Universidade Federal do Rio de Janeiro

Escola Politécnica

Departamento de Eletrônica e de Computação

Rastreamento de Objetos Utilizando Processamento de Imagem

Autor:

Simon Medeiros Soares

Autor:

Thiago de Castro Turino

Orientador:

Prof. Mariane Rembold Petraglia, Ph. D.

Examinador:

Prof. José Gabriel Rodriguez Carneiro Gomes, Ph. D.

Examinador:

Prof. Aloysio de Castro Pinto Pedroza, Dr.

DEL

Setembro de 2010

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
Escola Politécnica – Departamento de Eletrônica e de Computação
Centro de Tecnologia, bloco H, sala H-217, Cidade Universitária
Rio de Janeiro – RJ CEP 21949-900

DEDICATÓRIA

Trabalho dedicado à nação brasileira por contribuir para nossos estudos nesta Universidade.

AGRADECIMENTO

Primeiramente a Deus, por nunca me abandonar.
Aos meus pais, Naly e José Maria, por abdicarem de tantas coisas para que eu pudesse estudar.
À minha família por me incentivarem durante esses anos de estudo.
Aos meus amigos, por compartilharem momentos maravilhosos ao longo desses anos.
À professora Mariane, pelos ensinamentos durante o curso e agora como orientadora deste projeto.

Simon Medeiros Soares

Agradeço a Deus, antes de tudo, por ter permitido chegar tão longe.
Aos meus pais que me prepararam desde pequeno para superar os desafios e me passaram valores éticos e morais que me nortearam nesta jornada.
Ao meu irmão Bruno, companheiro para todas as horas, e amigos que compartilharam momentos memoráveis e também me apoiaram nos momentos difíceis.
Aos professores do DEL pelos conhecimentos que foram passados.

Thiago de Castro Turino

RESUMO

Este trabalho apresenta um método de rastreamento de objetos baseado em técnicas de visão computacional. Um dos principais problemas de rastreamento de objetos é a variação do ambiente o qual o objeto está contido. Assim, uma abordagem adaptativa do *background* é utilizada para compensar essa variação *frame* a *frame*. Com base na modelagem do *background*, os *pixels* de foreground são identificados em cada imagem da seqüência e agrupados em regiões conexas. Para cada uma dessas regiões são medidas características através dos momentos da imagem que possibilitam a diferenciação e associação dos objetos de foreground em cena. Ainda, um módulo eletrônico com motor de passo acoplado ao computador via porta paralela é responsável por movimentar o laser para a posição do objeto de interesse.

Palavras-Chave: Visão Computacional, Rastreamento de Objetos, Processamento de Imagens, Motor de Passo, Eletrônica.

ABSTRACT

This paper presents a method for object tracking based on computer vision techniques. A major problem in tracking objects is the variation of the enviroment in which the object is contained. Thus, an adaptative background approach is used to compensate this change frame by frame. Based on modeling of the background, the foreground´s pixels are identified in each image sequence and grouped in related areas. For each of these regions are measured characteristics through of the moments of the image that allow the differentiation and association of foreground´s objects in the scene. Still, an eletronic module sith a stepper motor couple to the computer via the parallel port is responsible for moving the laser to the position of the object of interest.

Keywords: Computer Vision, Object Tracking, Image Processing, Stepper Motor, Eletronic.

SIGLAS

UFRJ – Universidade Federal do Rio de Janeiro
DEL – Departamento de Eletrônica e de Computação
NCE – Núcleo de Computação e Eletrônica
LabIC – Laboratório de Inteligência Computacional
CI – Circuito Integrado

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Equações

1.....Introdução

1.1 Tema

O tema deste trabalho é o desenvolvimento de um sistema automático de rastreamento de objetos a partir de uma imagem digital usando técnicas de visão computacional. Este sistema possui um módulo eletrônico que se assemelha ao funcionamento de uma arma perseguidora. Exemplos de armas perseguidoras podem ser vistos em jogos e filmes onde geralmente são colocadas em um lugar e seguem o alvo enquanto este se move em sua área de visão.

1.2 Delimitação

Em sua mais simples forma, o rastreamento de objetos pode ser definido como o problema de estimar a trajetória de um objeto em uma imagem plana. Entretanto, a tarefa de rastrear objetos pode ser complexa devido a diversos fatores que influenciarão no resultado desejado e, portanto, devem ser identificados e se possível controlados.

Fundamentalmente existem três etapas em análise de vídeo: localização dos objetos em movimento, rastreamento dos objetos separadamente frame a frame e análise do movimento desses objetos para identificação de seu comportamento. Pode-se simplificar o rastreamento através da imposição de restrições ao movimento e/ou forma dos objetos. Por exemplo, quase todos os algoritmos de rastreamento supõem que o movimento do objeto é suave, sem alterações bruscas. Pode-se ainda restringir mais ainda o movimento do objeto com velocidade constante ou aceleração constante com base em uma informação a priori. Os conhecimentos prévios sobre o tamanho do objeto ou a forma do objeto também podem ser utilizados como informação para simplificar problemas. Diversas abordagens para controle do objeto têm sido propostas e estas, principalmente, diferem uma das outras com base na maneiras como abordam as questões: Qual representação do objeto é adequada para o monitoramento? Quais características da imagem devem ser usadas? Como deve ser o movimento, a aparência e a forma do objeto a ser modelado? As respostas a essas perguntas dependem do contexto em que o monitoramento é realizado e um grande número de métodos têm sido

propostos e que tentam responder às necessidades do ambiente a trabalhar.

Por fim, geralmente os maiores problemas enfrentados para projetos de visão computacional são a perda de informações causada pela projeção de um mundo 3D em uma imagem 2D, ruído em imagens, movimento complexo do objeto, oclusões parciais ou totais do objeto, forma complexa do objeto, mudança de iluminação etc. Dessa forma, para o desenvolvimento do trabalho, limites precisam ser impostos e deve-se adotar estratégias específicas de acordo com as características do ambiente de trabalho. Assim, as seguintes limitações serão consideradas no ambiente de teste: apenas um objeto a rastrear, rastreamento apenas de pessoas e iluminação controlada.

1.3 Justificativa

A proliferação de potentes computadores, a disponibilidade de câmeras de alta qualidade e de baixo custo, e a crescente necessidade de automatização de análise por vídeo geraram um grande interesse em algoritmos de rastreamento de objetos. A visão computacional e, mais especificamente, o rastreamento de objetos é pertinente no uso de tarefas como reconhecimento baseado em movimento, vigilância automática, indexação de vídeos, interação homem-máquina, monitoramento de tráfego, navegação veicular dentre outras.

Nos últimos anos têm surgido diversos sistemas que capturam movimento através da análise de vídeo. Mais recentemente a Microsoft lançou o sensor *Kinect* que substitui os famosos *joysticks* permitindo que o jogador interaja com o jogo apenas com o movimento do corpo. Além disso, sistemas de contagem de fluxo de pessoas e de veículos estão sendo utilizados respectivamente em shoppings e estradas em substituição aos sensores feitos por hardware. Ademais, sistemas de interação entre o homem e o computador através de gestos estão sendo apresentados em diversas feiras tecnológicas e indicam que esta será a forma de interação futura entre o homem e a máquina.

Por fim, com base nos diversos itens relacionados acima fica clara a importância do estudo de técnicas de rastreamento e a sua aplicabilidade em diversas áreas de atuação no campo da visão computacional. E, portanto, um método de rastreamento será apresentado ao longo deste trabalho e que poderá servir como base para a construção de sistemas de rastreamento.

1.4 Objetivos

O objetivo geral é construir um sistema de rastreamento robusto e eficiente que possa rastrear objetos e que possa trabalhar com uma margem aceitável de ruído nas imagens. Desta forma, têm-se como objetivos específicos:

- Possuir um rápido tempo de resposta;
- Rastreamento do objeto em duas dimensões;
- Adaptar-se em tempo real às variações do ambiente.

1.5 Metodologia

A primeira etapa no desenvolvimento de um sistema de rastreamento é selecionar, dentro de um conjunto grande, as técnicas para rastreamento que melhor se adaptem não só ao ambiente, mas também às características do objeto a rastrear. Em geral há uma forte relação entre a forma do objeto e o algoritmo para rastreamento, onde normalmente a representação do objeto é escolhida de acordo com o domínio da aplicação. Dessa forma, nesta etapa são definidas quais características do objeto serão procuradas de forma que ele possa ser facilmente distinguido do ambiente. Por exemplo, pode-se procurar por cores, formas geométricas, bordas, textura, dentre outras possibilidades. Feito isso, busca-se determinar de que forma o objeto encontrado será representado no sistema. Essa representação pode ser realizada de diversas formas tal como um único ponto (centro de massa), diversos pontos, bordas, contornos, formas geométricas e etc. Com o objeto representado, realizam-se diferentes testes variando as condições do ambiente e os parâmetros dos algoritmos, onde será possível encontrar a melhor configuração para o sistema de visão computacional.

Por fim, fazendo-se a integração do mesmo com o módulo eletrônico “arma perseguidora”, pode-se verificar a robustez do sistema através dos resultados obtidos.

1.6 Descrição

No Capítulo 2 será apresentada uma visão geral do sistema proposto com o intuito de fornecer ao leitor um apanhado geral do funcionamento. Ainda, será

explicitada uma abordagem superficial das diferentes soluções implementadas em todo o sistema.

Os Capítulos 3 e 4 darão ênfase, respectivamente, às técnicas de visão computacional e eletrônica utilizadas assim como o embasamento teórico das soluções apresentadas.

O Capítulo 5 dedica-se à análise e comentários dos resultados dos testes obtidos. Por fim, as dificuldades encontradas, a conclusão e propostas para trabalhos futuros serão mostradas no Capítulo 6.

2 O Sistema

2.1 Introdução

Neste capítulo será apresentada uma visão geral do sistema proposto de forma a elucidar ao leitor o funcionamento do sistema como um todo. Dessa forma, a melhor maneira de descrever como o mesmo opera é traçar o caminho dos dados junto ao sistema.

Inicialmente um dispositivo de câmera que se encontra fixo é posicionado de forma que o seu campo de visão alcance o objeto a rastrear. Um aplicativo de visão computacional recebe os frames da câmera e extrai informações que revelarão as posições ao longo do tempo do objeto procurado. Essas posições são convertidas para coordenadas do mundo real e enviadas para o módulo eletrônico, que então move a arma para a posição do objeto. Este processo repete-se indefinidamente frame a frame. A Fig. 2.1 apresenta uma visão geral do sistema.

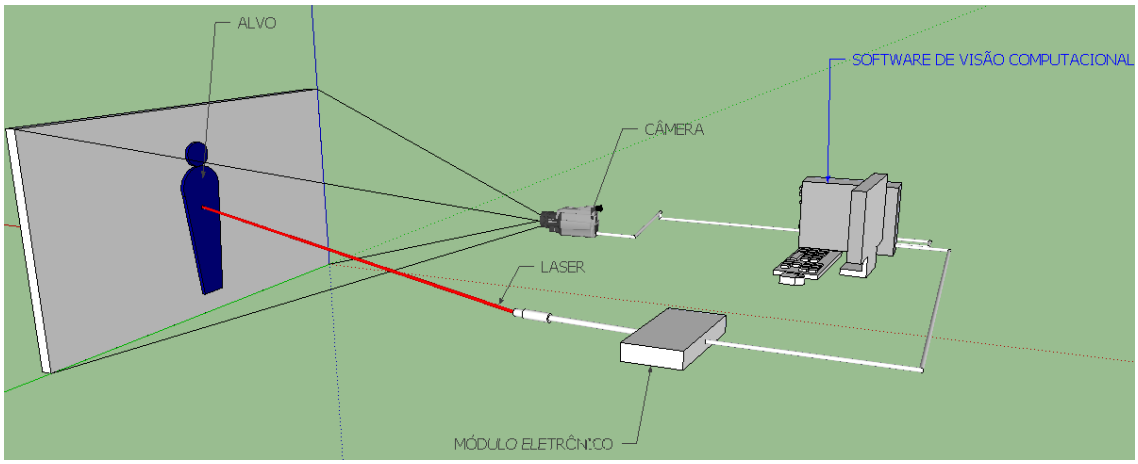

Figura 2.1 - Visão geral do sistema

2.2 Histórico

O histórico de desenvolvimento começa como um projeto desenvolvido para uma disciplina durante a graduação do curso de engenharia. Basicamente o sistema

funcionava de forma semelhante ao apresentado neste trabalho. Entretanto, por questões de simplificação do problema, a abordagem utilizada apresentava algumas limitações de funcionamento. Por exemplo, a localização do objeto era feita através das bordas do objeto onde se utilizava o algoritmo de Canny [1]. Ainda, o rastreamento do objeto era feito através do cálculo dos momentos da imagem de onde se extraía o centro de massa pelas bordas obtidas. Além disso, era necessário que o sistema fosse utilizado com um fundo homogêneo para que um ruído aleatório não fosse confundido com o objeto, o que ocasionaria o deslocamento do centro de massa esperado do objeto.

