Estudo comparativo de técnicas de detecção de cantos em imagens digitais

The comparison study of the techniques for the detection of corners in digital images

Rodrigo Fernandes Freitas
rodrigo@lesc.ufc.br
Universidade Federal do Ceará

Rodrigo Carvalho de Souza Costa
rodcosta@lesc.ufc.br
Universidade Federal do Ceará

Paulo César Cortez
cortez@lesc.ufc.br
Universidade Federal do Ceará

Resumo

Os vértices são os pontos mais importantes do contorno, capazes de representar eficientemente uma forma, sendo muito utilizados em várias tarefas de visão computacional, como visão estéreo, interpretação 3D e estimação de movimento. Neste trabalho, é realizada a análise de desempenho e do esforço computacional de diferentes abordagens para detecção de vértices existentes na literatura, utilizando métodos baseados em Transformada Wavelet, curvatura e morfologia matemática, bem como é proposta uma nova técnica de detecção de vértices, modificando-se uma técnica de Transformada Wavelet. Para realização das simulações, é utilizado o Matlab/Simulink. Os resultados deste trabalho possibilitam a determinação de um algoritmo que associe uma alta taxa de detecção de vértices a um baixo esforço computacional

Palavras-chave: Detecção de vértices. Visão computacional. Simulink.

Abstract

The corners are the most important points of the contour, they can represent efficiently a shape. The corners are characteristics often used in various tasks of Computer Vision, such as stereo vision, 3D rendering and motion estimation. This work presents analysis of performance and computational effort for different approaches to detection of corners existents in the literature using methods based on Wavelet Transform, Curvature and Mathematical Morphology, and proposes a new technique to detect corners, changing a Wavelet Transform technique. To perform the simulations Matlab / Simulink are used. The results enable to determine an algorithm that combines a high rate of detection of corners at a low computational effort.

Keywords: Corner Detection. Computer Vision. Simulink

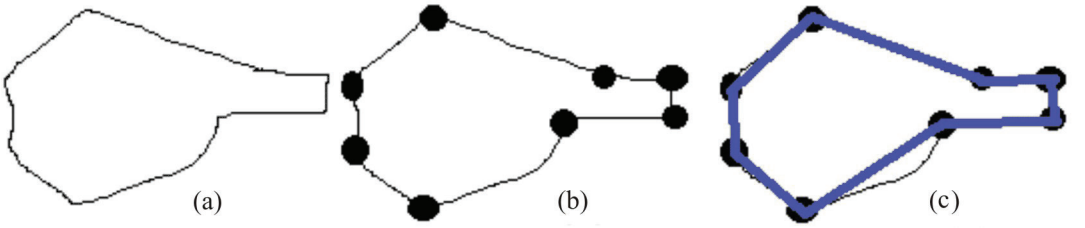
1 Introdução

Os vértices são pontos muito importantes de um objeto, visto que a informação sobre uma forma se concentra neles. São muito úteis na área de processamento de imagens digitais para representar e analisar formas e objetos (SHUI, ZANG, 2013; MASOOD, SARFRAZ, 2007).

Vértices, ou pontos críticos, são definidos como pontos da imagem em que a linha do contorno do objeto apresenta uma variação brusca na sua direção, ou seja, um ponto com alto valor de amplitude no sinal de curvatura (PAULA JÚNIOR, 2007).

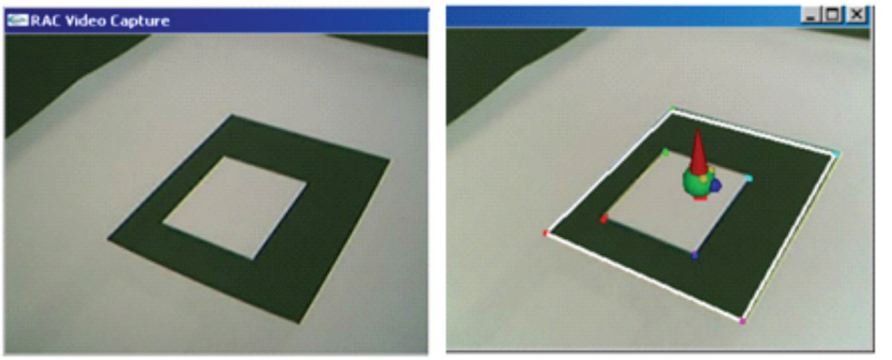
Pontos de vértices são interessantes pelo fato de serem formados por duas ou mais bordas, e as bordas geralmente definem o limite entre dois diferentes objetos ou partes do mesmo objeto (PARKS, GRAVEL, 2009). Um exemplo de utilização dos vértices para a representação de uma forma é mostrado na Figura 1.

