Proposta de Otimização para Algoritmos de Detecção de Bordas em Imagens

S. P. Zambiasi
Faculdades Barddal
Grupo de Estudos em Robôs Inteligentes - GERIS
Av. Madre Benvenuta, 416
Trindade, Florianópolis/SC 88.036-500 BRASIL

R. J. Tramontin Jr.
Universidade Federal de Santa Catarina - UFSC
Departamento de Automação e Sistemas - DAS
Grupo de Sistemas Inteligentes de Manufatura - GSIGMA
Trindade, Florianópolis/SC – 88.040-900 BRASIL

Resumo- Uma das técnicas utilizadas na robótica para a visão do agente é a detecção de bordas das imagens capturadas por seus dispositivos. Com o objetivo de maximizar o desempenho destes agentes, o estudo de algoritmos mais rápidos e alternativos é de grande importância. Este artigo apresenta algumas técnicas simples e eficientes para detecção bordas e avalia este problema como sendo linearmente separável a partir da utilização de um *perceptron*. As técnicas propostas visam ser uma opção prática e rápida para aplicações de robótica, onde requisitos de tempo real são essenciais.

I. INTRODUÇÃO

Pesquisas de visão para robôs com finalidades de reconhecimento de formas e objetos envolvem uma enorme gama de problemas ainda não resolvidos ou que possuem apenas soluções parciais. O grande objetivo é que as máquinas possam perceber as formas e reconhecer os diferentes objetos imersos em um cenário ou mesmo reconhecer o próprio cenário complexo em seu todo. Aplicações com objetivos mais específicos, que visam buscar o reconhecimento de objetos pré-definidos ou a diferenciação e seleção dos mesmos já podem ser encontradas. Mesmo assim, estas ainda são muito limitadas a um espaço estático e em geral não trabalham bem em um ambiente dinâmico. Os aplicativos existentes, em geral, trabalham com muita informação de entrada e requerem *hardwares* que comportam grande processamento e armazenamento de imagens. Apesar dos Microcomputadores atuais já possuírem um alto grau de processamento e uma grande capacidade de armazenagem de dados, uma minimização no uso dos recursos ainda é importante quando se trata de robótica. O problema se agrava quando é necessário reduzir o tamanho e preço dos agentes robóticos, utilizando-se de microcontroladores de baixo custo, baixo processamento e capacidade de armazenamento reduzida. Isso justifica a utilização de apenas informações parciais e a necessidade de aplicações mais eficientes para compensar essa falta de recursos.

II. DETECÇÃO DE BORDAS

A visão computacional aplicada à robótica é, inquestionavelmente, um problema muito difícil de se resolver. No entanto, ela é de grande necessidade para as atividades requeridas por um agente inteligente em funcionamento [1].

A navegação de robôs requer informações do ambiente para que o mesmo possa se locomover sem colisões e executar as demais funções que ele objetiva. Essas informações são muitas e precisam ser tratadas para então ser interpretadas, gerando uma descrição do ambiente onde o robô se encontra.

Diversas técnicas são utilizadas em conjunto para a análise das imagens recebidas do ambiente, possibilitando a modelagem do mesmo. Uma destas técnicas é a detecção de bordas na imagem. Estas bordas, ou linhas, geralmente na visão estéreo em que duas câmeras capturam imagens do mesmo ponto de diferentes ângulos, possibilitam a análise de distâncias, dimensões e formas de objetos [2], assim como localização de paredes e caminhos por onde o agente deve seguir.

A tarefa de detectar bordas de objetos em uma imagem é muito fácil para uma pessoa. Porém, para um computador, esta tarefa se torna muito complicada. Isto porque, quando se fala em bordas de objetos, assume-se a compreenção da noção de

objetos, mas na realidade, essa noção é um dos objetivos da visão computacional, criando assim um ciclo vicioso. Por outro lado, usando algum tipo de função de descontinuidade de intensidade, tem-se o problema de ruídos, e, em geral, as imagens são simplificadas e quantizadas. Uma terceira dificuldade refere-se a algumas extremidades que são geradas por sombras ou reflexos de objetos [3].

