SISTEMA ROBUSTO PARA A ESTIMAÇÃO VISUAL DA POSIÇÃO DE SISTEMAS MÓVEIS A PARTIR DE MARCOS VISUAIS PLANARES

GUILHERME A. S. PEREIRA MARIO F. M. CAMPOS

VERLab – Laboratório de Visão Computacional e Robótica
Departamento de Ciência da Computação, Universidade Federal de Minas Gerais
Av. Antônio Carlos 6627, 31270-010, Belo Horizonte, MG, Brasil
E-mails: {gpereira, mario}@dcc.ufmg.br

Resumo— Este artigo descreve um sistema de estimação visual de posição baseado em modelos para aplicações de robótica móvel. O modelo utilizado é o de um marco visual planar com três objetos de características conhecidas. A utilização deste marco resulta em um problema conhecido na literatura como "problema dos três pontos perspectivos". São propostas técnicas eficazes para aquisição e processamento de imagens coloridas, bem como uma metodologia, baseada em métodos de otimização, para resolução do problema dos três pontos perspectivos em tempo real. Resultados experimentais mostram que o sistema é preciso e robusto e pode ser utilizado para prover a localização espacial de diversos sistemas móveis.

Abstract— This paper describes a robust vision based pose estimation technique for mobile robots based on image sequences of planar artificial landmarks. This problem reduces to the well known "three point perspective pose estimation problem". Robust techniques for color image acquisition and segmentation coupled with efficient statistical optimization methods are used to efficiently provide pose estimation in real time. Results of several experiments show that the methodology is both accurate and robust and can be used to perform the localization and control of several kinds of mobile robots.

Key Words— pose estimation, three point perspective problem, mobile robot localization

1 Introdução

A estimação da posição é uma operação muito importante em diversas tarefas visuais. Por este motivo, este tema e suas aplicações práticas são bastante estudados na literatura. Na área de robótica as aplicações estão principalmente relacionadas com a cooperação e controle de robôs móveis (Jung et al., 1998), navegação em ambientes desconhecidos (Tomono e Yuta, 2000) e localização (Briggs et al., 2000). O objetivo do trabalho é estimar, de forma rápida e robusta, a posição de um marco conhecido através de imagens. Como o problema de estimação já é bastante conhecido e documentado, o maior desafio é fornecer as informações de posição e orientação do marco de forma robusta e consistente com a maior taxa possível.

O presente trabalho é continuação do trabalho desenvolvido por Campos e de Souza Coelho (1999) onde os autores fazem o controle de posição de um dirigível autônomo baseado na informação visual fornecida por marcos conhecidos. Apesar dos bons resultados obtidos, o principal problema encontrado estava relacionado à capacidade de recuperação do sistema quando o marco saía do campo de visão da câmera. Outro trabalho relacionado é apresentado por Carceroni et al. (1998), onde ao invés de um marco planar, é utilizado um marco tridimensional que, apesar de ser de mais difícil construção, torna o problema mais simples.

O problema em questão é estimar a posição e orientação de um objeto de dimensões conhecidas a partir de sua imagem ou seqüência de imagens obtidas através de uma câmera com parâmetros (foco, resolução, dimensões do CCD e posição) bem conhecidos. Este problema é chamado na literatura de "problema dos n-pontos perspectivos" ou "estimação da posição baseada em modelos" e dependendo do número de pontos característicos do objeto uma única imagem é suficiente para solucioná-lo (DeMenthon e Davis, 1992; Haralick et al., 1991). O sistema de visão computacional descrito aqui tem como entrada imagens de um marco visual (*beacon*) planar que contém três pontos característicos como mostrado na Fig. 1. Este é o número mínimo de pontos necessário para realizar a estimação da posição tridimensional da câmera em relação ao objeto (ou *vice-versa*) a partir de uma imagem (DeMenthon e Davis, 1992). Para realizar esta estimação são necessárias as seguintes etapas básicas:

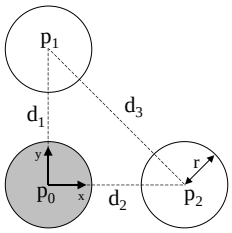

Figura 1. Marco visual planar.

1. Uma imagem do *beacon* deve ser capturada com uma câmera cujos parâmetros intrínsecos são conhecidos;
2. Devem-se determinar os *pixels* da imagem correspondentes aos pontos característicos deste marco;
3. A partir desta informação, calcula-se a posição e orientação do marco em relação à câmera e conseqüentemente a posição da câmera em relação ao marco.

