# *Navegação Visual de Robôs Móveis Autônomos Baseada em Métodos de Correlação de Imagens*

Leandro Facchinetti
Inst. de Ciências Matemáticas e de Computação - ICMC
Universidade de São Paulo - USP
São Carlos, Brasil
e-mail: leandro@grad.icmc.usp.br

Fernando Santos Osório
Inst. de Ciências Matemáticas e de Computação - ICMC
Universidade de São Paulo - USP
São Carlos, Brasil
e-mail: fosorio@icmc.usp.br

***Resumo* - Uma área importante de pesquisas na robótica é a navegação autônoma de veículos, onde o objetivo principal é fazer com que um robô móvel ou veículo seja capaz de se locomover sem a necessidade de alguém controlá-lo. Uma das abordagens que se destaca consiste em equipar o veículo com câmeras e a partir do processamento das imagens geradas orientar e controlar sua movimentação. Com este propósito, o presente trabalho propõe mecanismos para encontrar correlações entre imagens que ajudem a ajustar a rota (posição e orientação) de um veículo autônomo. A partir de uma região de interesse escolhida em uma imagem previamente armazenada do caminho a ser percorrido, o programa deve procurar uma área correspondente desta cena em outra imagem capturada em tempo real pelo veículo. A correlação entre as imagens previamente registradas do caminho e as imagens observadas pelo veículo, permitem que se determine o ajuste de sua orientação e o controle de seu avanço em uma determinada direção. Neste trabalho são propostos dois algoritmos de correlação que foram desenvolvidos e testados visando este tipo de aplicação, tendo seus desempenhos comparados. Buscou-se otimizar o desempenho destes algoritmos a fim de viabilizar a sua utilização em aplicações de controle de veículos móveis em tempo real.**

***Robótica Autônoma, Navegação Visual, Visão Computacional, Correlação de Imagens, Navegação por Pontos de Referência, Sistemas Robóticos Inteligentes***

## I. INTRODUÇÃO

A navegação autônoma de veículos é um tópico importante da área de pesquisas de robótica [3, 9, 10]. Existe grande variedade de aplicações para veículos inteligentes e autônomos [11]. Alguns exemplos são: auxiliar o motorista a conduzir nas estradas (detecção e desvio de obstáculos); estacionar autonomamente; evitar que o veículo saia da pista; condução não assistida de cargas em depósitos; entregas em escritórios; comboios com múltiplos veículos seguindo um líder; vigilância automatizada, entre outras aplicações. Essa é a principal motivação para o estudo de robôs móveis e de navegação autônoma para veículos, área de pesquisa do Projeto NAVIS: Veículos Autônomos Inteligentes com Sistema de Navegação baseado em Visão Computacional [3], que vem sendo desenvolvido pelo grupo de pesquisa envolvido com este trabalho.

No contexto deste projeto de pesquisa se enquadra a proposta do desenvolvimento do Sistema CIRMA, relativo à implementação de métodos baseados em Correlação de Imagens aplicados a navegação visual de Robôs Móveis Autônomos.

Tarefas simples, como o problema enfrentado de conduzir um robô móvel sem a assistência humana a atravessar uma região de acesso mais difícil (atravessar corredores estreitos, passar por uma porta), podem ser exemplos de desafios enfrentados pela robótica. Sensores de proximidade/distância podem sofrer de imprecisão e de falta de sensibilidade quando são usados para ajudar a atravessar passagens muito estreitas. A solução proposta e estudada neste trabalho é baseada no uso de uma câmera, visando guiar o veículo através de imagens, o que é conhecido como visão computacional e navegação visual.

Este trabalho tem por objetivo desenvolver algoritmos de navegação visual [3, 4, 5, 11] para tornar veículos inteligentes, capazes de guiarem a si mesmos. A abordagem difere da tradicional modelagem do ambiente através de mapas em 2D ou 3D (e.g. grades de ocupação, mapas geométricos), com a aplicação de algoritmos de auto-localização, planejamento e execução de trajetórias [9,10]. Busca-se trabalhar apenas com as imagens capturadas do trajeto a ser seguido pelo robô, através de uma abordagem denominada de controle servo-visual (*visual servoing*) [12].