Quando se trabalha com imagens, uma grande quantidade de informações precisa ser processada ou armazenada de forma permanente ou temporária. Porém, na robótica, busca-se uma maximização dos recursos e minimização dos custos de processamento. Neste caso, é de grande importância trabalhar apenas com informações parciais, não deixando de processar informações importantes para alcançar os objetivos.

O presente artigo apresenta algumas técnicas tradicionais para detecção de bordas em imagens e apresenta três alternativas às técnicas apresentadas. Por fim, uma comparação baseada no tempo de execução e na visualização dos resultados é feita entre os algoritmos apresentados.

A. O Operador de Roberts

O operador de Roberts, tal como descrito em [4], é o algoritmo mais antigo e simples para detecção de bordas. Ele utiliza um par de matrizes 2x2 (Fig. 1) para encontrar as mudanças nas direções x e y.

$$\begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}$$

Gx Gy

Fig. 1. Máscaras do operador de Roberts.

As matrizes são aplicadas sobre cada *pixel* da imagem para produzir medidas de gradiente separadas em cada direção (Gx e Gy). Quando uma matriz é aplicada a um dado *pixel*, este ocupa a posição superior esquerda (posição A) da matriz, cujo valor é multiplicado ao valor do *pixel*. Adicionalmente, os *pixels* à direita, abaixo e na diagonal direita serão também compreendidos pela matriz, ocupando respectivamente as posições B, C e D. Em outras palavras, o *pixel A* de entrada é comparado com os seus vizinhos para resultar nos gradientes Gx e Gy, conforme a seguinte fórmula:

$$\begin{aligned} Gx &= A - D \\ Gy &= B - C \end{aligned} \tag{1}$$

Estes valores são então combinados para calcular a magnitude absoluta do gradiente do *pixel A*, conforme a fórmula abaixo:

$$|G| = \sqrt{Gx^2 + Gy^2} \tag{2}$$

Entretanto, o gradiente pode ser computado de modo mais eficiente através da seguinte aproximação:

$$|G| = |Gx| + |Gy| \tag{3}$$

Se a magnitude calculada for maior do que o menor valor de entrada (definido de acordo com a natureza e qualidade da imagem que esta sendo processada), o *pixel* é considerado como parte de um borda.

B. O Operador de Sobel

O operador de Sobel [4] tem o funcionamento semelhante ao do operador Roberts, calculando os gradientes vertical e horizontal dos *pixels*, e combinando-os a seguir para obter o valor absoluto do gradiente. A diferença em relação ao operador de Roberts é que Sobel utiliza matrizes 3x3, conforme a Fig. 2.

$$\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \qquad \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Gx Gy

Fig. 2. Máscaras do operador de Sobel.

No algoritmo de Sobel, o *pixel* a ser avaliado encontra-se no centro da matriz, ou seja, na posição E, conforme a Fig. 3.

$$\begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix}$$

Fig. 3. Matriz das posições dos *pixels*.

Assim, o *pixel E* de entrada é comparado com seus vizinhos, em todas as direções, conforme a seguinte fórmula:

$$\begin{aligned} Gx &= A + 2 \cdot B + C - G - 2 \cdot H - I \\ Gy &= C + 2 \cdot F + I - A - 2 \cdot D - G \end{aligned} \tag{4}$$

A fórmula para calcular o valor absoluto do gradiente é a mesma do operador de Roberts, ou seja, (2) ou (3). Pelo fato de as matrizes serem 3x3 ao invés de 2x2, Sobel possui desempenho inferior a Roberts, pois é preciso realizar mais cálculos. Entretanto, Sobel é muito menos sensível a ruídos do que Roberts, além de apresentar resultados mais precisos.

C. Detecção de Bordas Utilizando Redes Neurais Artificiais

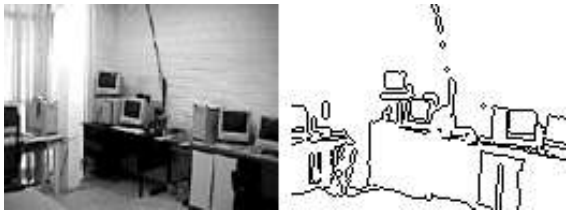
Para resolver o problema da detecção de bordas, Zambiasi [5] utilizou-se de uma rede neural artificial composta de 9 neurônios na camada de entrada, uma camada intermediária com 10 neurônios e um neurônio na camada de saída. Os neurônios de entrada representam uma matriz de 3x3, composta do *pixel* a ser avaliado e de todos os seus *pixels* vizinhos na imagem original. O neurônio da camada de saída representa o *pixel* correspondente na imagem de resultante. Para o treinamento da rede, foi utilizado o algoritmo de *Backpropagation*.