Figura 1 - Exemplo do uso de vértices para a representação de uma forma: (a) exemplo de uma forma qualquer; (b) vértices extraídos da forma; (c) comparação entre a forma original e a forma reconstruída a partir dos vértices extraídos.



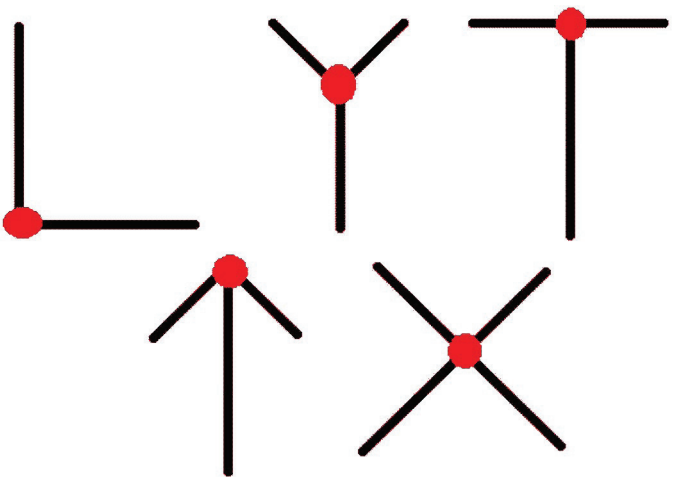
Uma vantagem do uso de vértices para a representação de uma forma é que eles correspondem à intuição humana de distinguir visualmente os pontos característicos da forma (XU *et al.*, 2010; PAULA JÚNIOR, 2007). Um exemplo de utilização da extração de vértices é mostrado na Figura 2. Na Figura 2(a), é mostrada uma imagem com uma folha em branco, a qual serve como referência para a inserção do objeto virtual. Na Figura 2(b), é mostrada uma imagem com os vértices da folha reconhecidos e o objeto virtual inserido baseado nesses vértices.

Figura 2 - Exemplo de uso da detecção de vértices para inserção de objetos virtuais em imagens: (a) imagem original com folha em branco que serve de referência para a inserção do objeto virtual. (b) vértices da folha em branco reconhecidos e inserção do objeto virtual baseado na localização dos vértices.



Os tipos mais comuns de vértices são as junções L, Y, T, Seta (*Arrow*) e X (PARKS, GRAVEL, 2009), conforme mostrado na Figura 3.

Figura 3 - Exemplos dos tipos de vértices presentes em imagens digitais (na linha de cima: junção em L, junção em Y e junção em T; na linha de baixo: junção em seta e junção em X).



Outra característica importante dos pontos de vértices é a sua invariância à rotação, translação e escala (PAULA JÚNIOR *et al.*, 2013; XU *et al.*, 2010; GAO *et al.*, 2006). Além disso, esse tipo de técnica possui um baixo custo computacional das técnicas de detecção de vértices, quando comparado com outros métodos de extração de atributos baseados em comparação de imagens (PARKS, GRAVEL, 2009).

Os vértices são utilizados em diversas aplicações, como visão estéreo, interpretação 3D, estimação de movimento e reconhecimento de estruturas através de movimentos (ZHANG, DONGSHENG, 2010; LAGANIÈRE, 1998). São também aplicados em rastreamento de objetos, casamento de imagens, formação de mosaicos, realidade virtual aumentada e reconhecimento de formas (ROSTEN, DRUMMOND, 2006). A localização eficiente é de fundamental importância para que as aplicações funcionem adequadamente.