Por fim, uma outra limitação era o movimento em apenas uma dimensão da "arma perseguidora", sendo permitido o deslocamento apenas sobre o eixo X.

2.3 Nova Abordagem

A nova proposta tem como objetivo tornar indiferente ao sistema o fundo do ambiente de forma que não seja necessário possuir um fundo homogêneo. A solução proposta é fundamentada na extração desse fundo padrão da imagem, que servirá como base de comparação com as futuras imagens do sistema. Assim, no instante que o sistema é iniciado, o programa captura um quadro simples que não deve possuir elementos em movimento ou algum alvo em potencial. Este quadro inicial é chamado de *background* e posteriormente será comparado com qualquer outro frame capturado e representa um ambiente sem qualquer alvo. Feito isto, a imagem resultante conterá, além do objeto a rastrear, outras informações não desejáveis que serão chamadas de ruídos. Esses ruídos precisam ser eliminados de forma que se tenha na imagem resultante apenas a informação pertinente ao sistema. Assim, é aplicado o algoritmo de Grupos Conexos que identificará todos os possíveis objetos na imagem e, através de uma lógica de decisão, efetuará a eliminação de todos os objetos exceto o procurado.

Por fim, a imagem possuirá como informação apenas o objeto desejado e suas coordenadas serão extraídas através do cálculo dos momentos da imagem e posteriormente repassadas ao módulo eletrônico. Este por sua vez, é composto de dois motores de passo conectados ao computador através da porta paralela e serão responsáveis por efetuar o movimento da "arma perseguidora" deslocando-se nos eixos X e Y. A Fig. 2.2 contém um diagrama de fluxo de execução do programa.

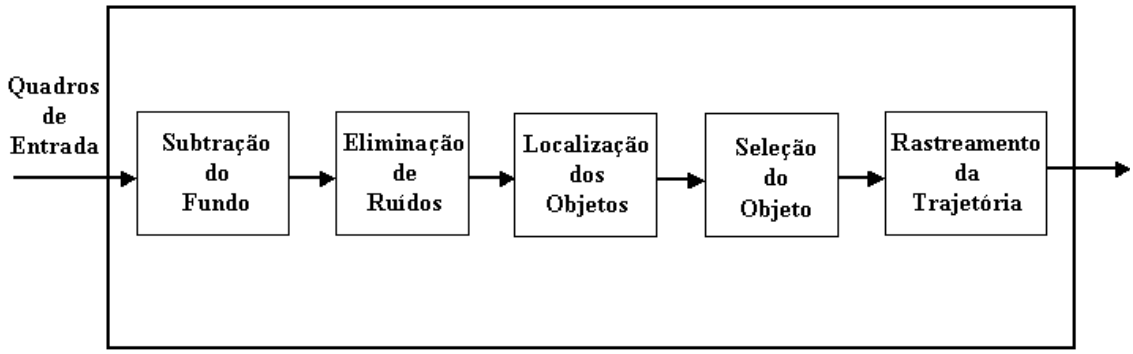

Figura 2.2 – Fluxo sequencial de execução do sistema

3 Processamento de Imagem

3.1 Introdução

Este capítulo está divido em 5 partes onde são apresentadas mais detalhadamente as etapas do algoritmo desenvolvido.

3.2 Subtração do Fundo

A subtração de fundo é uma técnica comumente utilizada para segmentar os objetos de interesse em uma cena para aplicações onde o fundo possa proporcionar confusão com o objeto a rastrear. O nome "subtração de fundo" vem da técnica não estatística de simples subtração entre a imagem observada e a imagem estimada com a posterior limiarização (*thresholding*) do resultado para separar os objetos de interesse.

Existem também as técnicas estatísticas, que são as mais utilizadas por apresentarem melhores resultados onde a mais simples assume uma única gaussiana 3D [2]. Entretanto, como cita Gao [3], esse modelo simplificado não é ideal para aplicações no mundo real, pois interferências como variações de iluminação, de movimento e geométricas não representariam com precisão o ambiente desejado. Uma melhoria substancial na modelagem do fundo é obtida quando se utiliza diversas gaussianas para representar cada cor do *pixel* de fundo. Essa modelagem pode ser implementada de maneira adaptativa e não-adaptativa. No método não-adaptativo os parâmetros das gaussianas são determinados para um conjunto fixo de quadros e os parâmetros do modelo não são atualizados. Na abordagem adaptativa os *pixels* do quadro de interesse para detecção do *foreground* contribuem para a composição da nova mistura de gaussianas. Assim, a cada verificação do quadro há uma atualização dos parâmetros das gaussianas o que os torna variáveis no tempo. Esta técnica implementada por Grimson e Stauffer [4] associa cada *pixel* a uma mistura de Gaussianas, as quais correspondem às probabilidades de se observar uma intensidade particular ou cor no *pixel*. Para isso considera-se o histórico de um pixel como sendo uma serie temporal de seus valores, onde o mesmo será um escalar para imagens em tons de cinza e um vetor para imagens

coloridas. Assim, para um instante t qualquer, a história do *pixel* ï ={xo,yo} é modelada pela equação 3.1:

$$\{X_1, ..., X_t\} = \{I(x_0, y_0, i) : 1 \leq i \leq t\}$$

Equação 3.1 – Representação do histórico do pixel

onde I é a sequência de imagens. Além disso, a recente história de cada *pixel* é modelada por uma mistura de K gaussianas, sendo a probabilidade do *pixel* atual possuir valor Xi,t no instante t é:

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t})$$

Equação 3.2 – Probabilidade de um pixel

onde K é o número de distribuições, Wi,t é o peso associado à *k*-ésima distribuição de gaussianas no tempo *t*, µi,t é a média dos valores da *k*-ésima mistura de gaussianas no tempo t, $\sum i,t$ é a matriz de covariância da *k*-ésima mistura de gaussianas no tempo t e, por fim, η é a função densidade de probabilidade gaussiana que é dada pela equação 3.3:

$$\eta(X_t, \mu, \Sigma) = \left| \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t - \mu_t)^T \Sigma^{-1} (X_t - \mu_t)} \right.$$

Equação 3.3 – PDF Gaussiana

O peso Wi,t possui importância na composição da cor do *pixel* e na variância de cada gaussiana. Assim, inicialmente ajustam-se as gaussianas do plano de fundo e posteriormente todos os *pixels* que não se ajustarem a essa distribuição serão considerados como parte do foreground. Ainda, para os *pixels* do *foreground*, um novo componente gaussiano será adicionado com a média daquele ponto, uma matriz de covariância e um pequeno valor de parâmetro de ponderação. Entretanto, esse método possui um tempo de aprendizado lento no início principalmente em ambientes

ocupados. Além disso, não se pode distinguir entre sombras em movimento e objetos em movimento. Por fim, [5] soluciona esse problema através da reinvestigação das equações permitindo que o sistema aprenda mais rápido e com mais precisão além de adaptar-se mais eficazmente para ambiente em mudança.

A Fig. 3.1 exemplifica a aplicação das misturas de Gaussianas para subtração de fundo com a técnica implementada por [5].

Figura 3.1 – Exemplo de subtração de Fundo: (a) Imagem de uma seqüência em que o alvo está andando pelo ambiente. (b) A média dos pesos mais altos das Gaussianas para cada posição do *pixel*. Eles representam as cores que temporalmente são observadas mais freqüentemente e representam o fundo estacionário. (c) Resultado da subtração do fundo (a média da Gaussiana com o segundo peso mais elevado).

Ainda, por esta abordagem ser uma técnica adaptativa, ela permite a atualização das variações em cena após um intervalo de tempo. Este intervalo dependerá fundamentalmente do número de gaussianas utilizadas para o cálculo adaptativo. Assim, quanto menor o número de gaussianas, mais rápido as diferenças serão incorporadas ao *background*. Por outro lado, quanto maior o número de gaussianas mais lentamente as diferenças serão incorporadas ao sistema. Entretanto, ser incorporado mais rápido significa ter um *background* menos preciso, ou seja, mais sujeito a ruídos. Ainda, mais lento significa possuir um *background* menos suscetível às variações do ambiente; entretanto, isso o torna mais lento na resposta às incorporações de novos objetos em cena. Portanto, a escolha do número de gaussianas é fundamental para se obter a resposta desejada. Para o trabalho seguiu-se a recomendação de [17] onde é citado que geralmente utiliza-se de 3 a 5 gaussianas.

Na Fig. 3.2 mostra-se um exemplo da incorporação de um novo objeto em cena (cadeira) utilizando o mesmo *background* da Fig. 3.1. Percebe-se que ao longo do tempo o novo objeto é incorporado do modelo de *background* capturado.

Figura 3.2 – Subtração de Fundo com inserção de objeto: (a) Imagem de uma seqüência do alvo e do novo objeto inserido. (b) O *background* sendo adaptado com a inclusão do novo objeto em cena. (c) Resultado da subtração do fundo com o alvo e o novo objeto.

3.3 Filtro de Média

O filtro de média, também conhecido como filtro passa-baixas é uma técnica muito utilizada para eliminação de pequenos detalhes na imagem como, por exemplo, ruídos. A idéia é bem simples por realizar a substituição de todo valor do *pixel* em uma imagem pela média dos níveis de cinza de uma vizinhança sendo esta definida por sua máscara. O processo resulta em uma imagem com reduzida transição abrupta nos níveis de cinza. Como os ruídos aleatórios consistem dessas transições, fica claro que a aplicação deste filtro é indicada para a remoção desses. Entretanto, essa transição abrupta é a principal característica de bordas em uma imagem e, portanto, o efeito indesejado pode ser a perda dessas informações [6].

O filtro é realizado através da máscara mostrada abaixo:

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(s,t)\in S_{xy}} g(s,t)$$

Equação 3.4 - Filtro de Média

Este filtro realiza a média dos valores de uma imagem g(x,y) em uma área definida por uma janela (vizinhança) de tamanho mxn o qual todos os coeficientes possuem valor 1/mn. Além disso, por questões de simetria, geralmente m é igual a n e seu valor é um inteiro ímpar.

Voltando ao contexto do problema abordado, a cada frame que é capturado pela câmera surgirão ruídos que serão acrescidos à imagem em questão. Esses ruídos podem ser pequenas variações na luminosidade, instabilidade de posição da câmera e etc. Dessa forma, é muito provável que a imagem atual contenha diferenças em relação à anterior

e, portanto, um filtro de média é uma importante ferramenta para eliminar os elementos indesejados. Entretanto, a utilização deste filtro tem o único intuito de retirá-los apenas da imagem sem destruir o objeto de interesse. Assim, o ideal é a utilização de uma máscara que não produza grandes diferenças em relação à imagem anterior, sendo estas diferenças imperceptíveis a olho humano, mas que para a máquina produzam o efeito desejado. As imagens da Fig. 3.3 apresentam os resultados da utilização do filtro para diferentes tamanhos da máscara.

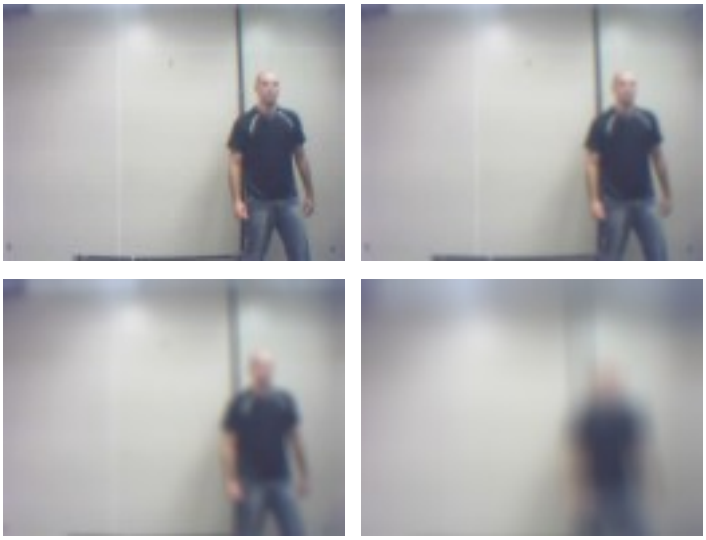

Figura 3.3 - Filtro de Média: (a) Imagem Original (b) Máscara de tamanho: 3x3 (c) Máscara de tamanho: 9x9 (d) Máscara de tamanho: 25x25.