Um processamento prévio é aplicado aos *pixels*, de modo que todos cujo valor é menor que 100 ficam 0 (preto) e os *pixels* de valor maior que 100 ficam 255 (branco). Para a entrada da rede, os valores são normalizados para 0 e 1, ou seja, 0 é preto e 1 é branco.

Para o treinamento da rede, tomou-se como base uma imagem com resolução de 320x240 *pixels* e 256 tons de cinza (Fig. 4). Esta imagem foi dimensionada para 128x96 *pixels*, resultando em uma amostra menor que a imagem original. Utilizando um filtro do Adobe Photoshop® chamado "Traçado de Contorno", com nível 100, foi gerada uma imagem com as bordas detectadas apartir da imagem original. As duas imagens geradas (Fig. 5) foram então usadas no treinamento da rede.

Fig. 4. Imagem original.

Pegou-se aleatoriamente uma quantidade de 500 amostras de *pixels* da imagem original. Cada amostra contém, além do *pixel* (x, y) a ser avaliado, os seus oito *pixels* vizinhos que vão compor a entrada da rede. Para cada amostra A(x, y) foi usado, como saída desejada da rede em treinamento, o *pixel* correspondente B(x, y) da imagem filtrada previamente pelo Adobe Photoshop®.



(A) (B)

Fig. 5. Imagem original (A) redimensionada e imagem após o filtro do Adobe Photoshop (B).

Como resultado da execução da rede treinada, tem-se uma imagem que contém apenas as bordas detectadas na imagem (Fig. 6). O valor do nível utilizado no treinamento foi definido como 100. Porém, este valor pode ser alterado para a execução da rede, permitindo assim regular o algoritmo para se ajustar a uma dada situação, dependendo da luminosidade da imagem.

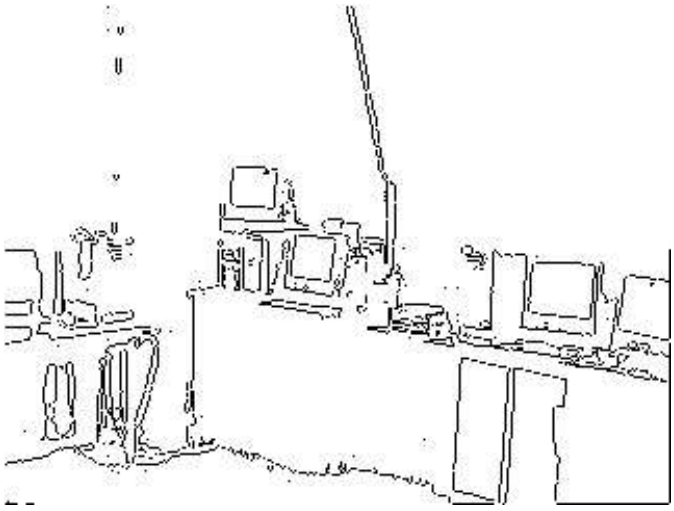


Fig. 6. Imagem Resultante.

O uso de redes neurais artificiais consiste em uma técnica alternativa para a detecção de bordas, mostrando-se uma abordagem eficiente para tal objetivo.

III. TÉCNICAS PROPOSTAS

O presente artigo apresenta três técnicas baseadas nos modelos anteriores como alternativas para a detecção de bordas em imagens. Elas visam simplificar algum aspecto em relação às originais,

visando otimizar o seu desempenho.

A primeira técinca simplifica a utilização de redes neurais artificiais, usando apenas um único percéptron. A segunda e a terceira são alternativas mais simples baseadas no operador de Roberts.

Como forma de comparação entre as técnicas, foram implementados protótipos para as três técnicas apresentadas anteriormente e as duas a serem apresentadas a seguir.