Portanto, para que se possa conduzir um veículo autônomo baseado apenas em imagens, não será necessária a determinação explícita de sua posição e de seu destino de acordo com um mapa pré-definido. Por outro lado, é necessário que sejam encontrados pontos ou elementos de referência nas imagens (e.g. *landmarks*), permitindo assim que o robô possa se orientar através destes e se deslocar em direção ao seu destino que define um "alvo visual".

A inspiração para este tipo de técnica vem da forma com que os seres humanos usualmente utilizam a visão para se orientar: não precisamos de nossas coordenadas GPS precisamente definidas, ou de mapas precisos descrevendo o ambiente, para que possamos assim nos locomover em lugares previamente conhecidos. Utilizamos principalmente nosso sistema de visão, considerando pontos de referência no ambiente que nos orientam e nos dão condições de determinar e ajustar nossa rota.

Na Navegação Visual, Pontos de Interesse são selecionados previamente junto à "memória do caminho" a ser percorrido, e depois são procurados na imagem capturada pelo robô, através de diferentes técnicas, onde uma técnica que se destaca é a chamada de Correlação de Imagens [4]. Esta técnica é ilustrada através da Figura 1 [5].

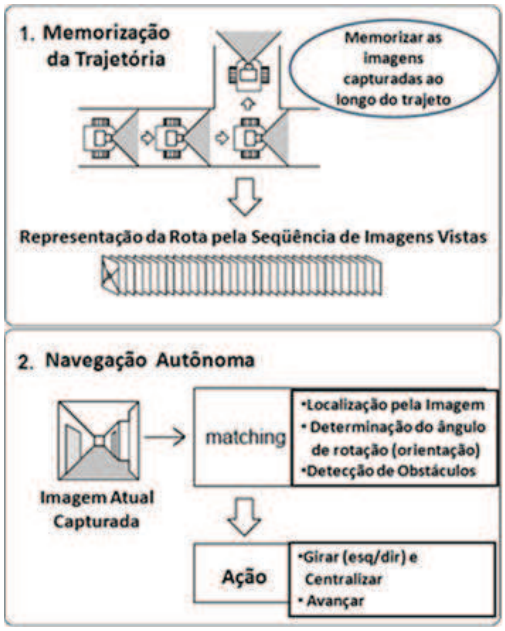


Figura 1. Navegação de Robôs Móveis baseada em imagens (adaptada de [5])

O uso de algoritmos de correlação desenvolvidos no Sistema CIRMA funciona da seguinte forma: um robô é inicialmente conduzido uma vez por uma rota, e depois ele será capaz de reproduzi-la, seguindo o caminho "aprendido" (imagens memorizadas do caminho), sem auxílio humano. Enquanto percorre o trajeto pela primeira vez, ele identificará pontos de interesse capazes de guiá-lo quando reconhecer novamente estes pontos, nas próximas navegações. O robô irá ajustar sua trajetória considerando como referências os pontos de interesse memorizados. Os algoritmos desenvolvidos no CIRMA servem para identificar o posicionamento dos pontos de referência nas imagens que o robô está vendo, a partir da correlação com as imagens geradas antes.

A principal contribuição deste trabalho está no uso e otimização de algoritmos robustos de correlação de imagens. Em trabalhos anteriores [4, 5] predomina o uso de técnicas de correlação baseada no método NCC (*Normalized Cross-Correlation*) sobre imagens monocromáticas, tendo sido investigadas também por Righes [13, 14] em imagens coloridas. Esta técnica, o NCC, é bastante custosa e de difícil aplicação em aplicações com restrições de tempo real. Neste contexto, este trabalho se propõe a utilizar a correlação de histogramas para a navegação visual, uma técnica que possui vantagens (maior invariância a posição, escala e rotação), porém com um custo computacional superior ao do NCC. Portanto, a correlação de histogramas precisa ser otimizada, para poder ser usada em aplicações robóticas de tempo real. Em função disto é proposto neste artigo o uso de um algoritmo piramidal que permite aumentar significativamente o desempenho dos algoritmos de correlação baseados em histograma, como o proposto neste trabalho.

As próximas seções deste trabalho estão organizadas da seguinte forma: a seção II contém um estudo de trabalhos relacionados; a seção III apresenta as metodologias empregadas nos algoritmos desenvolvidos no CIRMA; a seção IV trata dos resultados obtidos nos testes efetuados com o programa; por fim, a seção V apresenta a conclusão e a proposta de trabalhos futuros.