Na Fig. 3.3 nota-se que o filtro passa-baixas provoca o efeito de suavização das imagens (*smoothing*). Por outro lado, ele também possui o efeito indesejado de diminuir a resolução da imagem, provocando assim borramento, ou seja, diminui a nitidez e a resolução da imagem.

3.4 Fechamento

Geralmente os algoritmos de processamento de imagens provocam distorções na imagem o que faz com que a mesma não seja representada de forma próxima da desejada. Por exemplo, a imagem resultante da técnica de remoção de fundo adaptativo poderá produzir uma imagem do objeto de interesse com buracos, pedaços destruídos, ou até mesmo a divisão do objeto em duas ou mais partes. Assim, nesses casos são indicados os usos de operações morfológicas para reparar a imagem.

As operações fundamentais da Morfologia Matemática são a erosão e a dilatação, que se baseiam em testar a imagem contra um elemento estruturante e analisar de que forma este elemento cabe ou não cabe na imagem.

O elemento estruturante é um conjunto definido e conhecido (forma e tamanho), que é comparado, a partir de uma transformação, ao conjunto desconhecido da imagem. Pode assumir várias formas e sua origem pode ser definida em qualquer ponto [8]. As matrizes dos elementos estruturantes podem assumir as formas de linha, coluna, cruz, elipse e quadrado. É o formato destes elementos que determina quais *pixels* da imagem são retirados (erosão) ou quais são adicionados aos objetos (dilatação). A escolha do tipo de matriz a utilizar deve ser em função das propriedades do objeto como convexidade e simetria.

A dilatação é o crescimento ou expansão dos objetos (conjunto de pontos cujo nível de brilho é mais intenso) em relação ao fundo (todos os pontos que não compõem os objetos). Já a erosão é a operação morfológica dual da dilatação, e seu efeito sobre uma imagem é o encolhimento dos objetos em relação ao fundo. As transformações de dilatação e erosão mantêm uma similaridade, de modo que o que uma das operações faz para os objetos a outra faz para o fundo [10].

Dessa forma, para que pudéssemos eliminar as falhas contidas dentro do objeto de interesse a escolha pela operação morfológica dilatação seguida por uma erosão é a mais adequada. Essa operação é conhecida como Fechamento. Na Fig. 3.4 é mostrado o resultado dessa operação.

Figura 3.4 – Operação morfológica fechamento com elemento estruturante dado por uma matriz quadrada de 5x5 com centro na origem.(a) Imagem original. (b) Imagem resultante após a dilatação. (c) Imagem resultante da erosão.

O efeito da dilatação é notado na imagem resultante 3.4 (b), onde a falha encontrada na cabeça do objeto foi restituída. A contrapartida é o "engordamento" da imagem original, ou seja, o aumento da sua área assim como a junção de áreas não desejadas como, por exemplo, a área entre as pernas do objeto. Já a operação de erosão,

Fig. 3.4 (c), faz o processo inverso reduzindo a sua área de forma a torná-la mais próxima da imagem original.

3.5 Grupos Conexos

Segundo Gonzalez [6] a extração dos objetos de interesse é o problema principal em sistemas automáticos de análise de imagens. Essa técnica basicamente atribui etiquetas numeradas (*labels*) a cada objeto da imagem para distingui-los entre si e, uma vez localizados todos os grupos conexos, dados relevantes podem ser extraídos desses grupos tais como área, perímetro, altura, largura, centro de massa dentre outras. Entretanto, para melhor entendimento desse algoritmo, é necessário apresentar o conceito de conectividade que segundo [6] é estabelecida se os *pixels* são vizinhos e se seus níveis de cinza satisfazem um critério específico de similaridade (por exemplo, se seus níveis de cinza são iguais).

Formalmente o "Grupo Conexo" é o nome dado a um grupo de *pixels* que estão conectados entre si. Um *pixel* p nas coordenadas (x, y) está conectado a outro quando está adjacente ao *pixel* em questão. Essa região adjacente ao *pixel* é chamada de vizinhança. Algoritmos de grupos conexos 2D consideram 2 tipos de vizinhança: vizinhança de 4 e 8 *pixels*. Quando a vizinhança é de 4 *pixels*, são considerados conectados os *pixels* que se encontram imediatamente ao laço, acima ou abaixo do *pixel* atual, por outro lado, quando a vizinhança é de 8, as diagonais também são consideradas [11].

O algoritmo de grupos conexos trabalha lendo toda a imagem, *pixel* por *pixel* de cima para baixo e da esquerda para a direita para identificar regiões conectadas, isto é, regiões de *pixels* adjacentes que compartilham o mesmo valor de intensidade V. Para imagens em tons de cinza (V ={0, 1,..., 255}) pode-se considerar uma faixa de valores, por exemplo, V = {110, 111,....,130}. Entretanto, estamos trabalhando com imagens binárias (V ={0, 255}), portanto, o algoritmo varre a imagem procurando por *pixel* branco (V = 255), verificando se algum *pixel* vizinho a ele também é branco. Ao final desse processo, cada conjunto de *pixels* branco recebe um número o qual é chamado de *label* e que começa por 1 e termina no número de grupos conexos encontrados. Quando este processo termina é possível que haja um mesmo grupo com *labels* distintos, assim deve-se varrer a imagem verificando-se se 2 grupos iguais se conectam. Caso isso

ocorra, deve-se agregar o grupo de maior *label* ao grupo de menor *label* e efetuar a reordenação dos grupos restantes.

Como em todo projeto, o custo computacional deve ser levado em conta para que a execução seja realizada dentro do intervalo de captura da câmera. Assim, primeiramente é necessário eliminar alguns grupos com um teste simples.

A primeira eliminação dos grupos é feita durante a execução do algoritmo de grupos conexos onde são eliminados quaisquer grupos cujas áreas sejam menores que um certo valor, o qual chamaremos de área mínima. Essa área foi determinada como sendo de 2% do valor da área total. Portanto, isso evita que tenhamos grupos pequenos e ruídos no resultado final. Um exemplo é ilustrado na Fig. 3.5.

O segundo teste feito deve levar em consideração a característica do objeto a rastrear. Por efeito, as condições do projeto são:

- Apenas um objeto a rastrear;
- Rastreamento apenas de pessoas

Dessa forma, a regra de decisão mais simples para encontrar o objeto de interesse é eliminar todos os grupos exceto o de maior área. Essa regra de decisão não é a mais apropriada para sistemas de rastreamento, pois nos casos ideais podem ser utilizados, por exemplo, classificadores. Entretanto, como o objetivo do trabalho é o estudo da etapa de rastreamento e não a de reconhecimento, essa regra de decisão é válida pelas condições iniciais impostas.

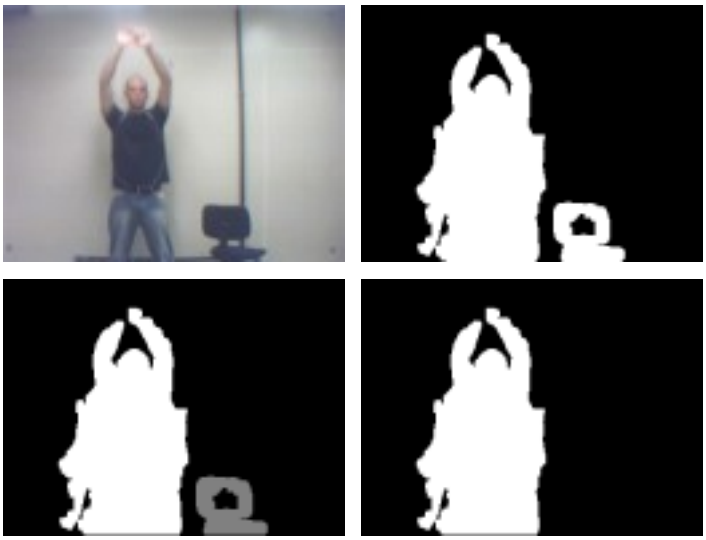

Figura 3.5 – Grupos Conexos: (a) Imagem original. (b) Imagem binarizada. (c) Imagem com todos os Grupos Conexos. (d) Imagem com o Grupo Conexo selecionado.

3.6 Momentos

O trabalho propõe a representação do objeto através do rastreamento das coordenadas do objeto ou, mais especificamente, do centro de massa do objeto. Dessa forma, a análise dos momentos da imagem configura um excelente caminho para rastreamento de objetos.

Uma imagem pode ser representada através de uma função *f* que fornece a intensidade (tons de cinza) da imagem no ponto de coordenadas (x, y). Assim, para uma imagem 2D, o momento de ordem (p + q) é definido como:

$$M_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^p y^q f(x,y) dx dy$$

Equação 3.5 – Momentos da imagem

Entretanto, como estamos trabalhando com imagens binárias, as integrais da equação acima são substituídas por somatórios e são representadas da seguinte forma:

$$m_{pq} = \sum_{1}^{n_x} \sum_{1}^{n_y} x^p y^q f(x, y)$$

Equação 3.6 – Momentos de uma imagem binária

onde nx e ny são respectivamente a altura e a largura da imagem.

A partir dos momentos apresentados anteriormente é possível obter uma grande quantidade de informações tanto geométrica como estatísticas da imagem. Para o projeto estamos interessados apenas nos centróides que podem ser calculados da seguinte forma:

$$x_c = \frac{m_{10}}{m_{00}}; y_c = \frac{m_{01}}{m_{00}}$$

Equação 3.7 – Centro de Massa

onde xc e yc são as coordenadas do centróide, m_{00} representa a área do objeto de interesse, e m_{01} e m_{10} representam as projeções dos pontos de interesse nos eixos X e Y, respectivamente [7]. Um exemplo do cálculo do centro de massa é mostrado na figura 3.6.

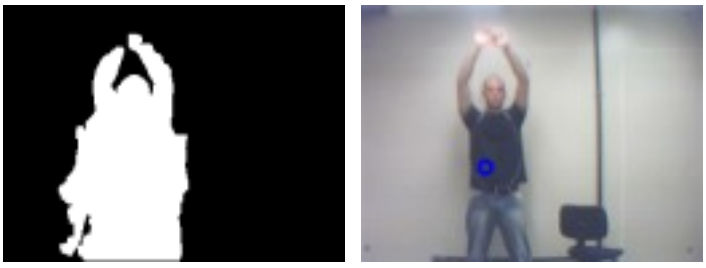

Figura 3.6 – Cálculo do Centro de Massa: (a) Grupo Conexo. (b) Centro de Massa.

4 Eletrônica

4.1 Introdução

O projeto de rastreamento de objetos por software utiliza um mecanismo eletromecânico que permite uma visualização real dos resultados produzidos pelo algoritmo. Para isso, foi necessário usar dois motores de passo para mover um laser tanto no eixo horizontal quanto no eixo vertical, fazendo um movimento chamado de *pan-tilt*, e cobrindo um plano no espaço. Os motores de passo são ideais para esse projeto porque conferem maior precisão no rastreamento do laser. Além da vantagem da alta precisão no posicionamento, eles possuem uma excelente resposta à aceleração e desaceleração, e seguem uma lógica digital, pois seu acionamento é feito através de pulsos elétricos que ativam sequencialmente suas bobinas, facilitando a comunicação com o computador [18].

Como frequentemente é necessário acionar os dois motores ao mesmo tempo, enviando dados simultaneamente, escolhemos a porta paralela para o projeto como meio de comunicação entre o computador e o módulo de hardware, pois ela permite que sejam enviados 8 bits simultaneamente, enquanto que a porta serial, por exemplo, envia um bit por vez.

A seguir serão descritos detalhes do módulo de hardware.

4.2 O Motor de Passo

O motor de passo é um transdutor que converte energia elétrica em movimento controlado através de pulsos, o que possibilita o deslocamento por passos, sendo o passo caracterizado pelo menor deslocamento angular [19]. Eles podem operar na forma unipolar ou bipolar, com passo inteiro ou meio passo. Escolhemos para o projeto o motor do tipo unipolar, por ser mais fácil de controlar. Os motores bipolares trabalham com alternância negativa/positiva de tensão, tornando o controle mais difícil. Para tal, seria necessário um circuito Ponte H.