Os métodos de detecção de vértices são separados em duas vertentes. A primeira consiste nas técnicas em que a imagem de entrada é convertida para uma imagem binária, é extraído o contorno e aplicado o método de detecção dos vértices sobre este, buscando-se os pontos de maior curvatura. A segunda vertente abrange técnicas que são aplicadas diretamente sobre a imagem digital, sendo esta em nível de cinza ou colorida (ZHANG, DONGSHENG, 2010; GONZALEZ, WOODS, 2008). Devido à sua importância, diversos algoritmos de detecção de vértices são apresentados na literatura (ROSTEN, DRUMMOND, 2006).

O Simulink é uma plataforma para simulação em multidomínios e projeto de sistemas dinâmicos baseados em modelos. Sua grande abrangência permite que ele seja utilizado nas mais diversas áreas da engenharia, como controle preditivo, controle robusto, identificação de sistemas, redes neurais, lógica *fuzzy*, entre outras (MATHWORKS, 2006). Ele fornece um ambiente gráfico interativo e um conjunto de blocos de bibliotecas personalizáveis, podendo ser estendido para aplicações especializadas, possibilitando a implementação de um conjunto de blocos especializados em detecção de vértices, mais especificamente, um conjunto formado pelos algoritmos avaliados neste estudo.

Neste trabalho, é realizado um estudo comparativo entre as técnicas de detecção de vértices baseadas em curvatura, morfologia matemática e Transformada Wavelet, bem como a implementação de uma modificação no algoritmo de Transformada Wavelet, a fim de melhorar o seu desempenho e a sua detecção de vértices utilizando o Simulink, destacando as vantagens e desvantagens de cada uma das técnicas. Os resultados possibilitam a identificação do algoritmo que possua a melhor relação custo-benefício entre eficiência de detecção e custo computacional.

O restante do artigo está organizado da seguinte maneira: a próxima seção apresenta a fundamentação teórica e a descrição de alguns algoritmos de detecção de vértices presentes na literatura. A seção seguinte apresenta a metodologia utilizada para a realização do trabalho e descreve o novo método de detecção de vértices proposto neste trabalho. Em seguida, são apresentados os resultados dos testes e, finalmente, na última seção, são feitas algumas conclusões e considerações acerca deles.

2 Fundamentação teórica

2.1 Imagem digital

Uma imagem digital é caracterizada por uma função em duas dimensões *f(x,y)*, em que *x* e *y* são as coordenadas de um determinado ponto, denominado *pixel* (*picture element*), e a amplitude de *f* de qualquer par ordenado *(x,y)* é chamado de intensidade ou nível de cinza. Essa função pode ser representada matematicamente por uma matriz, em que cada elemento dessa matriz é a amplitude de *f* e as coordenadas *x* e *y* correspondem às colunas e linhas, respectivamente (GONZALEZ, WOODS, 2008).

As imagens digitais podem ser coloridas ou em níveis de cinza. No sistema RGB, por exemplo, as imagens coloridas assumem três canais –R (*Red*), G (*Green*) e B (*Blue*) –, sendo representadas por uma matriz para cada canal. Pode-se denominar de vídeo digital uma sequência de imagens digitais, chamadas de *frames* (GONZALEZ, WOODS, 2008).

2.2 Métodos de detecção de vértices

Desde que os primeiros detectores de vértices foram desenvolvidos no final dos anos 1970, dúzias de detectores de vértices foram propostas, os quais podem ser classificados nas seguintes abordagens (PARKS, GRAVEL, 2009):

- **Métodos de relação de borda:** esta abordagem aplica operadores de geometria diferenciais para detectarem os vértices. Kitchen e Rosenfeld (1982) propuseram uma medida de grau de vértice para cada *pixel* baseado

na mudança da direção do gradiente (derivadas de segunda ordem) ao longo do contorno de borda ponderada pela magnitude do gradiente local. Os vértices são identificados pelo valor de máximo local dessa medida. Esse detector de vértices apresenta uma alta sensibilidade, visto que se baseia em termos de derivadas de segunda ordem e foi demonstrado ter uma baixa taxa de repetitibilidade e localização.