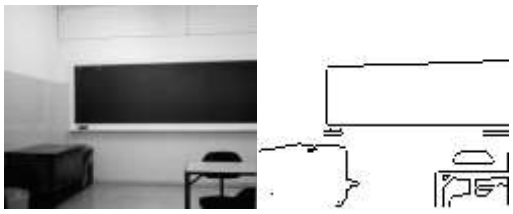
Para a comparação dos tempos de execução das implementações, foi selecionada a Fig. 7, com resolução de 2048x2560, e para as comparações visuais, a mesma figura foi selecionada, apenas diminuindo sua resolução para 320x240.

Fig. 7. Imagem original.

A. O Perceptron para Detecção de Bordas

O objetivo inicial deste experimento foi procurar uma alternativa mais simples que o modelo proposto por Zambiasi [5], sem fugir do escopo da inteligência artificial conexionista. *A priori* foi definida a utilização de um único *perceptron* para a detecção de bordas. Porém, a validação desse experimento depende do problema ser linearmente separável [6].

O método de aquisição utilizado seguiu de forma semelhante à definida por Zambiasi [5], conforme visto na seção II.C. Porém, nesta abordagem as amostras de *pixels* para o treinamento não foram obtidas aleatorimente.



(A) (B)

Fig. 8. Imagem de entrada (A) e saída (B) para o treinamento.

Para o treino, cada *pixel* da imagem original e todos seus vizinhos são selecionados como entrada do *perceptron* (Fig. 9). Para a verificação na saída, foi usado seu correspondente da imagem filtrada. Ainda na entrada, um *bias* de valor -1 é agregado ao *perceptron*.

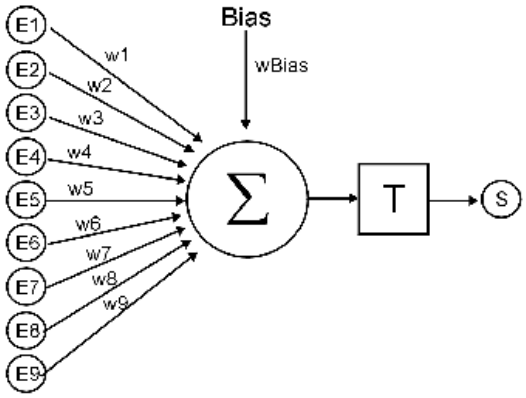


Fig. 9. *Perceptron*.

Como o objetivo do trabalho é diminuir o tempo de processamento na detecção de bordas, a função linear foi escolhida como função de transferência, ou ativação, por sua simplicidade.

Após executado o treinamento do *perceptron*, a imagem original (Fig. 7) foi usada para testes, gerando assim a imagem mostrada na Fig. 10 com suas bordas detectadas.

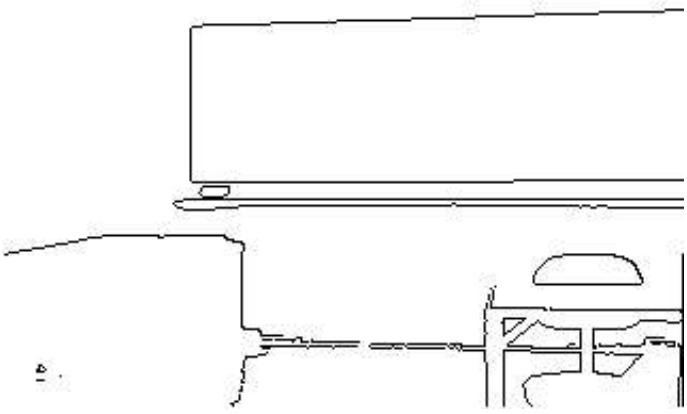


Fig. 10. Imagem resultante.

Segundo Barreto [7], os problemas envolvendo a implementação de uma função e que podem ser resolvidos por um único *perceptron* são definidos como problemas estáticos linearmente separáveis. Dessa forma, por existir um *perceptron* que resolve o problema definido por Zambiasi [5], este experimento pode caracterizá-lo como um

problema estático linearmente separável.

B. Um Condicional Simples como Alternativa à Detecção de Bordas

Essa técnica baseia-se na estrutura de máscaras do algoritmo de Roberts, onde apenas dois *pixels* são selecionados para a verificação da existência, ou não, de uma borda no local.

