (1999). On the use of linear camera-object interaction models in visualservoing. *IEEE Transaction on Robotics and Automation* , pp. 350-357.

Asada, H., e Slotine, J.-J. E. (1986). *Robot Analysis and Control.* Wiley-Interscience.

Astrom, K. J., e Hagglund, T. (1995). *PID Controllers: Theory, Design and Tuning.* ISA.

Augustson, T. M. (2007). *Vision based real time calibration of robots with application in subsea intervations.* Dissertação de Mestrado, Pontifícia Universidade Católica do Rio de janeiro.

Forsyth, D. A., e Ponce, J. (2003). *Computer Vision - A Modern Approach*. Prentice-Hall.

Harris, C., e Stephens, M. (1988). A combined corner and edge detector. *In Fourth Alvey Vision Conference* (pp. 147-151). Manchester, UK: The Plessey Company.

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

Houshangi, N. (1990). Control of a robotic manipulator to grasp a moving target using vision. *IEEE international Conference on Robottics and Automation*, vol.1, pp. 604-609.

Hutchinson, S., Hager, G., e Corke, P. (October de 1996). A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation* , pp. 651-670.

Inoue, H., e Shirai, Y. (1971). *Guiding a robot by visual feedback assembling tasks.* Eletrotechnical Laboratory Chiyoda-ku, Toquio, Japão.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *international Journal of Computer Vision* , pp. 91-110.

Ogata, K. (1997). *Modern Control Engineering.* Prentice-Hall.

Sanderson, A., e Weiss, L. (1980). Image-based visual servo control using relational graph error signals. *Proc. IEEE* , pp. 1074-1077.

Smith, E., e Papanikolopoulos, N. (1996). Vision-Guided Robotic Grasping: Issues and Experiments. *IEEE Intetrnational Conference on Robotics and Automation* , pp. 3203-3208, vol.4.

Tomasi, C., e Kanade, T. (1991). Detection and Tracking of Point Features. *Technical Report CMU-CS-91-132*

Trucco, E., e Verri, A. (1998). *Introductory Techniques for 3-D Computer Vision.* Prentice Hall.

Ziegler, J. G., e Nichols, N. B. (1942). Optimum settings for automatic controllers. *ASME Transactions* , pp. 759-768.

Apêndice I

Especificações dos motores *Banebots* utilizados na automatização da mesa coordenada XYθ, apresentado na Figura 20.