- **Métodos por topologia:** esta abordagem pode ser vista como uma busca por bordas de curvatura altas (isto é, pontos de máximo na superfície da imagem) calculando a curvatura gaussiana da imagem.
- **Autocorrelação:** esta abordagem considera uma janela local na imagem e determina a mudança média de intensidade a partir de uma pequena variação de uma janela em várias direções. Já que os vértices exibem uma grande variação de intensidade em todas as direções, esse operador é um detector de vértices, porém, com uma definição bem mais ampla deste.
- **Métodos alternativos:** métodos que não se encaixam nas categorias anteriores. Pode-se citar o operador de Espaço-Escala de Curvatura, que detecta vértices procurando diretamente pelo valor máximo local da curvatura absoluta. Esse método detecta vértices utilizando a noção intuitiva de localizar quando o contorno de um objeto faz uma curva abrupta.

Neste artigo, são comparados métodos de diferentes abordagens, descritos a seguir.

2.3 Detecção de vértices utilizando curvatura

A curvatura é um dos atributos mais importantes que podem ser extraídos dos contornos. Na verdade, uma forte motivação biológica tem sido identificada para se estudar a curvatura, a qual é aparentemente uma pista interessante explorada pelo sistema visual humano (COSTA, CÉSAR JÚNIOR, 2001).

A curvatura *k(t)* de uma curva paramétrica *c(t) = (x(t), y(t))* é definida como:

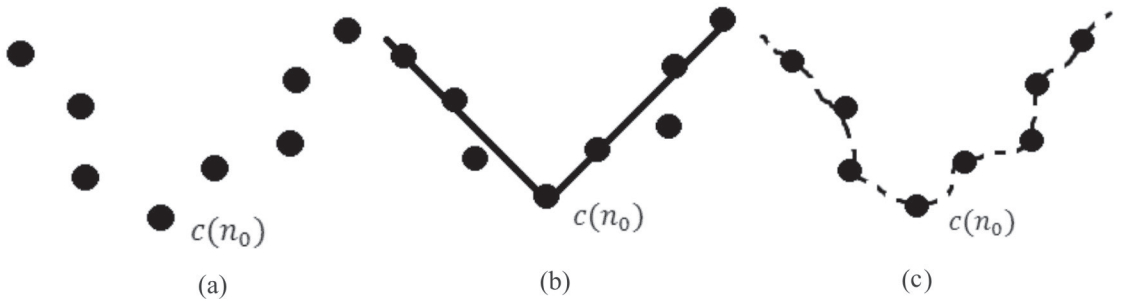
$$k(t) = \frac{\dot{x}(t)\,\dot{y}(t) - \dot{x}(t)\,\dot{y}(t)}{(\dot{x}(t)^2 + \dot{y}(t)^2)^{3/2}} \qquad (0)$$

Fica claro, pela equação, que para estimar a curvatura é necessário o cálculo das derivadas de *x(t)* e *y(t)*. Devido ao fato de o contorno ter natureza discreta (ou seja, espacialmente amostrado), o cálculo das derivadas se torna um problema computacional, dificultando a utilização direta dessa fórmula (COSTA, CÉSAR JÚNIOR, 2001).

Uma solução para o cálculo dessa equação é dada por Passarinho *et al*. (2004), que utilizam a Transformada de Fourier e as suas propriedades para resolverem essa equação.

Uma abordagem básica que evita o cálculo da fórmula é a definição de medidas de curvatura alternativas baseadas nos ângulos entre vetores definidos em termos dos elementos discretos do contorno, como é mostrado na Figura 4

Figura 4: Abordagem para estimação da curvatura de uma curva discreta: parte do contorno discreto (a), medida da curvatura baseada no ângulo (b) e estimação da curvatura baseada em interpolação.