2 Metodologia

A metodologia utilizada é baseada na proposta por Campos e de Souza Coelho (1999) mas com algumas modificações para melhorar o desempenho e a robustez da estimação. Em cada ciclo o

sistema faz a aquisição da imagem, localiza o *beacon* nesta imagem e calcula, a partir dos pontos característicos a posição da câmera em relação ao *beacon*. A próximas seções descrevem detalhadamente cada uma destas etapas.

2.1 Aquisição de Imagens

Em relação à aquisição de imagens, a principal diferença deste trabalho em relação aos demais (Campos e de Souza Coelho, 1999; Carceroni et al., 1998) é que aqui são utilizadas imagens coloridas que permitem a localização absoluta do *beacon* na imagem e evita o uso de técnicas de rastreamento que são pouco robustas à perda ou oclusão temporária do marco.

Para facilitar a segmentação das cores a placa de aquisição foi programada para adquirir imagens no espaço YUV ou YCbCr[a]. Neste espaço, Y é a componente de luminância e U ou Cb e V ou Cr são componentes de crominância. A princípio, as componentes de crominância são totalmente independentes da luminância e por isso são pouco sensíveis à variação de iluminação. Assim, com o uso deste espaço de cores o sistema é inerentemente mais robusto do que os que utilizam RGB. Para facilitar o processamento, cada componente da imagem original é considerada como uma imagem convencional em tons de cinza. Mais informações sobre os formatos mencionados podem ser obtidas em (Payton, 1996).

2.2 Processamento de Imagens

Uma vez capturadas as imagens, estas devem ser processadas e os pontos de interesse determinados. Ao contrário dos trabalhos anteriores, o uso de imagens coloridas permite que todo o processamento seja executado de forma absoluta, sem qualquer referência às imagens adquiridas anteriormente.

A etapa de **segmentação** é utilizada para separar os objetos de interesse da imagem. A princípio, como não se conhece a posição exata do marco, todos os objetos da imagem contendo as cores presentes no *beacon* devem ser segmentados. Assim, uma calibração deve ser executada *a priori* para se determinar as cores. Uma vez determinados os valores representativos das cores nas imagens de crominância, esta cor é separada das demais por uma etapa de **binarização**. Neste passo os *pixels* com tons de cinza similares ($\pm 10\%$) são ativados (255) e os demais são desativados (0). Esta etapa é executada para cada uma das cores presentes no *beacon* em todas as imagens de crominância. Para aumentar a robustez a ruídos, uma única imagem para cada cor é gerada através de um *and* lógico entre as imagens de crominância binarizadas.

Uma vez destacadas as cores de interesse, deve-se separar os objetos pertencentes ao marco de outros objetos e ruído. Para esta etapa é utilizada uma heurística que pressupõe que as cores estavam bem calibradas e portanto os objetos com maior área fazem parte do *beacon*. Como esta restrição é muito forte, pode-se utilizar, na heurística, outros conhecimentos sobre o marco, como por exemplo a sua dimensão física. Assim considera-se que os objetos do *beacon* devem estar próximos, onde a proximidade é definida em relação ao diâmetro dos círculos. O algoritmo utilizado é então da seguinte forma: os objetos da imagem são armazenados em um vetor de tamanho limitado (um para cada cor) em ordem decrescente de área; a partir dai são testadas todas as combinações possíveis dos objetos até que se encontre uma configuração que pode ser considerada um marco válido. Como o tamanho do vetor é fixo o tempo desta operação é limitado apesar de não ser fixo. Note também, que se o sistema de cores estiver bem calibrado os primeiros objetos dos vetores fazem parte do *beacon* e por isso, somente uma iteração é necessária para encontrar os objetos de interesse.

Para preencher os vetores de cores, é utilizado um algoritmo recursivo de **rotulação**. Este algoritmo inicializa uma pesquisa seqüencial na imagem a partir do primeiro *pixel*. Quando é encontrado um pixel ativado, é inicializada uma etapa recursiva em todos os seus vizinhos ativos. Este algoritmo calcula a área, o centro e as coordenadas máximas e mínimas do objeto e além disso, desativa todos os *pixels* do objeto para evitar que este seja visitado novamente. Como foi mencionado esta informação é guardada em um vetor na ordem decrescente de área. Este vetor é de tamanho limitado e por isso os menores objetos (geralmente ruído) não são armazenados. Ao fim da recursão, a busca seqüencial é reinicializada a partir do *pixel* inicial do objeto. Observe que terminada a busca, toda a imagem foi desativada. Note também, que o tempo de execução deste algoritmo é dependente do número e do tamanho dos objetos na imagem. Aparentemente este procedimento é mais eficiente que os tradicionais algoritmos de rotulação e caracterização de objetos (Ballard e Brown, 1982) pois toda a informação é obtida percorrendo-se a imagem uma única vez.