## II. TRABALHOS RELACIONADOS

Um dos primeiros trabalhos clássicos baseados em visão computacional que visavam implementar a navegação por imagens foi proposto por Jones et al. [4] e por Matsumoto et al. [5]. Estes trabalhos surgiram estudando navegação autônoma em ambientes internos, baseados em imagens monocromáticas em escala de cinza (*grayscale*). Estes trabalhos demonstraram a viabilidade de se realizar a navegação baseada em imagens, onde um robô foi capaz de seguir uma trajetória definida através de imagens do caminho a ser percorrido.

Esta abordagem permite que o robô faça constantes ajustes de seu direcionamento, melhor se orientando em relação a um alvo, como por exemplo, passar por uma porta muito estreita. Mesmo se os sensores de colisão forem imprecisos, não permitindo um ajuste correto do posicionamento do robô (saturação dos sensores), para que este possa ajustar sua trajetória adequadamente em relação ao alvo, com a ajuda do sistema de visão será possível então alinhar precisamente a trajetória do robô e passar através deste obstáculo.

Os trabalhos de [4, 5] são baseados em técnicas de correlação de imagens como o NCC (*Normalized Cross-Correlation*), utilizando imagens monocromáticas e executando algoritmos computacionalmente custosos. Righes [13, 14] propôs a aplicação do NCC em imagens coloridas, entretanto constatou-se que o custo computacional tornava-se ainda mais proibitivo, e onde o ganho em termos de robustez/precisão da correlação das imagens coloridas não justificava o aumento da carga de processamento necessária para tratar este tipo de imagens.

Em Matsumoto et al. [15] é proposto o uso de uma câmera omnidirecional, de modo que se possa ter uma visão de 360 graus ao redor do robô e com isto, podemos definir um caminho de ida e volta baseado em apenas uma seqüência de imagens. Neste trabalho a resolução das imagens adquiridas era muito grande, o que causava problemas de performance e obrigou os autores a re-escalar as imagens para um tamanho inferior (com menos resolução e por conseqüência menos detalhes e precisão).

Estes trabalhos demonstraram a viabilidade do uso da correlação de imagens para a navegação visual, entretanto fica claro que existe ainda um problema relativo à melhoria da performance de tais algoritmos visando sua aplicação em problemas de robótica em tempo real.

## III. METODOLOGIA

Quando o robô faz pela primeira vez o caminho a ser percorrido, ele é guiado por um humano, pois ainda está aprendendo. Nessa etapa, sua função é buscar nas imagens que a câmera faz do ambiente os pontos de referência que serão usados depois para a correlação. Exemplos de boas escolhas, em geral, são elementos que se destacam nas imagens de um modo geral, como placas, marcações e objetos que sejam facilmente reconhecíveis em uma próxima visita a este mesmo local. A idéia é bastante semelhante ao processo que os humanos usam para se guiar pela memória visual de um caminho previamente trilhado.

Neste trabalho, foi considerado que um usuário irá identificar estes pontos de referência e marcar (manualmente) as regiões de interesse nas imagens que foram capturadas do caminho trilhado. Não será foco deste trabalho a identificação automática das regiões de interesse, que pode vir a ser automatizada, mas sim o tratamento da correlação entre a região de interesse selecionada, buscada em outra imagem "similar" (adquirida em uma posição e orientação próximas a daquela em que esta imagem original foi capturada).

Portanto, os algoritmos de correlação criados trabalham com um par de imagens. Uma delas, a imagem original, tem uma região de interesse escolhida com base nos pontos de referência determinados na primeira passagem pelo caminho. Na outra imagem, obtida pelo robô enquanto navega procurando ajustar a sua rota ao caminho previamente percorrido, o programa deve determinar a localização desta região de interesse. Através da identificação da região de interesse (pontos de referência) em ambas as imagens, podemos determinar se o robô deve girar para a esquerda ou direita a fim de melhor ajustar seu direcionamento. Uma vez ajustado o direcionamento, o robô pode avançar, verificando se ele já alcançou a imagem seguinte, de acordo com o apresentado na Figura 1.

A ferramenta escolhida para o desenvolvimento do sistema foi o OpenCV [6], uma biblioteca de código livre que teve origem em trabalhos feitos pela Intel. Ele contém funcionalidades de manipulação e tratamento de imagens, úteis em visão computacional. É escrito em C (linguagem adotada no projeto) e contém módulos em Python, que não chegaram a ser usados. A razão da escolha desta ferramenta foi a de se aproveitar as estruturas de dados e funções básicas de manipulação de imagens providas por este pacote, facilitando o desenvolvimento do sistema proposto.