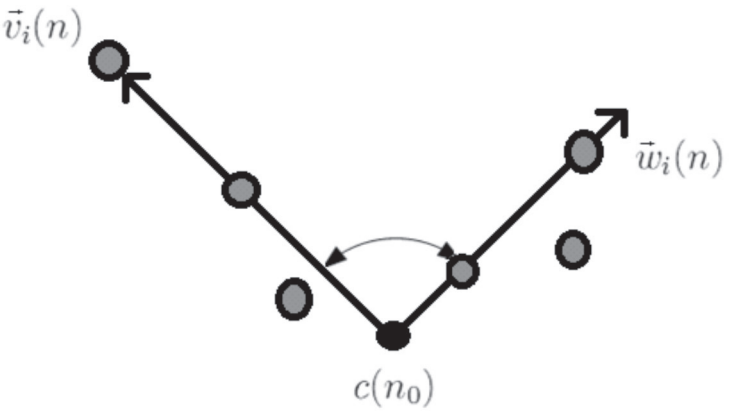
As abordagens que foram propostas para se estimar o ângulo definido pelos vetores ao longo do contorno diferem na forma como esses vetores são encaixados ou no método aplicado para se estimar esses ângulos. Considere *c(n) = (x(n), y(n))* como sendo uma curva discreta. Os seguintes vetores podem ser definidos (COSTA, CÉSAR JÚNIOR, 2001):

$$v(n) \quad (x(n) \quad x(n \quad i), y(n) \quad y(n \quad i)) \tag{0}$$

$$w(n) \quad (x(n) \quad (x(n \quad i), y(n) \quad y(n \quad i) \tag{0}$$

em que $\vec{v}_i(n)$ é o vetor formado pelo ponto atual do contorno e um ponto do contorno à esquerda, e $\vec{w}_i(n)$ é o vetor formado pelo ponto atual do contorno e um ponto do contorno à direita, conforme mostrado na Figura 5.

Figura 5 - Indicação da curvatura baseada no ângulo: os vetores da esquerda e da direita representam $\vec{v}_i(n)$ e $\vec{w}_i(n)$, respectivamente.



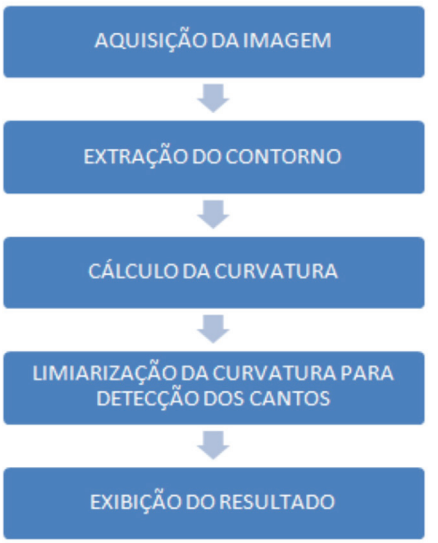
O modelo digital de pontos de curvatura alta proposto por Johnston e Rosenfeld (1973) é definido pela seguinte equação:

$$\vec{r}_i(n) = \frac{\vec{v}_i(n)\vec{w}_i(n)}{\|\vec{v}_i(n)\|\|\vec{w}_i(n)\|} \tag{0}$$

em que $r_i(n)$ é o cosseno do ângulo entre os vetores $v_i(n)$ e $w_i(n)$. Dessa forma, tem-se que $-1 \le r_i(n) \le 1$, com $r_i(n) = -1$ quando o ângulo se torna 180° e $r_i(n) = 1$ quando o ângulo se torna 0° (o menor ângulo possível). Nesse sentido, $r_i(n)$ pode ser usado como uma medida capaz de localizar pontos de curvatura alta, ou seja, maior que um limiar.

Um diagrama de blocos do algoritmo de detecção de vértices baseado em curvatura é mostrado na Figura 6.

Figura 6 - Diagrama de blocos do algoritmo de curvatura.