Ao fim da etapa de processamento da imagem tem-se um vetor ordenado pelo rótulo contendo os três objetos que formam a projeção do marco. Este vetor possuí além do centro do objeto, a sua área e as dimensões do retângulo em que o objeto melhor se adapta. Na próxima seção será mostrado como estas informações podem ser utilizadas para determinar a posição da câmera.

2.3 Metodologia Geométrica

Os pontos projetivos do *beacon* na imagem, determinados com a metodologia anterior, podem ser utilizados em uma metodologia geométrica para determinação da posição espacial dos pontos originais. Nesta etapa, o problema se resume ao "problema do três pontos perspectivos" (Haralick et al., 1991). Este problema é ilustrado na Fig. 2. Nesta figura, os parâmetros conhecidos *a priori*

[a] A rigor YUV é utilizado para imagens contínuas e YCbCr para imagens discretas. Neste texto não faz-se distinção entre estas nomenclaturas.

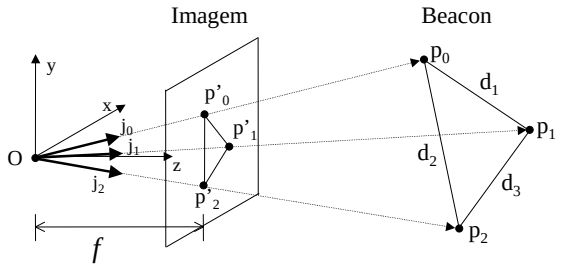

Figura 2. Problema dos 3 pontos perspectivos.

são as distâncias entre os pontos do *beacon* (d_1, d_2 e d_3) e as características da câmera (f e distância entre os *pixels*). Os parâmetros conhecidos após o processamento da imagem são os pontos projetivos do *beacon* (p'_0, p'_1 e p'_2) e conseqüentemente os vetores unitários $\vec{j_0}$, $\vec{j_1}$ e $\vec{j_2}$[b]. O problema é então, calcular a posição dos pontos do marco ou seja, os vetores que ligam a origem do sistema de coordenadas aos pontos P_0, P_1 e P_2. Estes vetores podem ser definidos respectivamente como:

$$\vec{P_0} = s_0\vec{j_0}\,, \quad \vec{P_1} = s_1\vec{j_1} \quad e \quad \vec{P_2} = s_2\vec{j_2}\,.$$

onde s_0, s_1 e s_2 são os módulos dos vetores e são as únicas incógnitas do problema. Assim, são necessárias três equações para a resolução do problema. Alguns autores preferem montar estas equações através da aplicação da lei dos cossenos ao diagrama da Fig. 2 (DeMenthon e Davis, 1992; Haralick et al., 1991). Neste trabalho entretanto, preferiu-se adotar a mesma técnica utilizada por Campos e de Souza Coelho (1999) para montagem das equações. Esta técnica baseada na subtração de vetores é mais intuitiva e mais fácil de ser visualizada. Assim, as três equações do problema são:

$$\begin{aligned} |\ s_0\vec{j_0} - s_1\vec{j_1}\ | &= d_1 \\ |\ s_0\vec{j_0} - s_2\vec{j_2}\ | &= d_2 \\ |\ s_1\vec{j_1} - s_2\vec{j_2}\ | &= d_3 \end{aligned}$$

Uma forma de resolver o sistema de equações anterior é parametriza-lo através de duas novas variáveis. Assim, considerando que $s_1 = u\,s_0$ e $s_2 = v\,s_0$ tem-se:

$$\begin{aligned} s_0\,|\ \ \vec{j_0} - u\vec{j_1}\ | &= d_1 \\ s_0\,|\ \ \vec{j_0} - v\vec{j_2}\ | &= d_2 \\ s_0\,|\ u\vec{j_1} - v\vec{j_2}\ | &= d_3 \end{aligned}$$