O OpenCV vem com estruturas para armazenar imagens e histogramas, como por exemplo: funções para transformar imagens coloridas em tons de cinza; redimensionar imagens; aplicar diversos filtros; gerar histogramas; etc. Essas funcionalidades foram necessárias a pesquisa na parte referente a correlação, histogramas e processamento piramidal de imagens.

Os algoritmos desenvolvidos baseiam-se na análise de histogramas, que são uma representação da distribuição de cores em uma imagem. Ele é feito dividindo o espectro de cores possíveis de serem representadas em intervalos, então são contados quantos pontos estão presentes em cada partição (freqüência de ocorrência de cada cor/partição). Essa é uma forma bastante funcional de interpretar uma imagem, visto que o histograma permite caracterizar uma região da imagem, sendo bastante robusto (invariante) a pequenas alterações na posição, orientação, e mesmo na escala, que possa vir a afetar uma determinada região da imagem. Em função destas propriedades dos histogramas optou-se por utilizar este tipo de atributo para descrever a região de interesse a ser buscada, resultando na implementação de um método baseado na correlação de histogramas.

Um aspecto muito importante dos histogramas, que também contribuiu na sua escolha para o estudo, foi seu comportamento quando ocorrem mudanças de luminosidade no ambiente. Se forem analisadas as cores, puramente, uma pequena variação no brilho causa grande diferença nas cores de uma imagem. É uma situação bastante comum nas imagens do mundo real em que se pretende trabalhar na aplicação final. Os histogramas, por sua vez, diferem apenas no brilho do conjunto de cores, mas mantém o "formato" deste histograma, o que facilita a correlação. Isso os torna menos sensíveis ao descontrole da intensidade da iluminação do ambiente, que é uma característica bastante desejável.

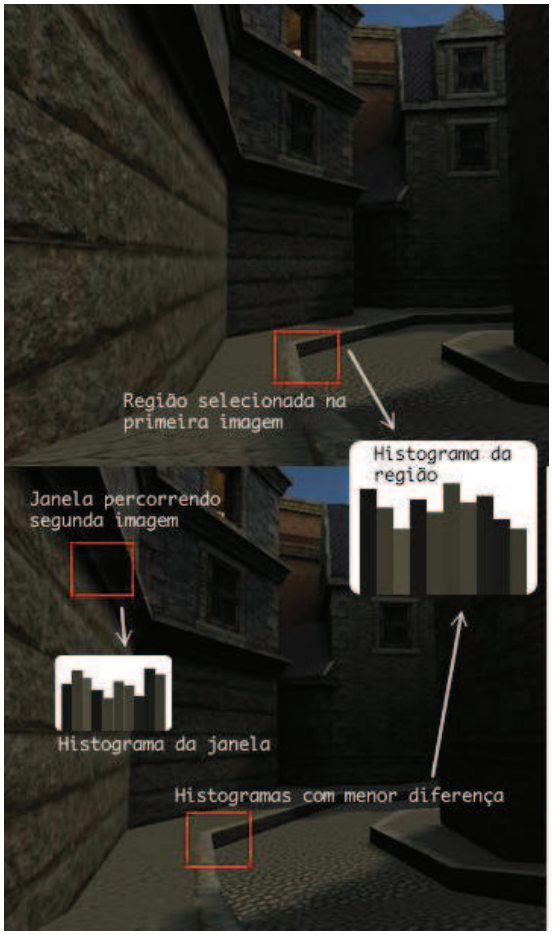


Figura 2. Correlação de duas Imagens distintas baseada em Histogramas das Regiões de Interesse (Algoritmo de correlação simples)

Em uma navegação feita por robôs, é esperado que a posição que ele ocupe a cada nova viagem seja ligeiramente diferente, causando uma pequena mudança na posição, orientação e escala da imagem capturada pela câmera (devido a projeção 3D=>2D), bem como as condições de iluminação local também podem variar. Por isso é importante que os algoritmos de correlação sejam robustos em relação a estas perturbações.