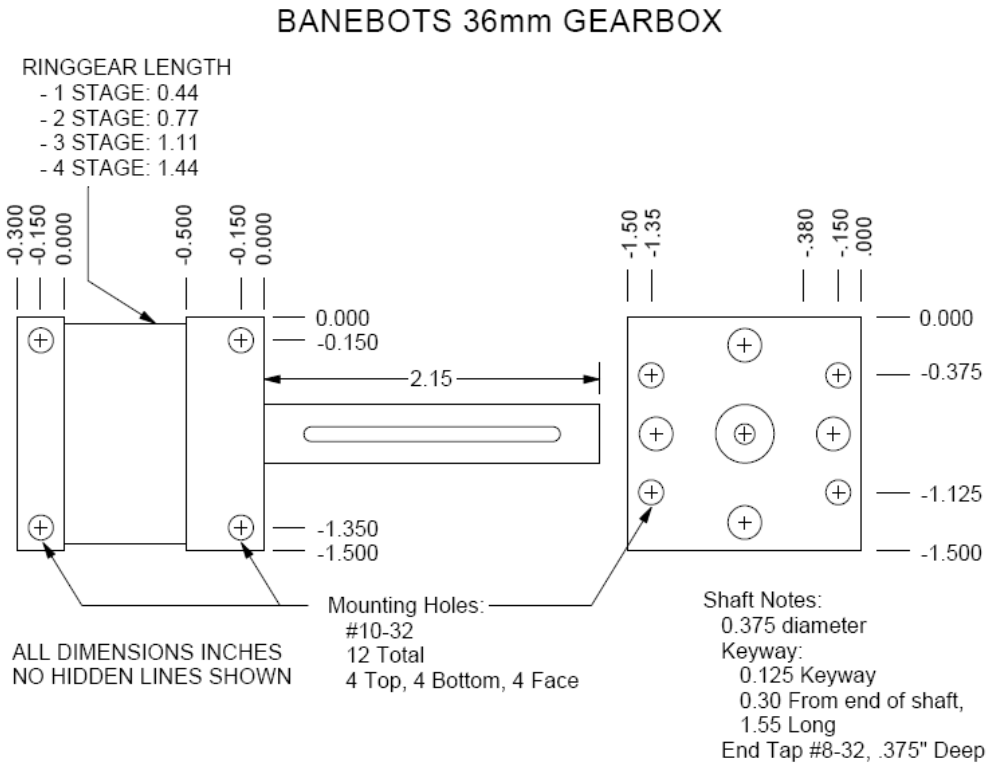
Physical

Type	: Planetary
Reduction	: 16:1
Stages	: 2
Gear Material	: All Metal
Weight (Gearbox only)	: 6.2 oz (174g)
Weight (with motor)	: 11.6 oz (327g)
Length (Gearbox only)	: 1.6 in (40mm)
Length (with motor)	: 3.7 in (93mm)
Width (Sqaure)	: 1.5 in (38mm)
Shaft Diameter	: 0.375 in (10mm)
Shaft Length	: 2.15 in (55mm)
Shaft Key	: 0.125 in (3.2mm)
Shaft End Tap	: #8-32
Mounting Holes (12)	: #10-32

Calculated Performance*

Motor	: RS-540 (Pinion)
Operating v	: 4.5v – 12v
Nominal v	: 12v
No Load RPM	: 1050
No Load A	: 1ª
Stall Current	: 42ª
Stall Torque	: 632 oz-in 4461 mN-m
Kt	: 15 oz-in/A 106 mN-m/A
kV	: 88 rpm/v
RPM - Peak Eff	: 908
Torque - Peak Eff	: 99.3 oz-in 701 mN-m
Current - Peak Eff	: 6.6ª

Esquema do Motor:



Apêndice II

Datasheet do microcontrolador PIC 16F767, utilizado no sistema eletrônico da mesa coordenada,e apresentado na Figura 25.

Low-Power Features:

- Power-Managed modes:
  - Primary Run (XT, RC oscillator, 76 µA, 1 MHz, 2V)
  - RC_RUN (7 µA, 31.25 kHz, 2V)
  - SEC_RUN (9 µA, 32 kHz, 2V)
  - Sleep (0.1 µA, 2V)
- Timer1 Oscillator (1.8 µA, 32 kHz, 2V)
- Watchdog Timer (0.7 µA, 2V)
- Two-Speed Oscillator Start-up

Oscillators:

- Three Crystal modes:
  - LP, XT, HS (up to 20 MHz)
- Two External RC modes
- One External Clock mode:
  - ECIO (up to 20 MHz)
- Internal Oscillator Block:
  - 8 user-selectable frequencies (31 kHz, 125 kHz, 250 kHz, 500 kHz, 1 MHz, 2 MHz, 4 MHz, 8 MHz)

Analog Features:

- 10-bit, up to 14-channel Analog-to-Digital Converter:
  - Programmable Acquisition Time
  - Conversion available during Sleep mode
- Dual Analog Comparators
- Programmable Low-Current Brown-out Reset (BOR) Circuitry and Programmable Low-Voltage Detect (LVD)