A partir da imagem original (Fig. 7), um processamento prévio é aplicado, assim como é feito nas outras técnicas, deixando a imagem apenas em preto e branco (Fig. 11).

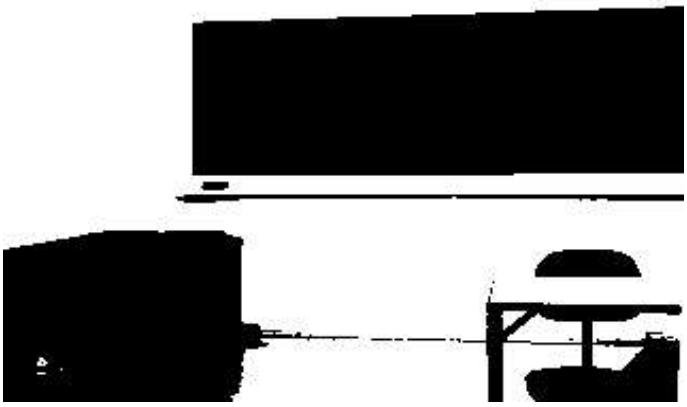


Fig. 11. Imagem filtrada em preto e branco.

Para a verificação da borda do *pixel* local (x1,y1) pertencente à imagem mostrada na Fig. 12, cada *pixel* (x1, y1) é comparado com seu vizinho (x2, y2). Se eles forem diferentes, então o *pixel* (x1, y1) da imagem de saida deve fazer parte de uma borda.

$$\begin{bmatrix} (x1,y1) & (x2,y1) \\ (x1,y2) & (x2,y2) \end{bmatrix}$$

Fig. 12. Máscara para o algoritmo do condicional.

Apesar deste algoritmo ser baseado no operador de Roberts, apenas uma das matrizes é usada como máscara. Isso pode gerar uma imagem menos sensível às informações, pois as que poderiam ser detectadas pela outra matriz não serão consideradas. Para corrigir este problema, pode-se criar um condicional composto, testando também a segunda matriz da máscara de Roberts. Sendo assim, se (x1,y1) é diferente de (x2,y2) ou (x2,y1) é diferente de (x1,y2), então o *pixel* faz parte de uma borda.

Os resultados visuais do Condicional Simples e Condicional Composto podem ser observados nas Imagens mostradas nas Fig. 15 e Fig. 16 respectivamente.

IV. COMPARAÇÃO

A. Comparação dos Tempos de Execução

Para comparar o tempo de execução dos algoritmos implementados, foi utilizada a imagem mostrada na Fig. 7 (2048x2560 *pixels*). Na Tabela I são apresentados os tempos de execução tirados em três tomadas de tempo.

Os testes foram executados em um computador Pentium 4 de 3GHz, com 1Gbyte de RAM, sistema operacional Linux e os protótipos foram implementados em linguagem C.

Convém ressaltar que os algoritmos de Roberts e de Sobel implementam a equação apresentada em (2) e as suas variantes implementam a otimização apresentada em (3).

TABELA I
COMPARAÇÃO DOS TEMPOS DE EXECUÇÃO (EM SEGUNDOS)

Algoritmo	1ª Execução	2ª Execução	3ª Execução
Roberts	0.60	0.60	0.60
Variante de Roberts	0.10	0.09	0.10
Sobel	0.64	0.64	0.65
Variante de Sobel	0.15	0.15	0.15
Rede Neural	19.39	19.49	19.48
Perceptron	1,05	1,06	1,05
Condicional Simples	0.08	0.08	0.08
Condicional Composto	0.06	0.06	0.06

Conforme pode-se observar na Tabela I, existe uma grande diferença dos algoritmos de Roberts e de Sobel quando comparados com suas variantes de cálculos mais simples. A Rede neural, por mais que tenha resolvido o problema da detecção de bordas, não é uma alternativa eficaz, pois seu tempo de processamento excede, em muito, as demais abordagens. Apesar do *Perceptron* possuir um tempo de execução muito menor do que a Rede Neural, ele está muito acima dos tempos de execução de Sobel e de Roberts. Por fim, os algoritmos dos Condicionais (tanto o simples quanto o composto) obtiveram tempos de resposta ainda menores que os algoritmos variantes dos operadores de Roberts e de Sobel.