O algoritmo de curvatura funciona da seguinte maneira: primeiro, segmenta-se na imagem os objetos de interesse através da técnica de limiarização (os valores de limiares utilizados dependem dos objetos a serem segmentados) e extrai-se deles os seus contornos; depois, aplica-se o cálculo da curvatura sobre todos os pontos do contorno, achando o valor da curvatura de cada um deles; realiza-se a limiarização dos valores de curvatura encontrados, a fim de determinar quais correspondem a vértices e quais não; e, por fim, faz-se a exibição dos resultados.

2.4 Detecção de vértices utilizando morfologia matemática

A morfologia matemática na área de processamento de imagens denota uma ferramenta para extrair componentes da imagem úteis na representação e descrição da forma do objeto, como bordas e esqueletos, e para a realização de pré-processamento ou pós-processamento, tais como filtragem morfológica e afinamento (GONZALEZ, WOODS, 2008).

O princípio de todos os operadores morfológicos básicos é investigar a imagem através de um elemento estruturante, o qual consiste em um conjunto de *pixels* com uma origem definida. Dessa forma, para se analisar um dado ponto da imagem, o elemento estruturante é transladado sobre a imagem de modo que sua origem coincida com esse ponto. Assim, podemos definir as operações morfológicas básicas de erosão e dilatação (LAGANIÈRE, 1998).

Considerando a imagem como sendo A e o elemento estruturante como sendo B, a erosão pode ser definida, de acordo com Gonzalez e Woods (2008), por:

$$A \ominus B = \{z \mid (B)_z \subseteq A\} \tag{0}$$

Em outras palavras, a erosão de A por B é o conjunto de todos os pontos z tais que B, transladado por z, esteja contido em A. Já a dilatação pode ser definida por:

$$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \varnothing\} \tag{0}$$

Ou seja, a dilatação de A por B é o conjunto de todos os deslocamentos z tais que a reflexão de B, denotada por $\hat{B}$, e A coincidam em pelo menos um ponto.

Duas outras operações morfológicas podem ser definidas a partir dessas duas operações básicas: abertura e fechamento. A abertura é definida por:

$$A \circ B = (A \ominus B) \oplus B \tag{0}$$

A abertura é a erosão de A por B, seguida da dilatação do resultado por B. Já o fechamento é definido por:

$$A \bullet B = (A \oplus B) \ominus B \tag{0}$$

Ou seja, a dilatação de A por B, seguida pela erosão do resultado por B.

As definições de abertura e fechamento assumem que é utilizado o mesmo elemento estruturante B nas operações de erosão e dilatação.

O algoritmo de detecção de vértices por morfologia matemática proposto por Laganière (1998) apresenta um fechamento assimétrico, isto é, a dilatação de uma imagem usando um dado elemento estruturante seguida por uma erosão utilizando outro elemento estruturante. O objetivo é fazer a dilatação e a erosão complementares em termos do tipo de vértices que elas afetam.

Dessa forma, o algoritmo consiste em um fechamento no qual a dilatação é realizada com um elemento estruturante em cruz (B_1), seguida por uma erosão com um elemento estruturante em forma de losango (B_2), definida por:

$$A \bullet \{B_1, B_2\} = (A \oplus B_1) - B_2 \tag{0}$$

Porém, apenas essa primeira etapa de fechamento não detecta todos os vértices existentes na imagem. Para resolver isso, é realizada outra operação de fechamento, definida a seguir, com elementos estruturantes em forma de X (B_3) e em forma de quadrado (B_4):

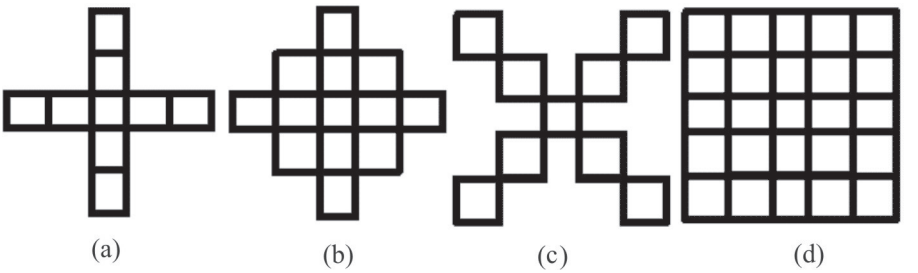
$$A \bullet \{B_3, B_4\} = (A \oplus B_3) - B_4 \quad (0)$$