Isolando-se s_0 e expandindo os vetores unitários em suas coordenadas (x, y, z), tem-se através de alguma manipulação algébrica, três novas equações em função de u e v:

$$s_0^2 = \frac{d_1^2}{(x_0 - u\,x_1)^2 + (y_0 - u\,y_1)^2 + (z_0 - u\,z_1)^2} \qquad (1)$$

$$s_0^2 = \frac{d_2^2}{(x_0 - v\,x_2)^2 + (y_0 - v\,y_2)^2 + (z_0 - v\,z_2)^2} \qquad (2)$$

$$s_0^2 = \frac{d_3^2}{(u\,x_1 - v\,x_2)^2 + (u\,y_1 - v\,y_2)^2 + (u\,z_1 - v\,z_2)^2} \qquad (3)$$

[b]As coordenadas x e y deste vetor são determinadas com o processamento de imagem e a coordenada z é dada pelo foco da câmera.

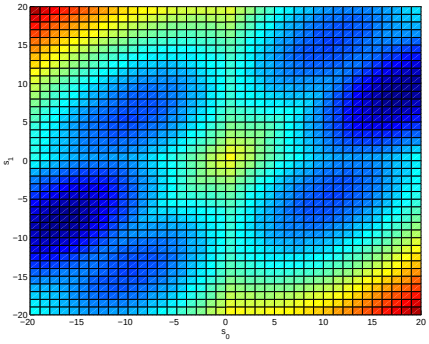

Figura 3. Função de similaridade em função de duas variáveis. Os pontos mais escuros mostram os mínimos.

Igualando-se as Eq. (1) e (2) e também (2) e (3) obtêm-se duas equações em u e v. Isolando uma destas variáveis em uma das equações e substituindo na outra, obtém-se uma equação de quarta ordem para uma destas variáveis. Esta metodologia e algumas variações da mesma são sugeridas por Haralick et al. (1991). A solução da equação de quarta ordem obtida produz uma solução analítica para o problema que a princípio possui 4 soluções. Além do problema de selecionar a resposta correta dentre aquelas disponíveis, Haralick et al. (1991) mostram, ainda, que a solução do problema leva a pontos de singularidade independentemente do método numérico utilizado.

Assim, para evitar os problemas provenientes da solução analítica, Campos e de Souza Coelho (1999) sugerem o uso de técnicas de otimização para minimizar uma função de similaridade entre o marco real e a sua projeção no plano de imagem da câmera. Esta função de similaridade é definida como:

$$S(s_0, s_1, s_2) = \sqrt{\sum_{i=0}^{1}\sum_{j=i+1}^{2}\left(|\ s_i\vec{j_i} - s_j\vec{j_j}\ | - |\ \vec{P_i} - \vec{P_j}\ |\right)^2} \qquad (4)$$

Desta forma, as variáveis escalares s_0, s_1, s_2 são determinadas como a solução do problema:

$$\min S(s_0, s_1, s_2)$$

Uma análise mais detalhada da Eq. (4) mostra que esta é uma função que possui mais de um mínimo global. Isto pode ser observado na Fig. 3, que é um gráfico da função de similaridade em função de duas variáveis para uma determinada posição da câmera em relação ao marco. Devido a esta característica multimodal da função de similaridade, torna-se necessário a utilização de um método de otimização global que encontra a região de ocorrência da solução de interesse, e um método local, responsável por encontrar a solução final do problema. Estas duas etapas de otimização serão mostradas a seguir.

Otimização Global

Nesta etapa o objetivo é encontrar um solução inicial para o processo de otimização local. A grande dificuldade é que, utilizando-se somente informação da projeção dos pontos característicos do

marco, pode-se obter soluções que a princípio não são válidas. Outras características do *beacon* devem então ser utilizadas.

A proposta de Campos e de Souza Coelho (1999) é utilizar a informação referente à área dos círculos do *beacon* de forma a escolher um valor inicial para a etapa de otimização local. Este método está baseado no fato de que a projeção perspectiva torna os círculos mais próximos à câmera maiores que aqueles que estão mais distantes. Desde que os três círculos usados no *beacon* triangular tenham o mesmo formato, o mesmo tamanho e sejam planares, pode-se mostrar através da lei dos inversos dos quadrados que:

$$\frac{|\vec{P}_i|}{|\vec{P}_j|} = \frac{s_i}{s_j} = \sqrt{\frac{A_j}{A_i}}\ ,\ 0 \leq i < j < 3$$

onde A_k, $0 \leq k < 3$ é a área em *pixels* na imagem do círculo P_k. Em particular, pode-se relacionar as áreas dos círculos com as variáveis u e v, definidas previamente, como:

$$u = \frac{s_1}{s_0} = \sqrt{\frac{A_0}{A_1}} \quad \text{e} \quad v = \frac{s_2}{s_0} = \sqrt{\frac{A_0}{A_2}}$$