B. Comparação Visual

Como nesse artigo o fator em evidência é o tempo de resposta, apenas os algoritmos mais rápidos foram eleitos para a comparação visual. Os algoritmos escolhidos são o variante de Roberts, o variante de Sobel e os Condicionais. Para facilitar a visualização, a imagem original na Fig. 7. foi

reduzida para 320x240.

Observando os resultados de Roberts (Fig. 13) e de Sobel (Fig. 14) percebe- se que o primeiro apresenta linhas não tão bem definidas se comparado com o segundo. Isto ocorre devido ao fato de que Roberts utiliza uma matriz de 2x2 e Sobel uma matriz de 3x3. Além disso as linhas de Sobel são mais bem definidas pois para cada *pixel* da imagem são considerados todos os seus adjacentes, o que ocorre parcialmente com Roberts .

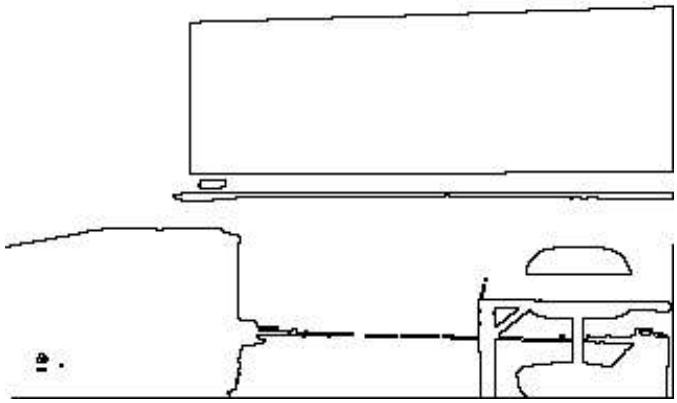


Fig. 13. Variante de Roberts .

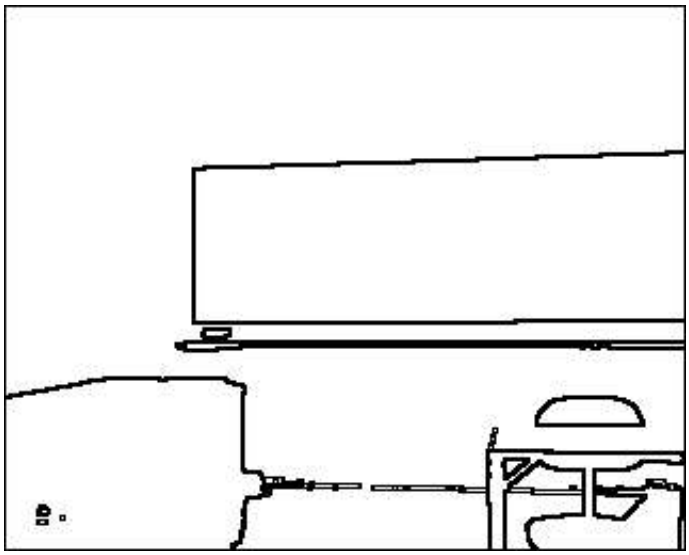


Fig. 14. Variante de Sobel

Por serem variantes de Roberts, os resultados apresentados pelos Condicionais Simples (Fig. 15) e Composto (Fig. 16) foram semelhantes aos resultados apresentados por Roberts (Fig. 13). O Condicional Composto se aproxima mais do resultado de Roberts, pois considera as duas matrizes. Já o Condicional Simples apresenta um resultado um pouco menos preciso, visto que utiliza apenas uma das matrizes. Entretanto, a qualidade do seu resultado não é muito diferente quando comparado com as demais técnicas, e portanto, as diferenças não são muito relevantes visualmente.