Para detectar todos os vértices da imagem, recorre-se ao seguinte operador:

$$c(A) = |A \bullet \{B_1, B_2\} - A \bullet \{B_3, B_4\}| \quad (0)$$

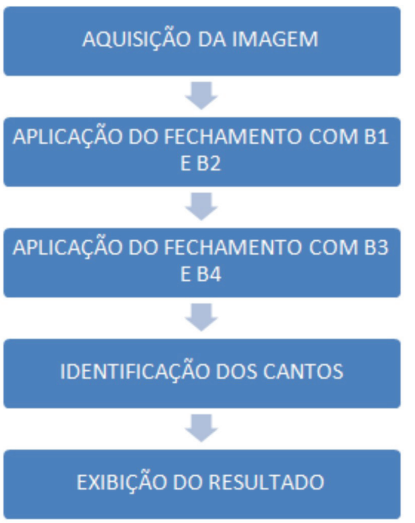
em que B1 é representado esquematicamente na Figura 7(a); B2, na Figura 7(b); B3, na Figura 7(c); e B4, na Figura 7(d).

Figura 7 - Elementos Estruturantes utilizados no Fechamento Assimétrico. (a) Elemento estruturante em forma de cruz. (b) Elemento estruturante em forma de losango. (c) Elemento estruturante em forma de X. (d) Elemento estruturante em forma de quadrado.



O diagrama de blocos do algoritmo de detecção de vértices por morfologia matemática é mostrado na Figura 8.

Figura 8 - Diagrama de blocos do algoritmo baseado em Morfologia Matemática.



2.5 Transformada Wavelet

A Transformada Wavelet é uma ferramenta de análise de espaço-frequência que tem sido bastante estudada e pesquisada nas últimas décadas. As *wavelets* vêm sendo aplicadas nas áreas de computação gráfica e de processamento de imagens, sendo nesta última em aplicações de edição, compressão, controle automático do nível de detalhes para filtragem e análise de textura (COSTA, 2006).

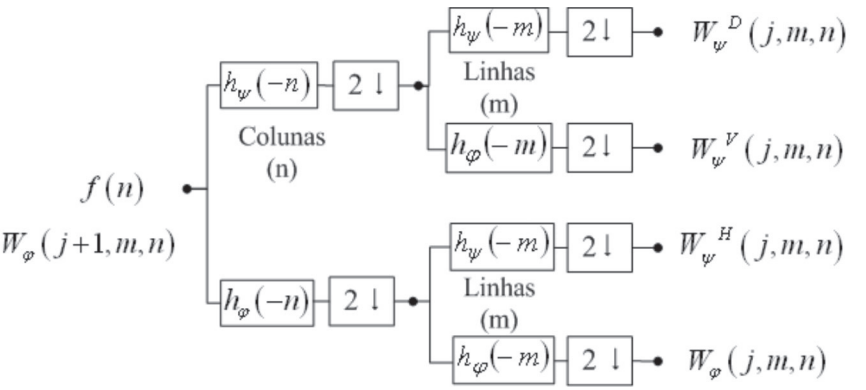
A Transformada Wavelet 2D, utilizada em sinais bidimensionais como imagens, é uma simples extensão da Transformada Wavelet 1D. No caso da Transformada Wavelet 2D, é necessário uma função de escalamento $\varphi(x,y)$ e três *wavelets* bidimensionais $\psi^H(x,y)$, $\psi^V(x,y)$ e $\psi^D(x,y)$ (GONZALEZ, WOODS, 2008).