### *A. Correlação Simples*

O primeiro algoritmo desenvolvido foi chamado de algoritmo de correlação simples. Ele funciona calculando um histograma da região selecionada em uma das imagens. Então, começa a busca pela região correspondente na outra imagem. Uma janela deslizante do tamanho da seleção percorre pixel a pixel a segunda imagem, e para cada posição é calculado o histograma desta área. Em seguida é feita a comparação (correlação) entre este histograma e aquele da imagem original, onde a região com a menor diferença é a região escolhida como correspondente. O algoritmo é detalhado a seguir:

1. Calcular o histograma da região selecionada na imagem original
2. Criar uma janela com o tamanho da região selecionada na imagem original e na imagem final, posicionada no canto superior esquerdo
3. Deslocar a janela, pixel a pixel, por toda a imagem final. Para cada passo:
   3.1. Calcular o histograma da região na qual a janela se encontra
   3.2. Calcular a diferença entre o histograma da região selecionada na imagem original, calculada no passo 1, e o da região no qual a janela se encontra, resultado do passo 3.1
   3.3. Se a diferença for menor do que as outras encontradas até então, guardar
4. Retornar a janela guardada no passo 3.3.

O histograma é obtido usando as funções internas do OpenCV, em particular a cvCalcHist(), e a diferença entre histogramas é calculada usando a implementação do algoritmo denominado de Intersecção de Histogramas [7], que está disponível na função cvCompareHist() da biblioteca. Este processo é ilustrado na Figura 2.

### *B. Correlação Piramidal*

O segundo algoritmo é um aperfeiçoamento do algoritmo de correlação simples, através do uso de um algoritmo piramidal. Nele, as imagens são reduzidas sucessivas vezes, formando uma pirâmide Gaussiana de imagens [1, 2]. Por isso ele é chamado de algoritmo de correlação piramidal. Para gerar a pirâmide, é usada a função cvPyrDown(), do OpenCV. A idéia de janela deslizante de cálculo de histogramas é aplicada na menor imagem e a área em que ocorre a maior correlação é projetada sobre a imagem que está no nível anterior. Sobre esta projeção é dada uma margem de erro, o que determina uma região na imagem deste nível, e nela passa outra janela deslizante. A melhor correlação é projetada na imagem seguinte, e assim por diante. A Figura 3 ilustra este processo, que é descrito no algoritmo a seguir:

1. Se os parâmetros de entrada determinam que deva ocorrer um novo nível na pirâmide:
   1.1. Reduzir as imagens de entrada, usando o método da pirâmide Gaussiana
   1.2. Chamar recursivamente este algoritmo com as imagens geradas no passo 1.1 e um nível a menos, na pirâmide
   1.3. Restringir a janela de busca na imagem destino como sendo a janela resultado do passo 1.2 acrescida de uma margem de erro, para cada lado
2. Retornar a janela gerada pelo algoritmo de correlação simples nas imagens com as regiões definidas.

A intenção desse algoritmo é se beneficiar do fato de que em imagens menores (*zoom out*) é preciso calcular menos histogramas para encontrar uma boa correlação. A informação obtida em uma etapa guia a escolha da região a ser pesquisada na etapa seguinte, de forma que a janela deslizante não precisa percorrer a imagem toda, pois já é conhecida a posição aproximada da maior correspondência. Executar a janela deslizante na imagem maior servirá para aumentar a precisão da resposta, porque com o resultado da etapa anterior obtemos apenas uma estimativa de onde aproximadamente se encontra a correlação buscada.

Tanto a quantidade de imagens na pirâmide quanto o tamanho da margem de erro afetam a qualidade da resposta e o tempo de execução do algoritmo. Uma margem de erro grande faz com que mais histogramas sejam calculados, por outro lado se ela for pequena, o algoritmo sofrerá de imprecisão. Se forem feitas etapas demais na pirâmide, a busca inicial da correlação se dará em imagens bastante reduzidas, nas quais as informações sobre os objetos de interesse podem ter se perdido, produzindo resultados piores. Poucos níveis na pirâmide provocam buscas iniciais em imagens maiores, o que é mais custoso. Fica a cargo do usuário do programa determinar os parâmetros apropriados para seu conjunto de imagens.

Outra característica que afeta o desempenho de ambos os algoritmos é o tamanho da imagem na qual a busca é feita. Quanto maior for essa figura, mais cálculos devem ser feitos, e maior o tempo de execução.

Há ainda, uma restrição nas imagens usadas no algoritmo piramidal: elas devem ter dimensões múltiplas de $2^n$ onde n é o número de etapas da pirâmide. Isso vem da forma com que é feita a pirâmide gaussiana junto ao OpenCV.