Utilizando estes valores de u e v nas Equações 1, 2 e 3, pode-se escolher dentre os três valores de s_0 resultantes, aquele que produza o menor erro, ou seja o que produza o menor valor para a similaridade S, definida na Eq. (4). O valor escolhido pode, então, ser utilizado como valor inicial para o processo de otimização local. A grande vantagem desta metodologia, além do pequeno tempo de cálculo, é que possíveis problemas de ambigüidades de soluções são facilmente resolvidos. Além disso, se as áreas forem medidas com boa precisão a solução encontrada já é muito próxima da solução ótima exigindo menor esforço da otimização local. A grande desvantagem é que quando o *beacon* está longe da câmera, as projeções dos marcos têm praticamente a mesma área e isso pode conduzir a resultados errôneos. Além disso, para aplicar esta metodologia no caso de vértices de cores diferentes, como é o caso neste trabalho, deve-se garantir que a calibração das cores esteja satisfatória e todo o círculo esteja visível.

Considerando que o marco se desloca pouco de uma imagem para outra, pode-se evitar que a etapa de otimização global seja executada para todas imagens adquiridas, utilizando-se a solução obtida para a imagem anterior como entrada para a etapa de otimização local. Assim, esta etapa pode ser efetuada somente após a primeira visualização do *beacon*.

Otimização Local

A etapa de otimização local é responsável por refinar a solução encontrada pela etapa anterior. Campos e de Souza Coelho (1999) utilizam um método de busca local para melhorar o resultado obtido através da relação das áreas. Apesar de ser um método de otimização simples, que não depende de derivadas, este pode ser tornar lento se a solução inicial não estiver próxima à solução ótima. Neste trabalho é utilizado um método mais rápido de otimização. Este método, conhecido como método de Powell, é um método de direções conjugadas que também não depende de derivadas. A implementação completa do método pode ser encontrada em (Press et al., 1989).

2.4 Sistemas de Coordenadas

O algoritmo de otimização produz todos os resultados com relação ao sistema de coordenadas da câmera. Este sistema é uma base ortonormal que possui a origem no foco da câmera como é mostrado na Fig. 2. O eixo x é paralelo à largura do CCD, o eixo y é paralelo à sua altura e o eixo z coincide com o eixo principal da câmera (Fig. 2).

Por outro lado, o *beacon* também possui um sistema de coordenadas com a origem no ponto P_0, o vértice do ângulo reto do triângulo (ver Fig. 1). O eixo x é paralelo à linha que vai de P_0 a P_2 crescendo de P_0 a P_2 e o eixo y é paralelo à linha que vai de P_0 a P_1 na direção de P_1. Para formar uma base ortonormal o eixo z é perpendicular ao plano xy e saindo do papel.

Os sistemas de coordenadas da câmera e do *beacon* estão relacionados através de matrizes de transformações que podem ser definidas como uma rotação e uma translação (Craig, 1989). Assim, através de coordenadas homogêneas um ponto no sistema de coordenadas do *beacon* pode ser expresso no sistema de coordenadas da câmera como:

$$\begin{bmatrix} {}^cx \\ {}^cy \\ {}^cz \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & {}^cx_0 \\ r_{21} & r_{22} & r_{23} & {}^cy_0 \\ r_{31} & r_{32} & r_{33} & {}^cz_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^bx \\ {}^by \\ {}^bz \\ 1 \end{bmatrix} \qquad (5)$$