Para o caso do motor unipolar, é usado o circuito integrado ULN2003 que é um *driver* de potência. Esse CI é composto por sete transistores *Darlington*. Como podemos

constatar, ele funciona como uma espécie de chave, pois assim que ele detecta o nível lógico alto de 5V na saída da porta paralela, ele “fecha” o contato entre a fonte de 12V e o motor, o que faz com que a potência da fonte forneça os 500 mA (o máximo que esse CI suporta em regime permanente, sendo que suporta 600 mA em regime transitório) e faz com que a bobina dentro do motor se polarize, e seu eixo se alinhe (gire) com o campo induzido por essa bobina. Outra maneira de implementar um *driver* seria usando transistores de potência como os BD135 e o DB241. Porém, escolhemos o CI ULN2003 pela praticidade, além de o processo de montagem dos transistores no CI ser automatizado, reduzindo o erro em relação à montagem manual.

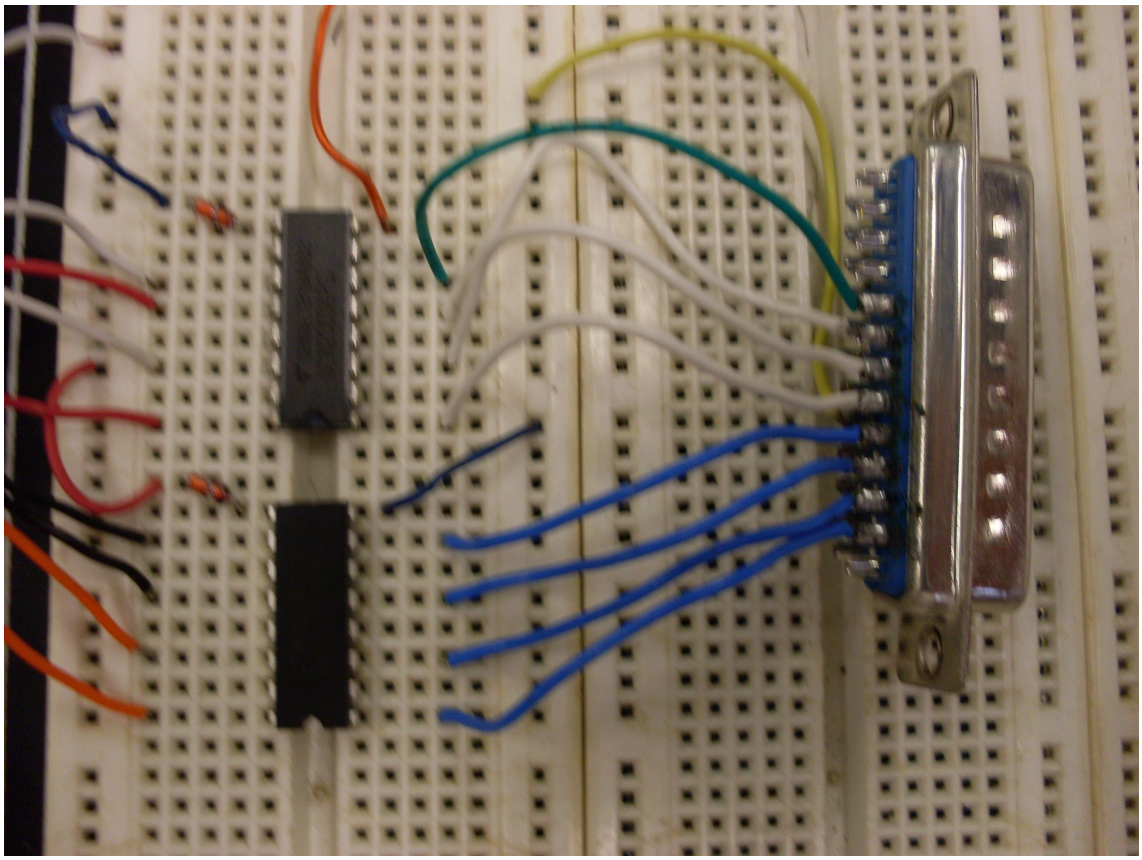

Figura 4.1 – Circuito Adaptador

Os motores unipolares podem ter cinco ou seis fios, sendo quatro desses para polarizar a bobina e um para alimentação. No caso de seis fios, deve-se juntar dois deles para a alimentação. Para determinar quais fios devem ser conectados na alimentação, deve-se medir a resistência entre todos os eles, tomados dois a dois. Apenas três valores podem aparecer: R, $2R$ e infinito. Quando a resistência for $2R$, isto significa que os fios

fazem parte de bobinas diferentes. Quando aparecer resistência *R,* significa que um desses fios é de alimentação, que é comum a todas as bobinas, e o outro fio é o de polarização. Quando a resistência for infinita, significa que eles não estão ligados entre si, portanto nenhum deles é de alimentação.

Um fator importante que se deve levar em consideração é a fonte de alimentação, que deverá fornecer uma amperagem suficiente para fornecer voltagem a dois circuitos integrados ULN2003, dois motores de passo e um laser. Inicialmente utilizamos uma fonte apenas para todos os componentes. Porém, quando o projeto chegou à fase de testes, o brilho do laser passou a diminuir conforme o motor era solicitado. A solução para este problema foi adquirir uma fonte de 12V e 500 mA exclusiva para o laser, deixando a outra fonte com o restante do circuito. A tensão exigida pelo circuito integrado é 12 V, a mesma voltagem é usada para alimentar o motor de passo.

O diodo D1N4007 foi usado para absorver o campo magnético reverso produzido quando o motor é desligado [19].

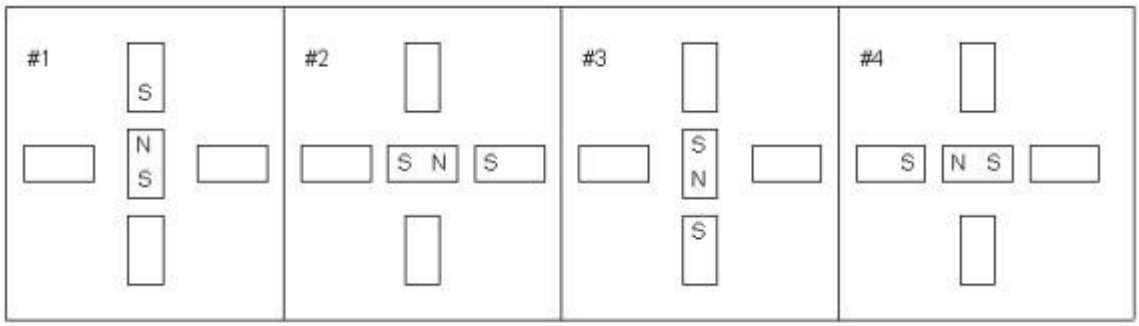

Figura 4.2 – Motor unipolar operando em passo completo

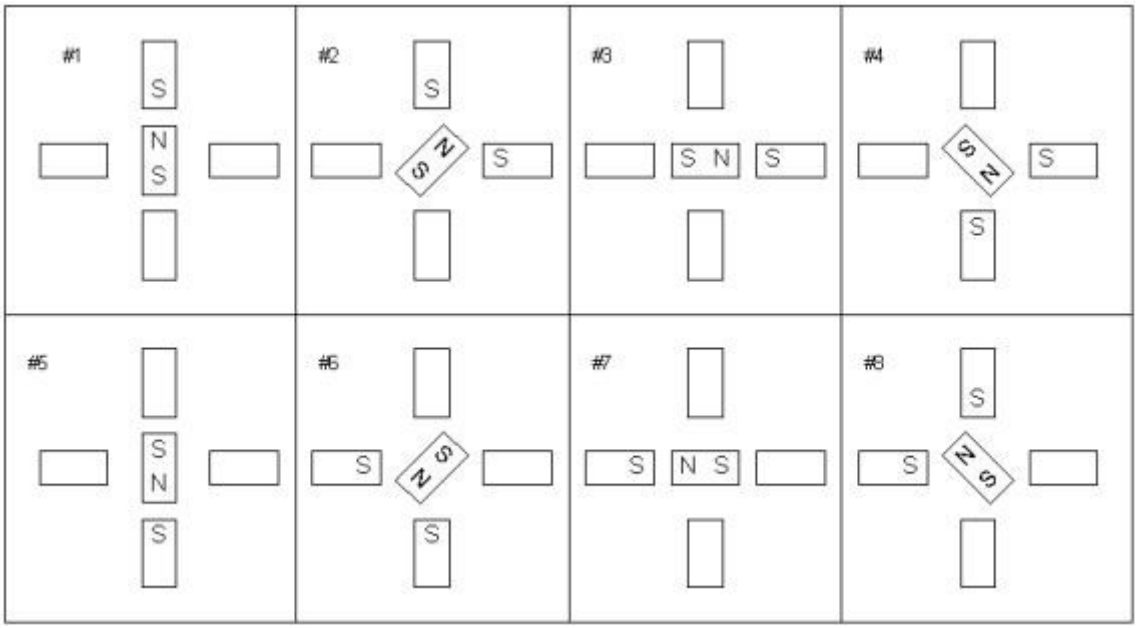

Figura 4.3 – Motor unipolar operando em meio passo

4.3 Comunicação

Conforme foi dito na introdução deste capítulo, escolhemos a porta paralela como meio de comunicação entre o computador e o módulo de hardware porque permite o envio de dados simultaneamente, possibilitando movimentar os dois motores ao mesmo tempo. Cada motor possui quatro bobinas, sendo cada uma delas acionada por um fio. Portanto, temos para o controle de cada motor, quatro fios. Sendo assim, utilizamos oito pinos mais o pino terra do conector DB25.

A porta paralela possui três registradores: dados, status e controle, como podem ser visto na Fig. 4.4. O registrador de status não pode ser usado, pois ele envia dados para o computador, e para esse projeto não precisaremos dessas informações. Escolhemos, então, os pinos situados no registrador de dados porque são exatamente oito e não são inversores. São eles os pinos 2 ao 9, sendo os quatro primeiros para o motor no eixo horizontal e os quatro restantes para o motor no eixo vertical. O pino terra escolhido foi o 18.

No conector DB25, o pino está em nível lógico alto, ou nível lógico 1, quando a sua tensão está entre 3,1V e 5V. Já quando a tensão está entre 0V e 0,4V, dizemos que está em nível lógico baixo ou nível lógico 0.

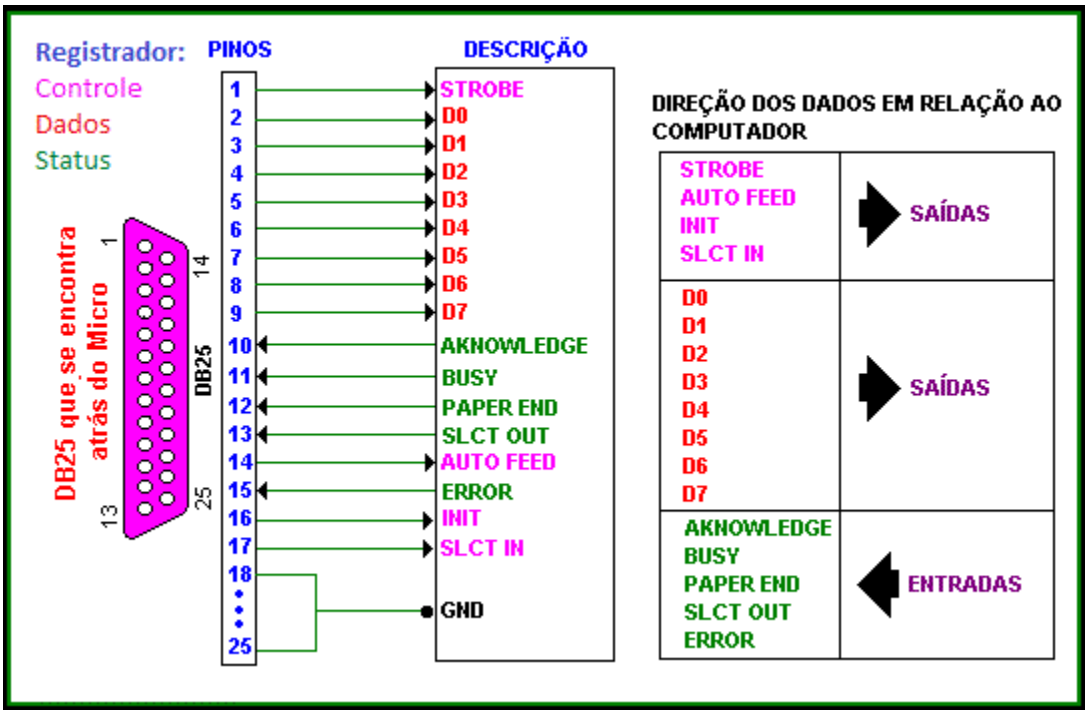

Figura 4.4 – Registradores porta paralela

4.4 Protótipo

Na busca de uma solução para a construção do mecanismo físico do rastreamento, utilizamos uma cantoneira que permitiu uma configuração angular de 90º entre os motores. Dessa forma, a rotação do motor que fica na base (eixo X) gira o conjunto cantoneira mais o motor responsável pelo movimento do laser no eixo Y, permitindo que o laser aponte para qualquer ponto no plano XY.