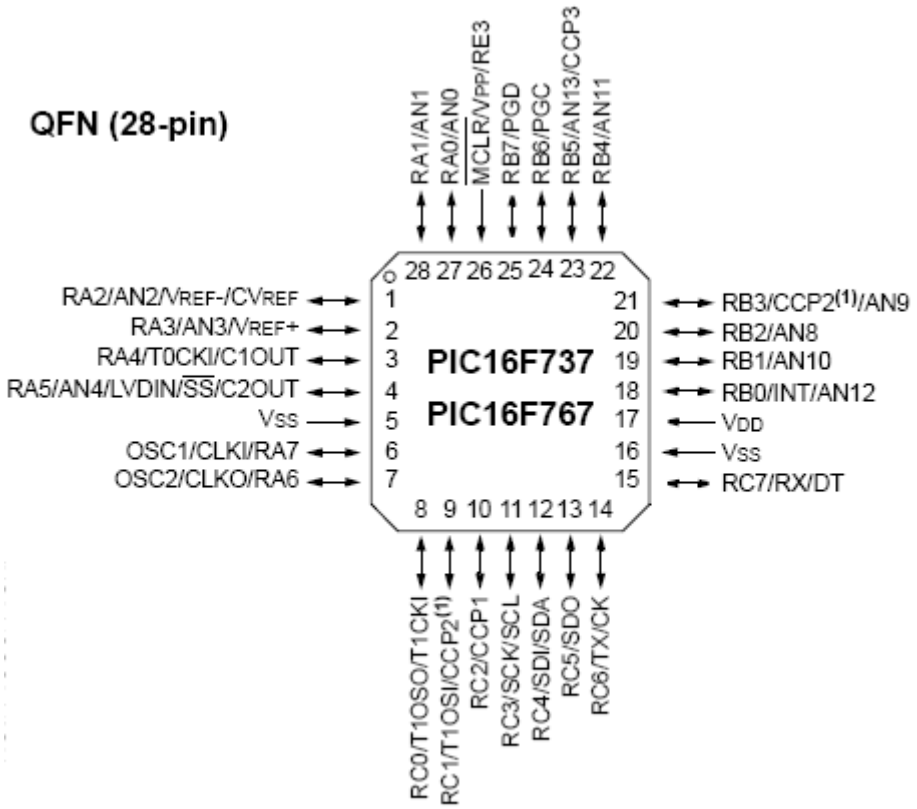
Peripheral Features:

- High Sink/Source Current: 25 mA
- Two 8-bit Timers with Prescaler
- Timer1/RTC module:
  - 16-bit timer/counter with prescaler
  - Can be incremented during Sleep via external 32 kHz watch crystal
- Master Synchronous Serial Port (MSSP) with 3-wire SPI™ and I^2C™ (Master and Slave) modes
- Addressable Universal Synchronous Asynchronous Receiver Transmitter (AUSART)
- Three Capture, Compare, PWM modules:
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10 bits
- Parallel Slave Port (PSP) – 40/44-pin devices only

Special Microcontroller Features:

- Fail-Safe Clock Monitor for protecting critical applications against crystal failure
- Two-Speed Start-up mode for immediate code execution
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Programmable Code Protection
- Processor Read Access to Program Memory
- Power-Saving Sleep mode
- In-Circuit Serial Programming™ (ICSP™) via two pins
- MPLAB® In-Circuit Debug (ICD) via two pins
- $\overline{\text{MCLR}}$ pin function replaceable with input only pin

Device	Program Memory (# Single-Word Instructions)	Data SRAM (Bytes)	I/O	Interrupts	10-bit A/D (ch)	Comparators	CCP (PWM)	MSSP		AUSART	Timers 8/16-bit
								SPI™	I^2C™ (Master)		
PIC16F737	4096	368	25	16	11	2	3	Yes	Yes	Yes	2/1
PIC16F747	4096	368	36	17	14	2	3	Yes	Yes	Yes	2/1
PIC16F767	8192	368	25	16	11	2	3	Yes	Yes	Yes	2/1
PIC16F777	8192	368	36	17	14	2	3	Yes	Yes	Yes	2/1



Apêndice III

Esquema da placa eletrônica utilizada no trabalho (Figura 25).