onde os parâmetros r_i são responsáveis pela rotação e os demais termos pela translação. A translação é representada por um vetor que descreve a origem do referencial do objeto em relação ao referencial da câmera, ou seja, o próprio valor de P_0 medido em relação ao referencial da câmera. Após a execução do algoritmo de otimização, tem-se a posição dos três pontos do *beacon* em relação ao sistema de referência da câmera. Como o *beacon* é um objeto conhecido, pode-se montar 9 equações para se determinar as 9 incógnitas na matriz da Eq. (5). Nesta etapa, como o *beacon* é planar tem-se problema durante a determinação dos valores de r_{13}, r_{23} e r_{33} relativos à coordenada z. Por isso, é necessário a criação de um novo ponto através de um produto vetorial de dois dos pontos conhecidos. Assim, o novo ponto em relação ao sistema do *beacon* é ${}^bP_3 = {}^bP_1 \times {}^bP_2$ e o seu correspondente no sistema da câmera é ${}^cP_3 = ({}^cP_1 - {}^cP_0) \times ({}^cP_2 - {}^cP_0)$. Resolvendo as equações tem-se então a seguinte matriz de transformação:

$$\begin{bmatrix} \frac{{}^cx_1 - {}^cx_0}{d1} & \frac{{}^cx_2 - {}^cx_0}{d2} & \frac{{}^cx_3 - {}^cx_0}{d3} & {}^cx_0 \\ \frac{{}^cy_1 - {}^cy_0}{d1} & \frac{{}^cy_2 - {}^cy_0}{d2} & \frac{{}^cy_3 - {}^cy_0}{d3} & {}^cy_0 \\ \frac{{}^cz_1 - {}^cz_0}{d1} & \frac{{}^cz_2 - {}^cz_0}{d2} & \frac{{}^cz_3 - {}^cz_0}{d3} & {}^cz_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (6)$$

Baseado na metodologia de ângulo fixos proposta em (Craig, 1989) é fácil obter a orientação do sistema de referência do *beacon* em relação ao sistema de referência da câmera. Assim, os ângulos

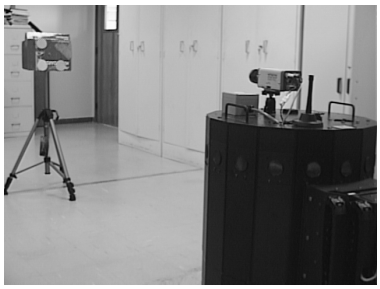

Figura 4. Robô Nomad 200 "olhando" para o *beacon*.

(*pitch*, *yaw* e *roll* respectivamente) que definem a rotação do *beacon* em relação à câmera são:

$$\beta = Atan2(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2})$$
$$\alpha = Atan2(r_{21}/\cos(\beta), r_{11}/\cos(\beta))$$
$$\gamma = Atan2(r_{32}/\cos(\beta), r_{33}/\cos(\beta))$$

onde $Atan2(x, y)$ calcula $\tan^{-1}(\frac{x}{y})$ mas usa os sinais de x e y para determinar o quadrante em que está o resultado. Se $\beta = \pm\pi/2$, $\cos(\beta) = 0$ e por convenção os demais ângulos são calculados como:

$$\alpha = 0.0 \quad \text{e} \quad \gamma = \pm Atan2(r_{12}, r_{22})$$

Em outras aplicações, principalmente no que diz respeito à localização de robôs móveis, pode ser necessário conhecer a localização da câmera em relação ao *beacon*. Neste caso a metodologia anterior se aplica deste que seja calculada a matriz de transformação inversa à mostrada na Eq. (6).

3 Experimentos

Para validar a metodologia proposta, utilizou-se um robô móvel Nomad 200 (Fig. 4). Este robô é equipado com uma câmera Hitachi colorida e uma placa Matrox Meteor para aquisição de imagens. O sistema é controlado por um computador Pentium Pro 200MHz com 96M *bytes* de memória RAM, rodando o sistema operacional Linux. Todos os algoritmos desenvolvidos foram escritos em C++ e compilados com o GCC.

As imagens da câmera são adquiridas através de um *device driver* especialmente desenvolvido para a placa Matrox. A imagem YUV adquirida pela placa é do tipo 4:2:2 o que significa que a componente Y tem resolução duas vezes maior que as demais. Além disso, as linhas pares e ímpares das componentes U e V são fornecidas separadamente.

O marco utilizado possui as seguintes características (ver Fig. 1): $d_1 = d_2 = 15,00\,\text{cm}$, $d_3 = 21,21\,\text{cm}$, $r = 6,00\,\text{cm}$. O foco da câmera, determinado por meio de calibração é de $f = 6,7\,\text{mm}$. A cor de P_0 foi escolhida como sendo vermelha e P_1 e P_2 foram representadas por círculos cor de rosa. Estas cores foram escolhidas no momento da calibração por se destacarem do ambiente quando visualizadas através de suas componentes de crominância. Após os testes constatou-se que cada ciclo do algoritmo que inclui desde a aquisição da imagem até o cálculo das componentes angulares e lineares da posição do *beacon* foi executado em um tempo médio de 100 ms ou seja 10 quadros/s.