Inicialmente houve dificuldade de se obter uma cantoneira que tivesse características que não comprometessem o projeto. Dentre as características negativas, ela não poderia ser feita de um material muito pesado, porque devido à assimetria do aparato eletromecânico, o conjunto poderia tombar com o movimento do motor. Também era desejável que o material não fosse muito denso, pois os motores seriam presos através de furos, sendo necessário, portanto, perfurar o metal.

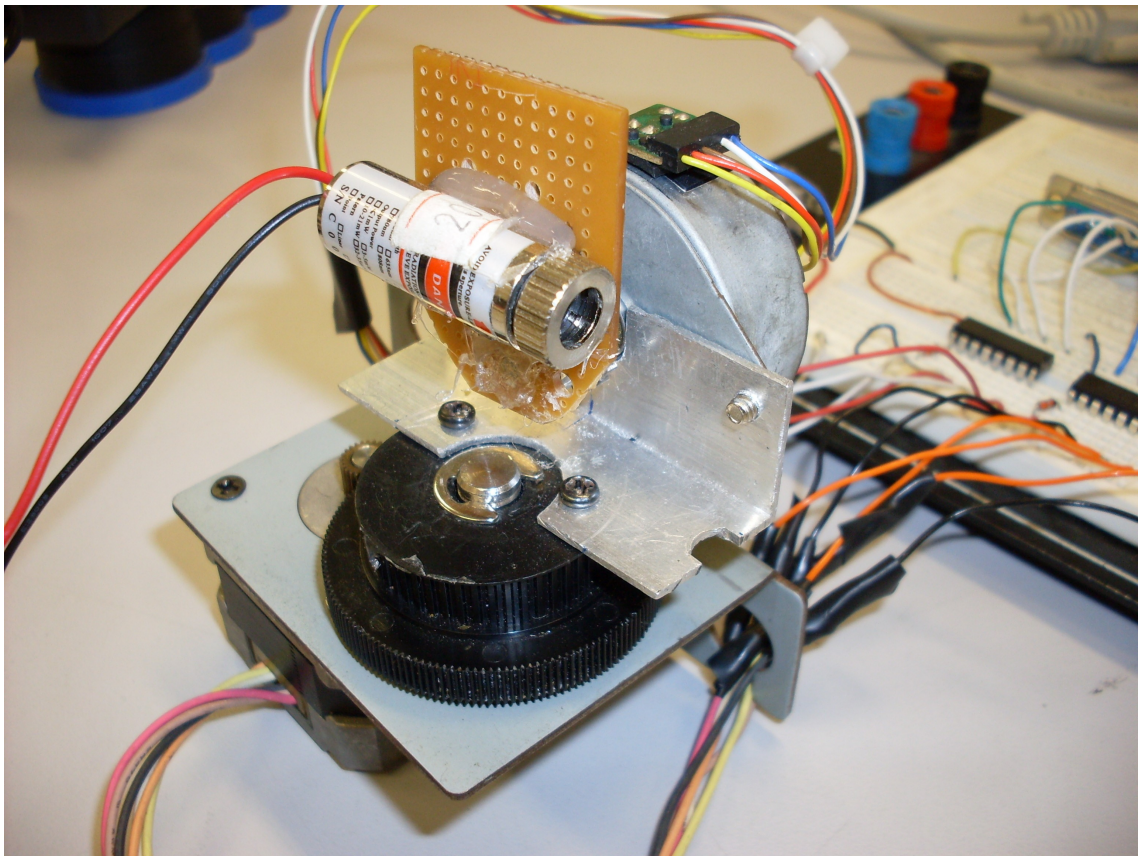

Figura 4.5 – Protótipo

Existem muitos parâmetros que definem um motor de passo. Dentre eles, podemos dizer que o número de graus por passo é o mais importante, pois a principal característica do motor de passo é a precisão. Para obtermos este número nos motores utilizado no projeto, fizemos uma marca em um dos dentes e contamos quantos passos o motor executa para completar uma volta, ou seja, 360º. Sendo assim, encontramos que o motor no eixo X realiza 200 passos para completar esta volta, ou então que um passo realiza um movimento de 1,8º. Já o motor no eixo Y necessita de 160 passos para fazer uma volta completa, ou seja, cada passo executa 2,25º.

Podemos perceber que a precisão do motor no eixo X é maior que a do motor no eixo Y. Mesmo assim, usamos uma peça dentada chamada de redução no motor do eixo X, o que elevou o número de passos de 200 para 1.200, permitindo que agora cada passo realize 0,3º.

Para reduzirmos essa discrepância, configuramos para o motor no eixo Y operar em meio-passo. Com isso, cada passo corresponde à metade do ângulo da situação anterior, passando de 2,25º para 4,50º.

Após a conclusão do módulo do hardware, a etapa seguinte seria a integração com o

software. Nesta fase tivemos algumas dificuldades de adaptação, como na comunicação através da porta paralela. No entanto, podemos finalmente observar a diferença entre o resultado obtido pelo cálculo da posição do objeto e o local para onde o laser estava apontando de fato. Após algumas correções, conseguimos chegar a um erro aceitável.

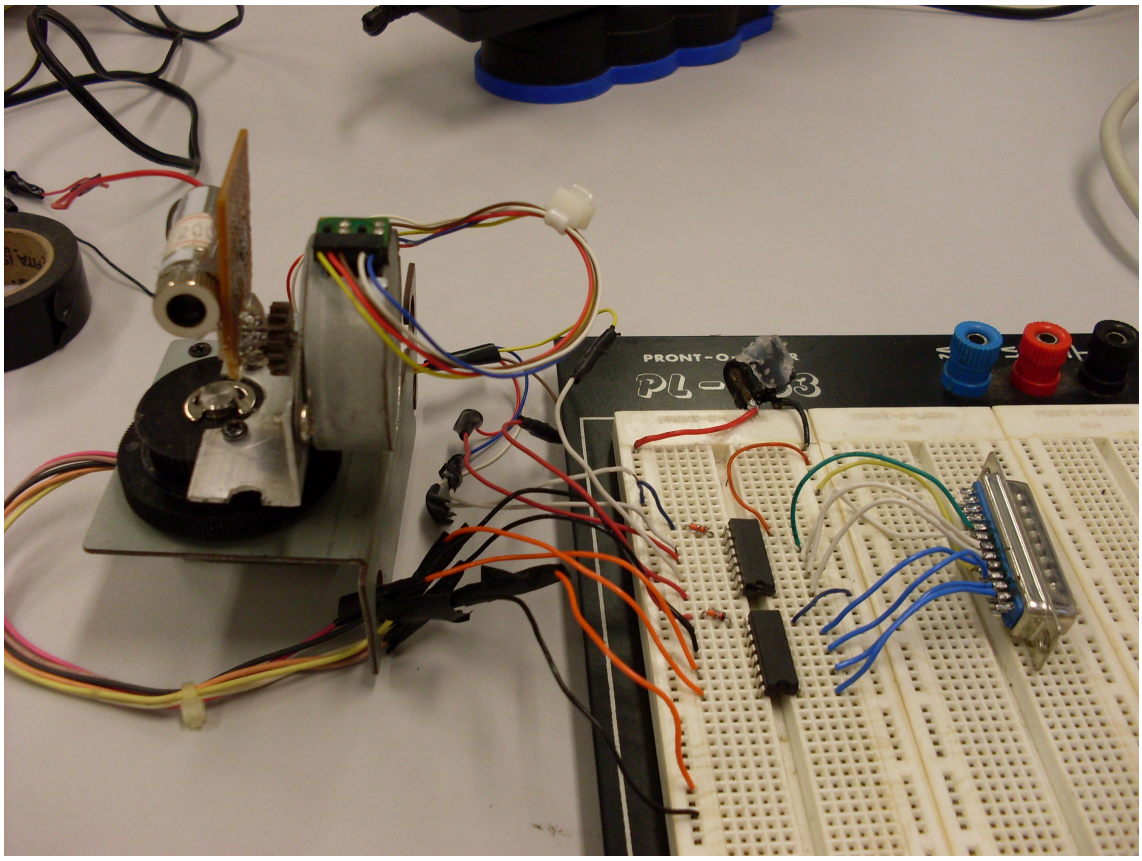

Figura 4.6 – Módulo do Hardware

4.5 Calibração

O sistema de visão computacional representa o objeto de interesse em unidades de *pixels* para fornecer a sua posição. Entretanto, esse mesmo objeto é rastreado pelo motor de passo em um espaço que difere da unidade fornecida pelo sistema. Dessa forma, para que o motor execute corretamente o rastreamento do objeto fica claro que é necessário realizar a transformação dos dados entre os diferentes espaços que o representam.

A idéia desta retificação é descobrir quantos passos o motor precisa executar nos

eixos X e Y, a partir de um referencial, para que ele consiga varrer toda a imagem que está no campo de visão da câmera. É evidente que esse número é dependente da distância obtida no eixo Z da câmera até o fundo, pois quanto maior essa distância maior será o seu campo de visão e, portanto, maior será o número de passos para varrer a imagem por completo. Assim, o processo de calibração deve ser repetido sempre que a distância no eixo Z for alterada. Ademais, ele deve ser realizado com a câmera e o módulo eletrônico localizados no centro do ambiente de teste. Esta condição é necessária, pois se a câmera não estiver localizada fisicamente no centro da imagem, um efeito conhecido como efeito trapézio (*keystoning*) ocorre. Outro problema encontrado é que as lentes das câmeras provocam distorção na imagem, ou seja, uma imagem originalmente retangular ao ser capturada pela câmera possuirá em seus cantos linhas curvas.

Com essas condições uma simples regra de três pode ser utilizada, onde são relacionados o número de passos máximos dos eixos X ou Y, o comprimento ou a altura da imagem gerada pela câmera, e o número de passos que o motor precisar executar. Com isso, a fórmula abaixo pode ser utilizada:

$$NUM\_PASSOS = \frac{diff \times \max\_passos}{tamanho}$$

Equação 4.1 – Relação de calibração

onde:

tamanho: é o comprimento ou altura da imagem gerada pela câmera;

max_passos: é o número de passos necessário para varrer toda a imagem da câmera;

diff: é o deslocamento em *pixels* do objeto;

NUM_PASSOS: é o número de passos necessário para que motor mova-se da posição atual até a desejada.

Obs: o processo de calibração é detalhado no apêndice A.1.1 deste trabalho.

4.6 Controle de Posição do Motor

O módulo eletrônico do sistema recebe frame a frame a posição do objeto de

interesse e realiza a sua perseguição. Podemos dizer que ele não é um sistema inteligente, pois não possui nenhum sensor que forneça a informação da sua própria posição a cada instante. Dessa forma, é possível que em algumas situações ele possa se perder, o que tornará o sistema não robusto. Assim, uma simples solução pode ser resolvida através de lógica de software.

A idéia básica é armazenar a posição do motor de passo a cada movimento do mesmo. Essa posição é dada pelo número de passos que o motor movimentou-se para a direita ou para a esquerda a partir de um referencial previamente definido. Assim, cada vez que o sistema efetua o cálculo do número de passos, é realizada uma verificação para certificar-se de que esse número não excede o máximo de passos (definido na etapa de Calibração) que o motor pode movimentar-se em cada eixo. Isso evita que o motor saia do campo de visão da câmera. Com isso, caso o motor receba a informação para executar um número de passos que faça com que o mesmo saia do campo de visão da câmera, ele não executará o número de passos calculado, mas sim será deslocado para os extremos (limites mínimo ou máximo) do campo de visão.

5 Análise dos Resultados

5.1 Introdução

Neste capítulo serão apresentados os resultados para os dois testes com diferentes configurações com o intuito de apurar a acurácia a robustez do sistema.
Os três vídeos são compostos por uma câmera fixa e um único objeto em movimento a rastrear. Ainda, nos três vídeos, um fundo padrão é capturado antes do sistema entrar em operação.