É possível interpretar o algoritmo de correlação simples como sendo uma particularização do piramidal, no qual a pirâmide tem apenas uma etapa. O que ocorre para cada nível do algoritmo piramidal é a criação de uma janela deslizante percorrendo a imagem final (reduzida), calculando histogramas e comparando com o histograma da imagem original, buscando pela menor diferença. Isso é exatamente o que o algoritmo de correlação simples faz, logo, cada iteração do algoritmo piramidal é uma execução do simples.

Para estudar diferentes variações de entrada, o sistema suporta imagens coloridas e também em tons de cinza. A diferença está na quantidade de canais: enquanto a escala de cinzas usa apenas um canal (1 byte por pixel), as imagens coloridas precisam de três (3 bytes por pixel). No sistema CIRMA, as imagens coloridas fazem uso do sistema de cores RGB.

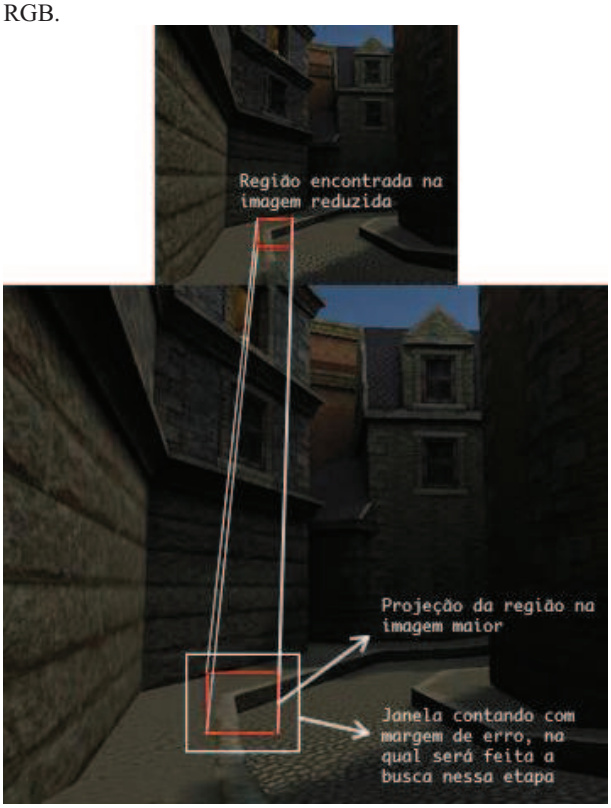


Figura 3. Uma etapa do algoritmo piramidal

Sendo assim, calcular histogramas em três canais é mais custoso do que em apenas um, necessitando de mais cálculos. Por outro lado, imagens coloridas são mais ricas em informações. Foram feitos estudos se elas tornam a correlação mais precisa e qual o sistema de cores (1 ou 3 canais) tem o melhor custo/benefício considerando desempenho e qualidade da resposta. Os resultados destes estudos são descritos na seção seguinte.

## IV. Experimentos e Resultados Obtidos

O programa desenvolvido com esses algoritmos já está operacional, sendo capaz de identificar correlações entre as regiões de duas imagens, conforme foi apresentado no exemplo da Figura 2 (estas imagens foram geradas a partir de testes com o programa implementado[1]). Ele é capaz de trabalhar com imagens tanto em tons de cinza quanto com imagens coloridas. Apesar de ser mais custoso calcular o histograma em imagens com cores (RGB), há um ganho, pois aumenta a quantidade de informações coletadas que podem contribuir para um melhor casamento das imagens. O programa também aceita um conjunto de parâmetros na linha de comando que torna possível automatizar os testes. Além disso, está integrado nele um conjunto de operações para medir o desempenho de sua execução, de modo a avaliar os algoritmos em diferentes situações.

A qualidade das soluções é complicada de ser medida. Como as imagens não são exatamente iguais, não existe uma correlação completamente correta e, portanto, a resposta obtida é intrinsecamente imprecisa. Então o critério foi através da realização de uma análise visual por um especialista no assunto, onde o resultado foi considerado satisfatório se indicar uma região equivalente da imagem original, e se esta for boa o suficiente para prover a informação que fará funcionar o algoritmo de navegação visual do robô.