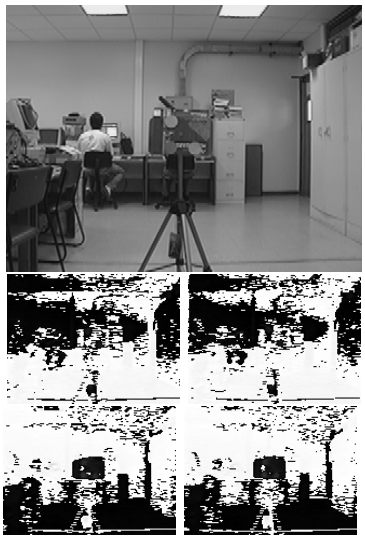

Figura 5. Componentes YUV da imagem do *beacon*. A imagem superior é a componente Y e as demais, da esquerda para a direita e de cima para baixo são respectivamente: U par, U ímpar, V par e V ímpar.

Figura 6. Imagens resultantes após à binarização.

Para testar o funcionamento do sistema, o robô foi colocado à frente do *beacon* e movimentado ao longo do eixo z da câmera (e consequentemente do *beacon*). Uma dessas imagens, mostrada nas suas componentes YUV, pode ser vista na Fig. 5. A imagem resultante, binarizada em torno dos limiares escolhidos para as cores do *beacon* é mostrada na Fig. 6. Note que, apesar de um ambiente visualmente confuso, não existe ruído em nenhuma das imagens já que as cores se destacam completamente. A Fig. 7 mostra as saídas y e z do sistema em função do tempo. Nesta figura, o robô começou a se movimentar em aproximadamente 6 segundos. Note que apesar da tentativa de mover o robô ao longo do eixo z, houveram variações (ainda que pequenas) na componente y. Essas variações podem ser explicadas pela vibração típica do robô durante o seu movimento.

Para verificar se os dados fornecidos pelo sistema estão corretos os resultados mostrados na Fig. 7 foram comparados com outro sistema de medição. Como todas as saídas do sistema estão relacionadas, optou-se por validar apenas a componente z, já que o robô provê uma maneira fácil de executar esta operação. Assim, a saída z do

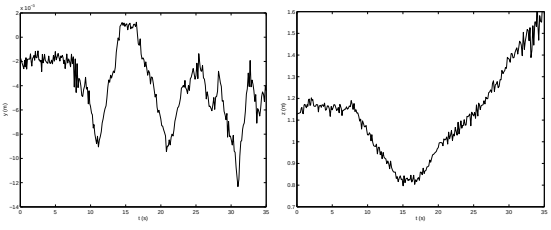

Figura 7. Saídas y e z do sistema.

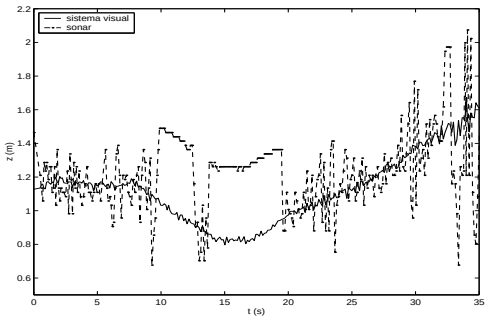

Figura 8. Comparação entre a saída do sensor de ultra-som do robô e a saída z do sistema visual.

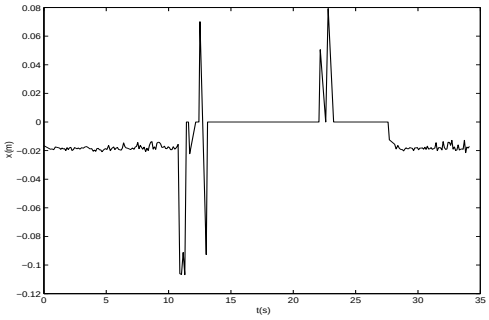

Figura 9. Comportamento do sistema em relação à perda e recuperação do *beacon*. Em aproximadamente 12 s o marco foi retirado e somente recolocado em 28 s.

sistema visual foi comparada com o valor fornecido pelo sensor de ultra-som do robô que está situado na mesma direção desta componente. Este resultado é mostrado na Fig. 8. Como a origem do sistema de coordenadas da câmera é diferente da localização do sensor o valor fornecido por este foi corrigido antes de ser executada a comparação. Observe que na média os dois sensores apresentaram o mesmo resultado apesar de o sistema visual ser mais preciso e apresentar menor variação.