5.2 Configurações do Teste

Os testes são compostos por uma *webcam* fixa em uma sala com dimensão de 18 metros quadrados com a parede modelo para extração do *background* possuindo dimensão de 3 metros. Este fundo padrão é capturado antes do sistema entrar em operação. Esta opção foi feita para facilitar a aplicação na etapa de identificação do objeto.

A resolução de trabalho da câmera é de 320x240 *pixels* com 256 cores em RGB, o que nos dá uma área total de 76800 *pixels*. Ela está posicionada no eixo Z a uma distância de 4 metros da parede modelo.

Para auxílio na etapa dos testes, foi elaborada uma interface gráfica no Borland C++ Builder 6.0, com o Windows XP para ambiente de teste.

Conforme citado na Seção 3.5, o sistema de visão computacional eliminará todos os objetos encontrados na imagem com área menor que 1536 *pixels* (2% da área total).

Por fim, conforme limitação imposta para o projeto na Seção 1.2, o ambiente de teste possuirá a seguinte característica:

- Apenas um objeto a rastrear;
- Rastreamento apenas de pessoas;
- Iluminação controlada

Antes de iniciar o sistema, é necessário realizar a calibração conforme citado na Seção 4.5. A tabela 5.1 relaciona a configuração de trabalho da câmera e o número de passos obtido para varrer a imagem por completo.

	tamanho *(pixels)*	max_passos (passos)	Precisão (pixels/passo)
X	320	160	2
Y	240	18	13,33

Tabela 5.1 – Parâmetros de Calibração

Agora, com todas as etapas do sistema configuradas, o projeto já pode ser testado.

5.3 Testes

A etapa de testes consiste em verificar se o sistema está rastreando o alvo com precisão. Para isso, existem duas maneiras distintas: a primeira, mais simples, consiste na simples observação da direção da luz do laser. Se ao longo do tempo o laser está sempre posicionado sobre o objeto, pode-se dizer que o sistema é preciso; a segunda consiste em armazenar a posição do laser ao longo do tempo e com isso gerar curvas da trajetória do laser para comparar com as curvas da trajetória do alvo. Entretanto, como o módulo eletrônico atua apenas recebendo informações do computador, essa medição da posição do laser será realizada de forma estimada. Assim, sabendo-se o número de passos que os motores executam ao longo do tempo e combinando-os com a informação de precisão do laser (obtida na calibração) é possível estimar a posição atual do laser.

O sistema foi analisado com o alvo movimentando-se em diversas posições, testando condições limites onde as falhas podem ser encontradas. Foram realizados três testes do sistema. O primeiro investiga a movimentação do motor através dos eixos X, o segundo através do eixo Y e o terceiro compõe os movimentos dos dois eixos simultaneamente.

5.3.1 Teste 1 – Movimento no eixo X

Na Fig. 5.1 são apresentadas as duas curvas dos movimentos do alvo e do motor ao longo do eixo X.

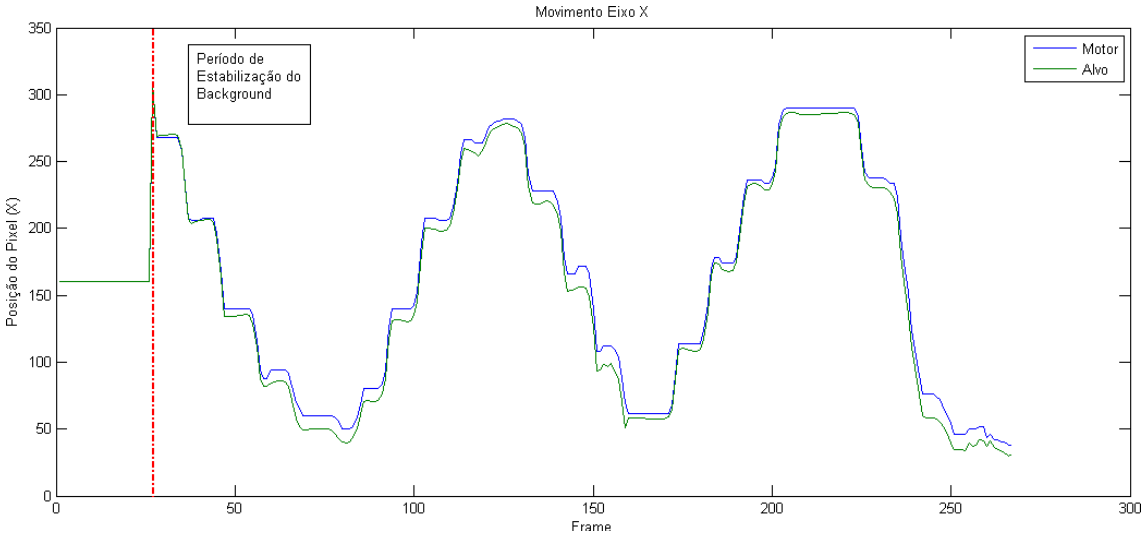

Figura 5.1 - Posição do objeto e do laser ao longo do tempo no eixo X

Durante o início do rastreamento, e ainda sem o objeto de interesse, há um período no qual os *frames* recebidos pelo sistema são levemente distintos do *background* modelo capturado. Essas diferenças proporcionam a detecção de falsos objetos em coordenadas aleatórias da imagem capturada. Portanto, durante esse período, os motores de passo recebem essas informações e deslocam-se para os pontos calculados. Geralmente esse período leva 30 frames o qual chamamos de Período de Estabilização do B*ackground* e está representado nos gráficos pela linha tracejada em vermelho.

Sobre as curvas de deslocamento traçadas fica claro que elas apresentam trajetórias similares, de onde se pode concluir que o motor consegue rastrear o objeto com elevada precisão. Ainda, realizando-se a subtração das duas curvas é possível obter o erro médio ao longo do tempo e seu desvio padrão.

Erro Médio (pixels)	Desvio Padrão (pixels)
6,76	4,67

Tabela 5.2 – Resultado da comparação das posições do laser e objeto no eixo X

Para a medição da velocidade do motor, inicialmente pensou-se em trabalhar em unidades de pixels por segundo. Entretanto, essa medida não auferiria informação correta da real velocidade do motor, pois como os motores possuem precisões distintas, o motor que possuísse menor precisão percorreria mais pixels em menos tempo do que um motor com precisão menor. Assim, decidiu-se mensurar a velocidade do motor em unidades de passos. A Fig. 5.2 apresenta o número de passos com o seu respectivo tempo.

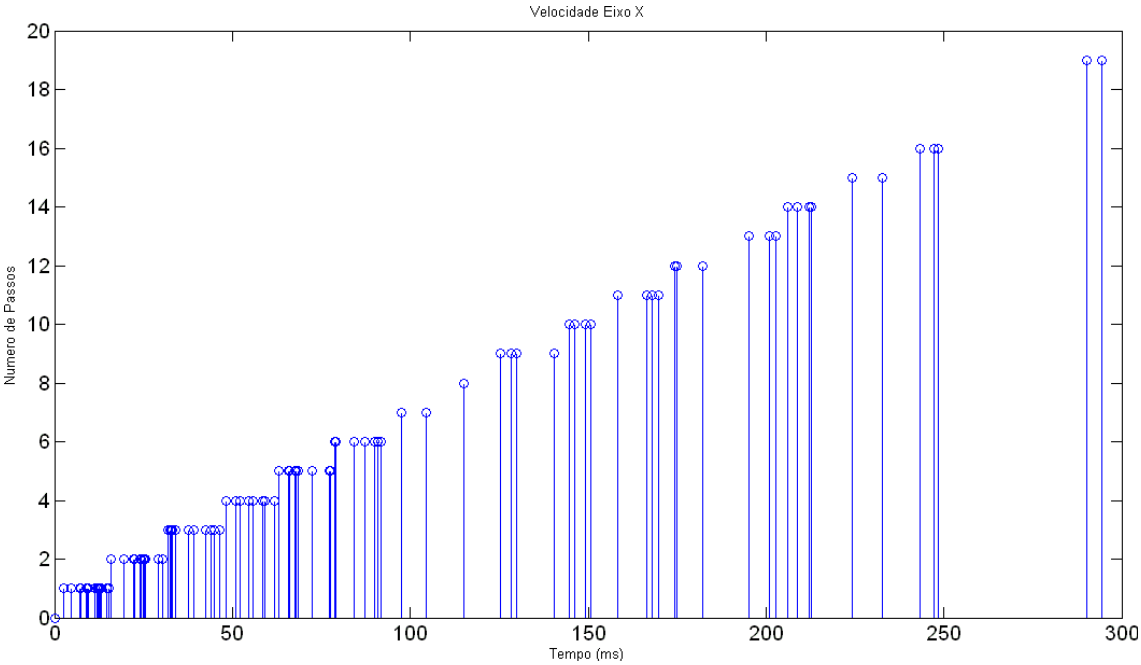

Figura 5.2 – Velocidade do motor no eixo X

Da Fig. 5.2, percebe-se que a resposta do motor comporta-se de forma linear, ou seja, o tempo gasto e a quantidade de passos são diretamente proporcionais. Além disso, verifica-se uma boa distribuição da quantidade de passos o que representa que o motor responde bem a pequenas variações na posição do eixo X.

Por fim, na tabela 5.3 é apresentada a velocidade média do motor no eixo X.

Vel. Média (passos/s)
68,17

Tabela 5.3 – Velocidade Média do Motor no Eixo X

5.3.2 Teste 2 – Movimento no eixo Y

Neste segundo teste o alvo movimentou-se no eixo Y a fim de testar se o sistema é capaz de rastrear enquanto apenas uma das posições deve variar.

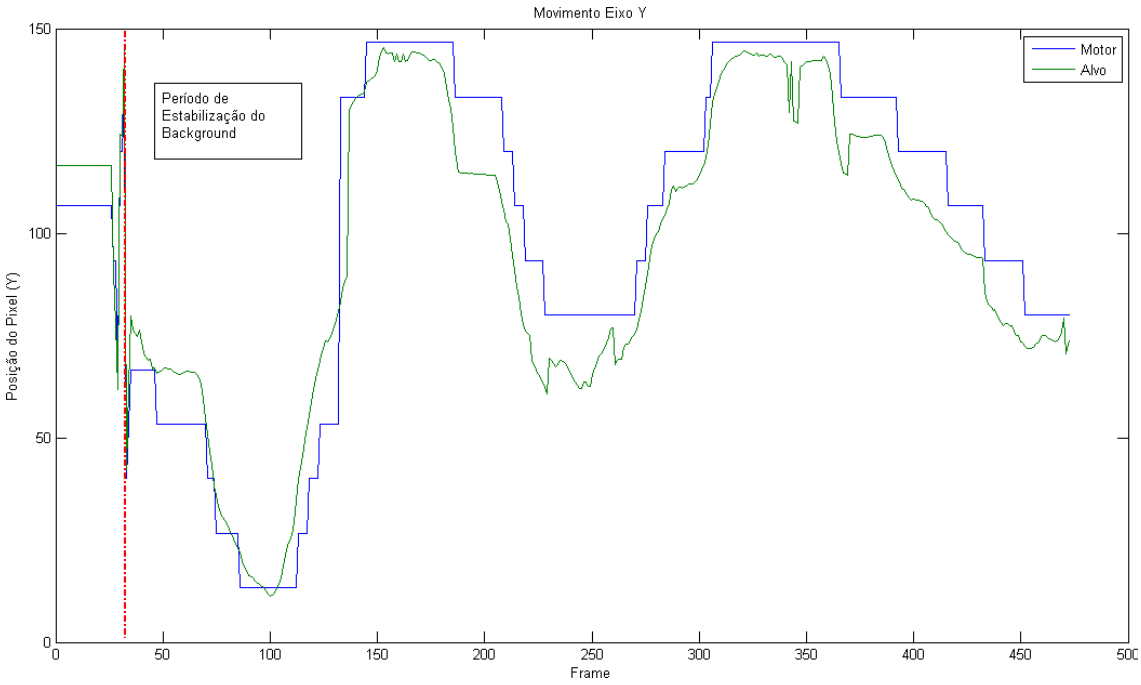

Figura 5.3 – Posição do objeto e do laser ao longo do tempo no eixo Y

A tabela 5.4 apresenta o erro médio obtido neste teste e seu desvio padrão.