Um teste adicional, feito para averiguar a qualidade das respostas, foi rodar o algoritmo usando a mesma imagem como origem e destino, mas não informá-lo de que estava operando nessas condições. Nesse caso, a resposta correta está bem definida: o quadro de correlação é igual ao quadro na imagem original. Ambos os algoritmos acertaram em 100% dos casos, nesse tipo de teste, independente da quantidade de canais.

Os testes iniciais foram realizados com uma webcam, onde constatou-se que o sistema dava bons resultados, no entanto as imagens deste tipo de câmera possuem muito ruído, sendo necessário o uso de um sistema de aquisição de imagens de melhor qualidade. Além disto, para que pudessem ser realizados testes mais precisos e repetitivos com o sistema integrando processamento de imagens e movimentação de um robô, foi adotado um ambiente virtual realístico com uma câmera virtual capaz de se deslocar neste ambiente.

Deste modo, foram adotadas como caso de testes imagens geradas sinteticamente. Para isso foi usado um programa que cria visualizações realísticas de cenas em três dimensões. As cenas apresentadas nas Figuras 2 e 3 foram geradas com o auxílio deste programa.

TABELA I. Exemplo de tempos de diferentes execuções dos algoritmos de correlação

| **Imagens de 544 por 464 pixels e Área de busca de aproximadamente 60 por 60 pixels (resultados em segundos)** | |
|---|---|
| **Coloridas** | |
| ***Simples*** | ***Piramidal*** |
| 53.439000 | 0.967000 |
| 47.362000 | 1.061000 |
| 51.667000 | 1.077000 |
| 55.101000 | 1.076000 |
| **Tons de cinza** | |
| ***Simples*** | ***Piramidal*** |
| 29.063000 | 0.172000 |
| 16.786000 | 0.218000 |
| 11.482000 | 0.203000 |

[1] Vídeos de demonstração disponíveis em:
http://www.youtube.com/watch?v=qCoa8Rjzl80 ou http://bit.ly/a4c0LM
http://www.youtube.com/watch?v=mlwXKJZQkk4 ou http://bit.ly/a7HLLA

Os resultados obtidos nos testes descritos na Tabela I mostram que o algoritmo de correlação simples leva em média 50 segundos para tratar imagens coloridas de uma resolução de aproximadamente 550x450. Usar somente um canal faz com que o tempo caia para perto da metade. A qualidade da resposta é satisfatória e não parece ser afetada pela escolha da quantidade de canais.

O algoritmo de correlação piramidal, tendo como entrada as mesmas imagens, coloridas, responde em um segundo, em média. Isto é, o algoritmo piramidal é 50 vezes mais rápido que o simples, nesse caso. Em relação à qualidade do resultado, esta não é afetada perceptivelmente. No entanto, usar o algoritmo piramidal com imagens sem cores produz resultados relativamente ruins. Desprezar a informação adicional trazida pelas cores parece desorientar o algoritmo piramidal em certas situações, apesar de melhorar significativamente o desempenho, para um quinto de segundo, em média.

## V. Conclusões e Trabalhos Futuros

A partir dos testes realizados, conclui-se que a melhor combinação tratando de custo/benefício é o algoritmo piramidal com imagens coloridas. Com ela obteve-se o resultado próximo de um segundo, bom o suficiente para aplicações do tipo *soft real-time*. Isto é, permite prover informações ao robô para que ele possa se mover adequadamente em tempo real, mas sem o compromisso de entregar respostas num intervalo fixo e muito curto, como podem exigir certas aplicações de tempo real com restrições mais rígidas. Em geral, usar um quadro por segundo é o suficiente, visto que a movimentação do robô considerado não ocorre em velocidades muito altas. Assim, os pontos de referência não terão mudado muito de uma imagem para outra.

Um uso do algoritmo de correlação desenvolvido e que se pretende estudar em breve é com imagens estéreo, que contêm informações de profundidade. São feitas com duas câmeras posicionadas uma ao lado da outra, onde nessas imagens podem ser extraídos dados sobre a profundidade dos objetos na cena. O uso da correlação com estas imagens pode trazer resultados possivelmente interessantes para a navegação autônoma.

Além disso, pretende-se desenvolver testes com o algoritmo desenvolvido aplicado diretamente sobre um robô móvel. Usando as informações de correlação obtidas pelo algoritmo de correlação, este sistema deverá ser capaz de orientar e controlar um veículo em sua rota.