Para verificação da precisão estática do sistema o robô foi posicionado a $1,2\,\mathrm{m}$ do marco. Nesta situação os desvios padrões para as variáveis de saída do sistema foram: $\sigma_x = 2,0\,\mathrm{mm}$, $\sigma_y = 0,3\,\mathrm{mm}$, $\sigma_z = 20,0\,\mathrm{mm}$, $\sigma_\beta = 1,94\,\mathrm{graus}$, $\sigma_\alpha = 1,15\,\mathrm{graus}$ e $\sigma_\gamma = 2,23\,\mathrm{graus}$. Como comparação, para as mesmas condições, o desvio padrão do sensor de ultra-som foi de $\sigma_{us} = 76,0\,\mathrm{mm}$.

Para se avaliar a robustez foram realizados experimentos para verificar o comportamento do sistema em relação à ausência de um marco válido. Para tanto, o robô foi mantido estacionário a $1,5$ metros do *beacon* e em certo momento este foi retirado do seu campo de visão e após algum tempo recolocado. O sistema foi programado para fornecer todas as suas saídas iguais à zero quando não fosse possível identificar o *beacon*. A Fig. 9 mostra o comportamento da saída x do sistema para esta situação. Algumas variações nesta saída antes da detecção da ausência do *beacon*, mostram que o comportamento do sistema pode ser melhorado nesses casos. Uma pequena variação em aproximadamente 23 s indica que em certos momentos outros objetos do ambiente foram considerados *beacons* o que também não é desejável.

4 Conclusão

O trabalho mostrou a implementação de um sistema para estimação visual da posição de uma câmera em relação a um marco planar. Os resultados práticos mostraram que o sistema é preciso e robusto apesar de possuir pequenos problemas. O uso de imagens coloridas permitiu que o sistema extraísse do ambiente em situações aparentemente muito difíceis, os pontos característicos do marco. Entre os problemas do sistema, o seu tempo de resposta pode limitar a gama de aplicações.

Agradecimentos

Os autores agradecem aos colegas do VERLab por suas valiosas opiniões e ao CNPq pelo suporte financeiro através dos projetos de pesquisa 140600/00-0 e 300212/99-2.

Referências

Ballard, D. H. e Brown, C. M. (1982). *Computer Vision*, Prentice-Hall.

Briggs, A. J., Scharstein, D., Braziunas, D., Dima, C. e Wall, P. (2000). Mobile robot navigation using self-similar landmarks, *Proc. of Int. Conf. on Robotics and Automation*, pp. 313–320.

Campos, M. F. M. e de Souza Coelho, L. (1999). Autonomous dirigible navigation using visual tracking and pose estimation, *Proc. of Int. Conf. on Robotics and Automation*, pp. 2584–2589.

Carceroni, R. L., Harman, C., Eveland, C. K. e Brown, C. M. (1998). Real-time pose estimation and control for convoying applications, *The Confluence of Vision and Control*, number 237 in *LNCIS*, Springer-Verlag, pp. 230–243.

Craig, J. J. (1989). *Introduction to Robotics Mechanics and Control*, second edn, Addison-Wesley.

DeMenthon, D. e Davis, L. S. (1992). Exact an approximate solutions of perspective-three-point problem, *IEEE Trans. on Pattern Analysis and machine Intelligence* **14**(11): 1100–1105.

Haralick, R. M., nan Lee, C. e Ottenberg, K. (1991). Analysis and solutions of the three point perspective pose estimation problem, *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 592–598.

Jung, D., Heinzmann, J. e Zelinsky, A. (1998). Range and pose estimation for visual servoing of a mobile robot, *Proc. of Int, Conf. on Robotics and Automation*, pp. 638–643.

Payton, C. A. (1996). *A Technical Introduction to Digital Video*, Jonh Wiley & Sons.

Press, W. H., Flannery, B. P., Teukolsky, S. A. e Vetterling, W. T. (1989). *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press.

Tomono, M. e Yuta, S. (2000). Mobile robot navigation in indoor environments using object and character recognition, *Proc. of Int. Conf. on Robotics and Automation*, pp. 313–320.