Erro Médio (pixels)	Desvio Padrão (pixels)
10,66 pixels	11,65 pixels

Tabela 5.4 - Resultado da comparação das posições do laser e objeto no eixo Y

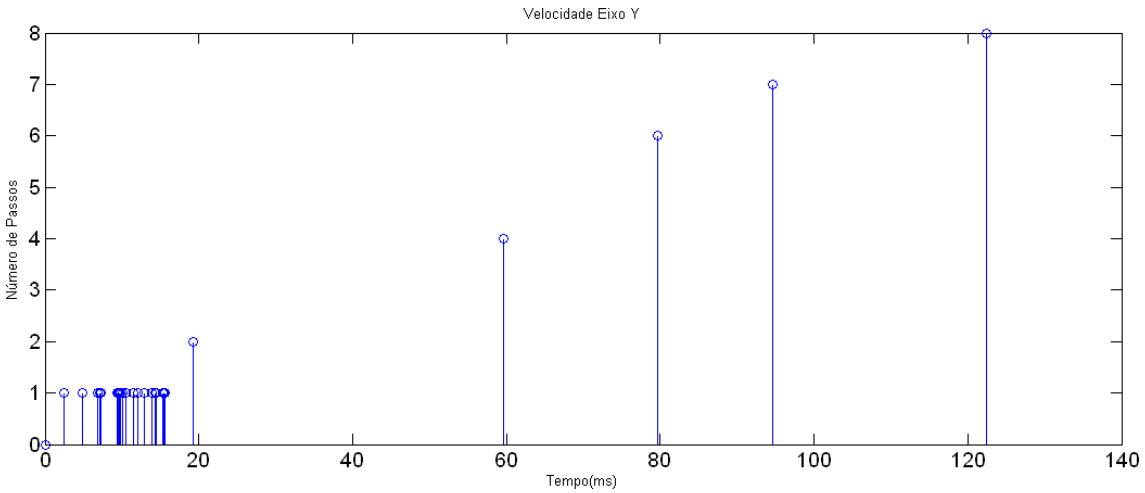

Figura 5.4 – Velocidade do motor no eixo Y

A Fig. 5.4 apresenta um grande número de passos com valor unitário e outros cinco pulsos distribuídos ao longo do eixo do tempo. Isso era esperado devido à reduzida precisão desse motor assim como à limitada e consequente pequena variação no eixo Y proporcionado pelo alvo. Entretanto, assim como no gráfico do motor para o eixo X, percebe-se que o gráfico para o eixo Y também apresenta a característica de linearidade. Portanto, a medição da velocidade média é possível.

Vel. Média (passos/s)
79,68

Tabela 5.5 - Velocidade Média do Motor no Eixo Y

5.3.3 Teste 3 – Movimento nos eixos X e Y simultaneamente

Neste último teste o alvo movimentou-se nos dois eixos a fim de testar se o sistema é capaz de rastrear quando as duas posições variam simultaneamente.

Para melhor visualização, os gráficos dois eixos foram são apresentados separadamente. A Fig. 5.5 apresenta o gráfico referente ao eixo X.

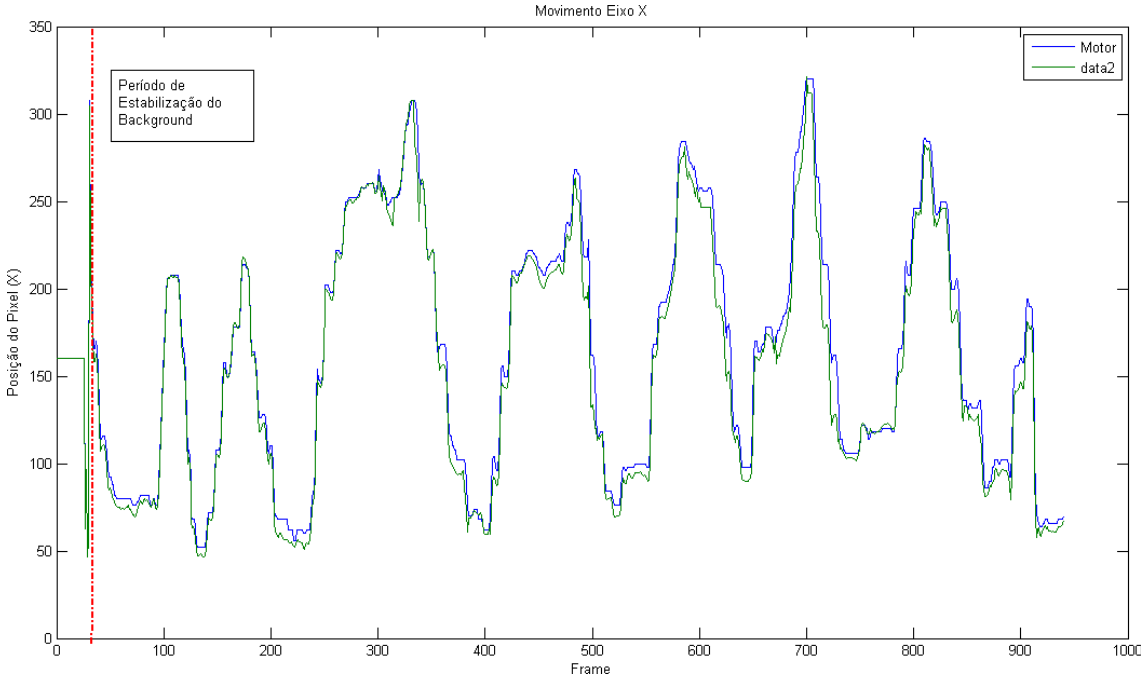

Figura 5.5 - Posição do objeto e do laser ao longo do tempo no eixo X

O erro médio da Fig. 5.5 e o seu desvio padrão estão na Tabela 5.6.

Erro Médio (pixels)	Desvio Padrão (pixels)
7,59 pixels	11,12 pixels

Tabela 5.6 - Resultado da comparação das posições do laser e objeto no eixo X

A figura 5.6 apresenta os resultados para o eixo Y.

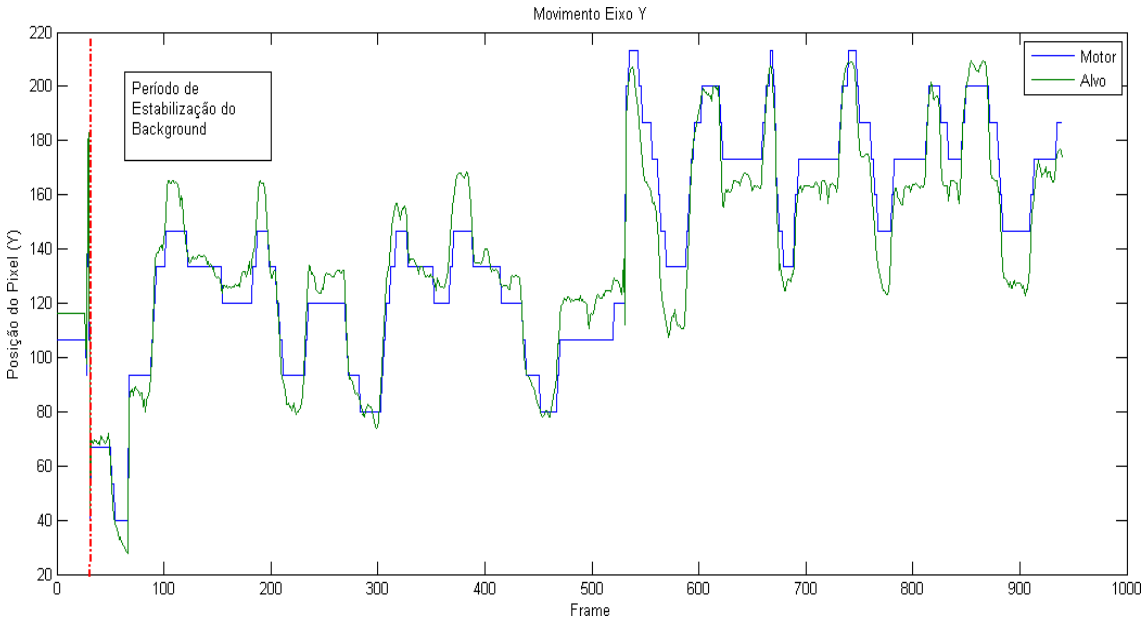

Figura 5.6 - Posição do objeto e do laser ao longo do tempo no eixo Y

O erro médio da Fig. 5.6 e o seu desvio padrão encontram-se na tabela 5.7.

Erro Médio (pixels)	Desvio Padrão (pixels)
9,40 pixels	11,12 pixels

Tabela 5.7 - Resultado da comparação das posições do laser e objeto no eixo Y

Dos resultados acima, verifica-se que os erros apresentados nos eixos X e Y em movimento conjunto praticamente mantiveram-se estáveis quando comparados aos erros verificados nos movimentos dos eixos X e Y separadamente. Portanto, conclui-se que o sistema de *pan-tilt* para o laser permite o movimento dos eixos tanto na forma independente como simultaneamente. Ainda, a pequena diferença entre os erros médios dos três testes já era esperada por dois motivos: o primeiro deles é o próprio processo de medição da posição do motor que apresenta falhas, assim como a sua calibração; o segundo é o erro de precisão do passo que é inserido quando o motor atua sob alguma carga. Ademais, com relação à carga sobre o motor, é sabido que a mesma provoca mudança na velocidade média do motor dependendo da carga utilizada. Cargas maiores ou menores provocarão mudanças no torque e na velocidade do motor. Portanto, a velocidade medida é válida somente para a carga considerada.

5.4 Tempo de Execução

A câmera do projeto trabalha com freqüência de 30 frames por segundo. Portanto, o tempo total do algoritmo deve ser inferior a 33 milisegundos. Assim, utiliza-se um contador com precisão de 64 bits para medir o tempo. A tabela 5.8 contém os tempos médios obtidos.

Etapa	Tempo (ms)
Processamento de Imagem	13.06
Motor de Passos	2.04
Total	15.10

Tabela 5.8 – Tempo médio de execução do sistema

Os tempos de execução obtidos acima foram realizados através de uma média, onde fica claro que o mesmo é diretamente proporcional ao número de objetos presentes em cena assim como da quantidade de passos executados pelos motores.

6 Conclusão

6.1 Problemas Encontrados

Durante o desenvolvimento do projeto surgiram diversas dificuldades sendo a eletrônica o módulo que apresentou os principais problemas. Apesar de o circuito e a operação do módulo do hardware serem relativamente simples, algumas dificuldades apareceram no decorrer do projeto, principalmente pela falta de equipamentos adequados. A montagem do cabo paralelo sem o auxílio de um instrumento ótico, como uma lente, tornou a tarefa árdua, pois os fios devem ser soldados nos pinos com muita perícia. Como os pinos ficam muito próximos, deve-se tomar o cuidado de não deixar o fio desencapado, pois com a manipulação do cabo esses fios podem entrar em contato e provocar um curto-circuito. Isso pode alterar a sequência de polarização das bobinas, fazendo com que o motor não gire em apenas um sentido.

A montagem da parte mecânica do aparato foi uma dificuldade à parte pela falta de experiência nessa área. O conjunto não poderia estar desnivelado, obrigando o laser percorrer uma reta paralela a um dos eixos quando apenas um motor é acionado. Caso ao contrário, o programa teria dificuldades de determinar as configurações iniciais. Como o auxílio da equipe do Laboratório de Inteligência Computacional do NCE (LabIC), que emprestou as ferramentas e seus conhecimentos, essa etapa não chegou a atrasar o cronograma do projeto.

Na parte de software o principal problema enfrentado foi o comportamento do motor de passos quando recebia informações do programa. Originalmente, o programa foi construído para enviar as informações dos movimentos dos eixos X e Y em uma mesma mensagem. Entretanto, reparou-se que o motor de passos perdia-se quando a informação era enviada dessa forma. Assim, precisou-se alterar o programa de forma que ele enviasse as informações de movimento do motor em mensagens separadas e uma após a outra.

6.2 Conclusões

De uma forma geral, o presente trabalho conseguiu alcançar os objetivos

definidos para esse trabalho. O sistema de visão computacional consegue rastrear objetos e o módulo eletrônico apresentou resultados satisfatórios dentro da margem de erro considerada. É evidente que o sistema atual funciona dentro das limitações impostas, entretanto ele serve como base para a construção de um sistema mais completo, robusto e eficiente.

6.3 Trabalhos Futuros

O projeto, por suas características, permite a implementação de uma série de melhorias tanto na parte de hardware como na de software.