É possível, ainda, melhorar o desempenho do algoritmo de correlação usando uma otimização em hardware. Existem trabalhos em desenvolvimento visando a implementação de algoritmos de correlação em hardware reconfigurável (FPGA), o que irá permitir uma execução otimizada do algoritmo. Este hardware pode ser projetado para calcular mais rápido os histogramas e/ou a correlação, chegando ainda mais perto do objetivo de navegação em tempo real.

## Agradecimentos

Os autores gostariam de agradecer ao apoio do CNPq e da FAPESP através do INCT-SEC (Instituto Nacional de C&T em Sistemas Embarcados Críticos), processos 573963/2008-9 e 08/57870-9. Além disto, agradecemos em particular ao CNPq pelo financiamento de parte deste projeto através do Edital Universal 2008 e da concessão de uma bolsa do programa PIBIC/CNPq.

## Referências

[1] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden, “Pyramid methods in image processing,” RCA Engineer 29 (1984): 33–41.

[2] A. Rosenfeld, “Some Uses of Pyramids in Image Processing and Segmentation,” Proceedings of the DARPA Imaging Understanding Workshop. pp. 112–120, 1980.

[3] OSÓRIO, Fernando. Projeto NAVIS: Veículos Autônomos Inteligentes com Sistema de Navegação baseada em Visão Computacional. Projeto - Edital Universal – CNPq 2008.

[4] JONES, S. D., ANDERSEN, C. S., and CROWLEY, J. L. Appearance based processes for visual navigation. In: IROS’97 – Proceedings of the IEEE IROS - Intelligent Robots and Systems Conference, Vol. 2, 1997. p. 551-557.

[5] MATSUMOTO, Y., INABA, M., and INOUE, H. Visual navigation using view-sequenced route representation. In Proceedings of the IEEE ICRA - International Conf. on Robotics and Automation, 1996. pp. 83-88, Minneapolis, Minnesota.

[6] Manual do OpenCV. 2008. p. 345-356.

[7] BRADSKI, Gary; KAEHLER, Adrian. Learning OpenCV: Computer Vision with the OpenCV Library. 2008. p. 193-221.

[8] WOLF, Denis F.; OSÓRIO, Fernando S.; SIMÕES, Eduardo; TRINDADE Jr., Onofre. Robótica Inteligente: Da Simulação às Aplicações no Mundo Real. [Tutorial] In: André Ponce de Leon F. de Carvalho; Tomasz Kowaltowski. (Org.). JAI: Jornada de Atualização em Informática da SBC. Rio de Janeiro: SBC - Editora da PUC Rio, 2009, v. 1, p. 279-330.

[9] DUDEK, G.; JENKIN, M. (2000). “Computational Principles of Mobile Robotics”. Cambridge, London, UK: The MIT Press, 280 p.

[10] SIEGWART, Roland and NOURBAKHSH, Illah R. (2004). “Introduction to Autonomous Mobile Robots”. A Bradford Book, The MIT Press: Cambridge, London. 317p.

[11] JUNG, C. R.; OSÓRIO, F. S.; KELBER, C.; HEINEN, F. (2005) “Computação embarcada: Projeto e implementação de veículos autônomos inteligentes”, In: Anais do CSBC’05 XXIV Jornada de Atualização em Informática (JAI). São Leopoldo, RS: SBC, v. 1, p. 1358–1406.

[12] Winters, N.; Gaspar, J.; Lacey, G.; Santos-Victor, J. Omni-directional vision for robot navigation. IEEE Workshop on Omnidirectional Vision, 2000 (Proceedings). pp.21-28.

[13] RIGHES, E.M.; (2004) "Processamento de Imagens para Navegação de Robôs Autônomos"; Relatório TCC - Informática; UNISINOS, São Leopoldo, RS. http://osorio.wait4.org/oldsite/alunos/tcc/righes-tc.pdf (Acesso em 08/02/2009).

[14] RIGHES, E.M.; OSÓRIO, F. S. Correlação de Imagens Coloridas Visando Auxiliar na Navegação e Controle de Robôs Autônomos. Anais do V ENIA – Congresso da SBC 2005, São Leopoldo – RS. pp. 1122-1125.

[15] Matsumoto, Yoshio; Ikeda, Kazunori; Inaba, Masayuki; Inoue, Hirochika. Visual Navigation using Ornnidirectional View Sequence. Proceedings of the IEEE/RSJ 1999 - International Conference on Intelligent Robots and Systems. pp.317-322.