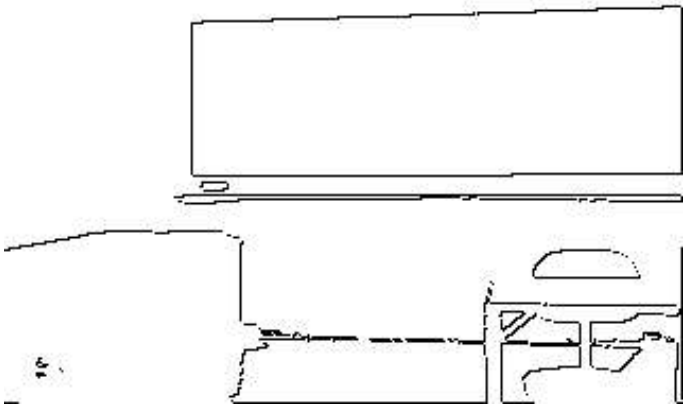


Fig. 15. Condicional Simples.

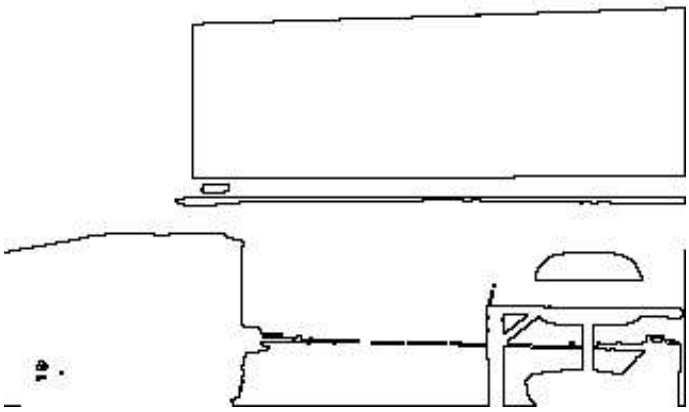


Fig. 16. Condicional Composto

V. CONSIDERAÇÕES FINAIS

A detecção de bordas é um problema essencial para as mais diversas aplicações de robótica. Este artigo apresentou algumas técnicas para tratar este problema, e propôs três variantes de maior desempenho e com resultados visuais igualmente satisfatórios.

Mais especificamente, foi proposta uma variante da abordagem de redes neurais que utiliza um único neurônio (*perceptron*) e duas variantes do algoritmo de Roberts. Apesar da abordagem que usa o

perceptron não ter um desempenho muito bom em termos absolutos, quando comparada com o desempenho de uma rede neural, ele é consideravelmente superior.

Pelo fato do problema da detecção de bordas em imagens ter sido resolvido através de um único *perceptron*, conclue-se que este é um problema linearmente separável.

De modo a complementar os resultados obtidos, faz-se necessário analisar outros aspectos não cobertos neste trabalho, tais como testes com diferentes imagens (com diferentes luminosidades e contrastes) e também a implementação dos algoritmos em um agente robótico para testes mais elaborados.

REFERÊNCIAS

[1] R. Arkin, C. Ronald; "Behavior-based robotics". The MIT Press, Cambridge, Massachusetts, p 237-299, 1998.

[2] S. Taraglio, A. Zanela, M. Salerno, at all; "A neural network to visually understand and autonomously navigate unknown environments". 1 ENEA - C.R. Casaccia, Via Anguillarese, 301-00060 Rome, Italy 2 Department of Electronic Engineering , University of Rome "Tor Vergata", Via di Tor Vergata, 110 - 00133 Rome, Italy.

[3] O. Faugeras; "Three-dimensional computer vision: a geometric viewpoint". 3 ed, The MIT Press, Cambridge, Massachusetts, p.69-119, 1993.

[4] R. Fisher, S. Perkins, A. Walker, E. Wokfart; "Feature Detectors". 2003. http://homepages.inf.ed.ac.uk/rbf/HIPR2/featops.htm Acessado em 28/09/2005.

[5] S. Zambiasi; "Detecção de bordas em Imagens Utilizando Redes Neurais Artificiais". Material do Seminário de Visão Computacional, CPGCC/UFSC, 2001.2.http://www.inf.ufsc.br/~visao/2001/saulo/saulo_artigo3.html - acessado em Outubro de 2005.

[6] S. Haykin, "Redes neurais: princípios e prática". - 2.ed. - Porto Alegre: Bookman, 2001.

[7] J. Barreto, "Inteligência Artificial no Limiar do Século XXI". ppp Edições, Florianópolis, 3 edição, 2001.