No que tange ao módulo da arma perseguidora, a primeira, e talvez a mais importante melhoria, é a implementação de um sistema que permita fornecer a posição do motor de passo a cada instante. Essa melhoria pode ser feita de diversas formas onde como sugestão será apresentada uma solução por software e outra por hardware.

A solução por hardware pode ser implementada utilizando para isso um simples sensor acoplado ao motor de passo. Já a solução por software exigirá o uso de um apontador laser com maior potência e uma câmera com melhor resolução. A posição do laser será localizada a cada instante através de técnicas de visão computacional que identificarão a cor do laser e fornecerão a sua posição. Ainda, com isso, o sistema poderá passar a rastrear o objeto através do erro que é definido como a diferença de posições entre o motor de passo e o objeto.

Outra melhoria importante é um sistema de calibração mais eficiente devido aos problemas relatados na Seção 4.5. Para que essa melhoria seja implementada, deve-se assumir que o sistema controle de posição do motor de passo esteja implementado. Com isso, as posições do motor de passo serão associadas a pontos nas coordenadas da câmera através de uma matriz de homografia [16].

Por fim, no que tange apenas ao sistema visão computacional, pode-se expandir o sistema para que diversos objetos possam ser rastreados simultaneamente. Essa solução implica na melhoria dos algoritmos de processamento de imagens e talvez da utilização de algum classificador. Além disso, pode-se implementar um sistema de visão estéreo para que objetos em diferentes posições no eixo Z possam ser identificados.

7 Bibliografia

[1] CANNY, J., A Computational Approach To Edge Detection, IEEE Trans. Pattern Analysis and Machine Intelligence, 8(6):679-698, 1986

[2] WREN, C. R.; AZARBAYEJANI, T. D.; PENTLAND, A. P. 1997. Pfinder: Real-time tracking of the human body. IEEE Trans. Pattern Anal. Mach. Intel., 19: 780-785.

[3] GAO, X.; BOULT, T.; COETZEE, F.; RAMESH, V.2000. Error analysis of background adaption. In IEEE Conference on Computer Vison and Pattern Recognition (CVPR). 503-510.

[4] STAUFFER, C.; GRIMSON, W. 2000. Learning patterns os activity using real time trakcing. IEEE Trans. Patt. Analy. Mach. Intel 22, 8, 747-767.

[5] KAEWTRAKULPONG, P.; BOWDEN, R. 2001. Adaptative Visual System for Tracking Low Resolution Colour Targets. In British Machine Vision Conference. University of Manchester..

[6] GONZALEZ, R. C., "*Digital Image Processing Using Matlab*", Segunda Edição, Gatesmark Publishing, 2009.

[7] CHAVEZ, G. C. Sistema celular para reconhecimento de padrão invariante. In IV Workshop em Tratamento de Imagens, páginas 2-3, 2003.

[8] VIEIRA, E.A.;MELO, R.H.C. Granulometria: Uma Aplicação para Contagem e Medição de Grãos em Imagens Digitais, Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal Fluminense, 2004.

[9] MELLO, C.A.B Morfologia Matemática, Notas de aula na Graduação em Engenharia da Computação – UPE, Poli, D.Sc., 2008.

[10] COSTA, J.A.T.B; SILVA, A.M.M; JARVORSKY, C.S. Técnicas de Processamento de Imagem para Segmentação de Fases em Imagem de Microssonda Eletrônica, In I Workshop em Microssonda Eletrônica, 1995.

[11] TRALLY, D.G.M; Ajuste Automático de uma Câmera de Vídeo, Projeto de Graduação em Engenharia Eletrônica e de Computação, Universidade do Rio de Janeiro, Escola Politécnica, Departamento de Eletrônica e de Computação, 2007.

[12] SANTOS, C.A; Morfologia Matemática em Radiologia, Seminário de Introdução à Visão Computacional, 1998.

[13] OPENCV LIBRARY, Open Source Computer Vision (Intel), in http://sourceforge.net/projects/opencvlibrary/, 2010.

[14] INPUOUT32 DLL, in http://logix4u.net/Legacy_Ports/Parallel_Port/Inpout32.dll_for_Windows_98/2000/NT/XP.html, 2010.

[15] PORTA PARALELA, in http://www.rogercom.com/, 2010.

[16] HOMOGRAPHY MATRIX, in http://plus.maths.org/issue23/features/criminisi/index.html, 2010.

[17] PICCARDI, M.; Background Subtration Techiniques: a review, in Faculty of Information Technology, Sydney, Australia, 2004.

[18] BRITES, F.G; SANTOS, V.P.A., Motor de Passo, Programa de Educação Tutorial, Grupo PET-Tele, Curso de Engenharia de Telecomunicações, UFF.

[19] Estudo do Motor de Passo e seu Controle Digital, in http://www2.eletronica.org/artigos/outros/estudo-do-motor-de-passo-e-seu-controle-digital, 2010

A Apêndice

A.1 Software

Neste tópico são apresentados os comandos disponíveis na interface gráfica do aplicativo de visão computacional. Essa interface foi criada para facilitar a etapa de calibração, ajuste do sistema e verificação de algum erro com o módulo eletrônico.

A.1.1 Calibração

A calibração do sistema se dá de forma bem simples. Foram criados quatro botões que permitem ao usuário movimentar o motor de passos sem a necessidade de o sistema estar em operação. Cada botão movimenta o motor nos seguintes sentidos: esquerda, direita, para cima e para baixo.

Para melhor calibração, o usuário deve posicionar de preferência o motor nos pontos (x,y) = (0,0) do campo de visão da câmera. Feito isso, utilizando-se os quatro botões deve-se movimentar o motor nos sentidos X e Y e realizar a contagem do número de passos executados para varrer a imagem por completo. Por fim, através de um arquivo de inicialização o número de passos deve ser passado ao sistema.

A.1.2 Ajuste do Background

A função desta etapa é realizar a captura de um fundo padrão que servirá de comparação com os frames posteriores. O usuário deve posicionar a câmera de forma que só haja objetos estáticos e que o objeto de interesse não esteja presente em cena. Feito isso, deve-se iniciar a captura através do botão "*Get*" e posteriormente utilizar o botão "*Save*" para salvar o fundo modelo.

A.1.3 Execução

Tendo realizado as etapas de calibração do sistema e de captura do fundo padrão o sistema pode entrar em operação. Para isso são disponibilizados os botões "*Start*" e "*Stop*" que iniciam e interrompem respectivamente o rastreamento do objeto.

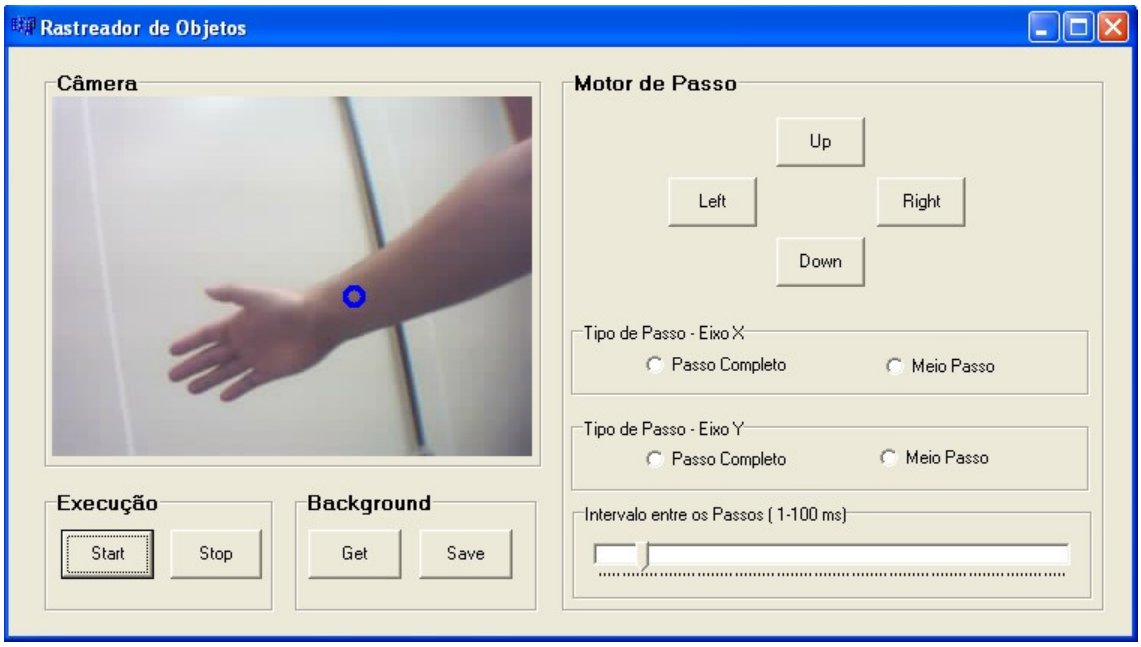

Figura A.1 – Interface do software

A.2 Lista de Materiais

Item	Quantidade
Motor de passo unipolar	2
Circuito Integrado ULN 2003	2
Diodo (D1N4007)	2
Suporte para o laser (cantoneira)	1
Laser 5mW	1
Cabo porta paralela	1
Conector DB25	2
Protoboard 1650 pontos	1
Fonte 12V / 500mA	2
WebCam 30fps	1
Computador Dual Core	1

Tabela A.1 – Lista de Materiais (hardware)

Item	Finalidade	Fonte
Borland 6.0 C++ Builder	Interface Gráfica	Licença LabIC/NCE
OpenCV Library	Processamento de Imagem e Captura de Vídeo	Open Source
InpOut32 dll	Comunicação Porta Paralela	Open Source
Windows XP sp3	Sistema Operacional	Licença Acadêmica

Tabela A.2 – Lista de Materiais (software)

A.3 Circuito

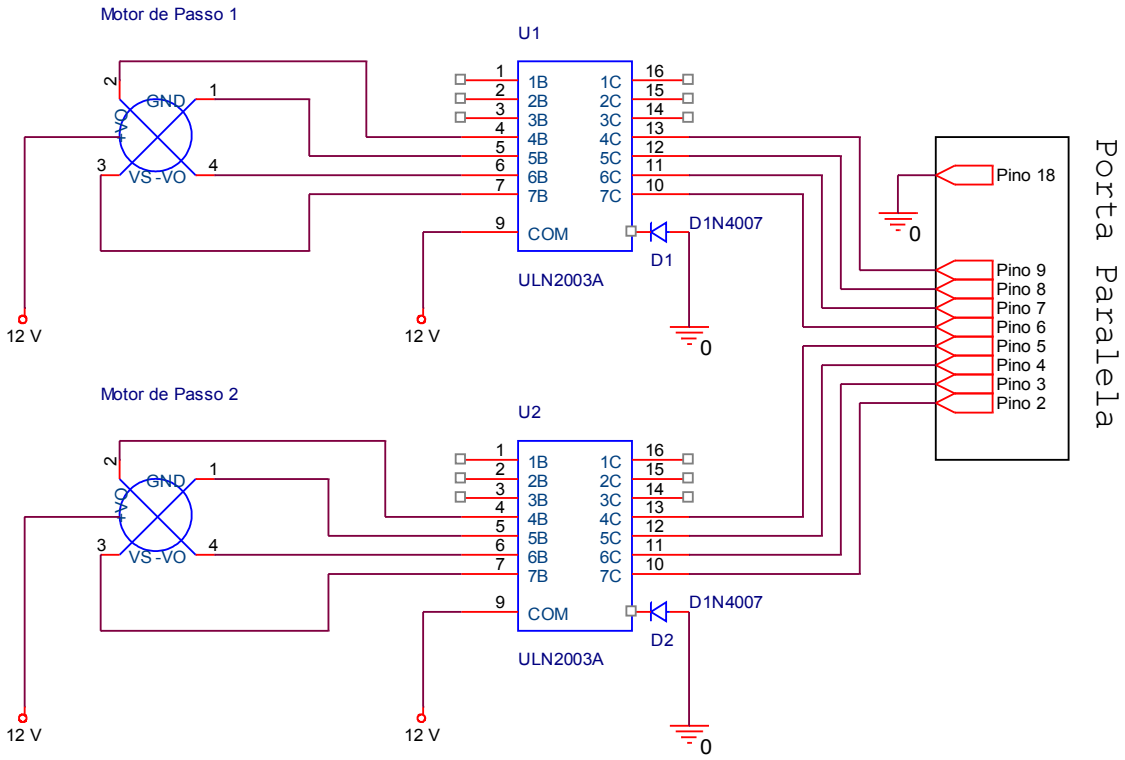

Figura A.2 – Circuito de controle do motor de passos