ψ^H Em uma imagem, as *wavelets* medem variações da intensidade do nível de cinza em diferentes direções, em que mede as variações através das colunas (por exemplo, bordas horizontais), ψ^V mede as variações através das linhas (por exemplo, bordas verticais) e ψ^D mede as variações diagonais (GONZALEZ, WOODS, 2008).

Em duas dimensões, a Transformada Wavelet pode ser computada com um algoritmo piramidal utilizando filtros digitais. Devido à separabilidade das funções de escalamento e *wavelet*, a Transformada Wavelet 2D pode ser vista como a Transformada Rápida Wavelet 1D (FWT-1D), aplicada na direção horizontal e, em seguida, na direção vertical, ou vice-versa (COSTA, 2006).

Primeiramente, convoluem-se as linhas da imagem com um filtro unidimensional, retendo uma a cada duas linhas. Em seguida, é realizada a convolução das colunas do sinal resultante por outro filtro unidimensional, retendo uma a cada duas colunas, conforme mostrado na Figura 9 (GONZALEZ, WOODS, 2008).

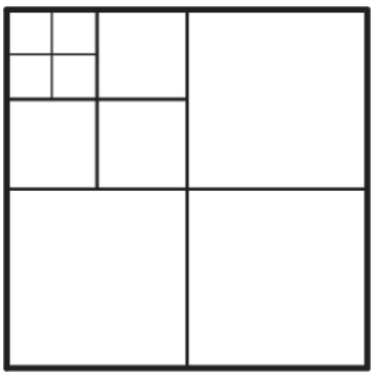
Figura 9 - Decomposição *wavelet* 2D.



Fonte: Costa, 2006.

Aplicando esse procedimento recursivamente no coeficiente de aproximação $W_\varphi(j,m,n)$, são gerados novos coeficientes $(W_\varphi(j-1,m,n)$, $W_\psi^H(j-1,m,n)$, $W_\psi^V(j-1,m,n)$ e $W_\psi^D(j-1,m,n)$, extraindo informações horizontais, verticais e diagonais em uma maior escala, resultando numa decomposição semelhante a uma árvore, com imagens de detalhes em diferentes escalas e orientações, mostradas na Figura 10, também chamada de decomposição *wavelet* padrão (piramidal).

Figura 10 - Decomposição *wavelet* do tipo padrão (piramidal).



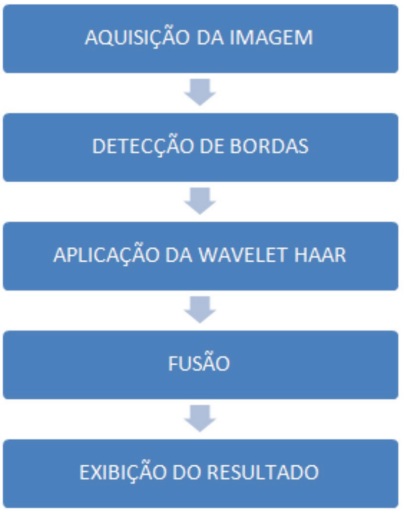
Fonte: Costa, 2006.

Devido a essa separação, pode-se reconstruir a imagem apenas com os vértices e as bordas, desprezando o coeficiente de aproximação e filtrando os coeficientes de detalhes para detectar os vértices da imagem.

2.6 Método proposto por Carneiro (2007)

Carneiro (2007) propõe um método baseado em Transformada Wavelet para a detecção dos vértices, cujo diagrama de blocos é mostrado na Figura 11.

Figura 11 - Diagrama de blocos do método proposto por Carneiro (2007).



O método possui as seguintes etapas:

1. Aplicação de um algoritmo de detecção de bordas (o detector de Sobel, Prewit ou Laplaciano da Gausiana). Em testes feitos pelo autor, o melhor resultado foi obtido para o método de Sobel. Esse método resulta em um mapa de bordas contendo valores de 0 para bordas e 1 para o fundo.
2. Sobre o mapa de bordas, é aplicada a Transformada Wavelet, utilizando as funções de escalamento e Wavelet Haar. Esta resulta nos coeficientes de aproximação e detalhes (horizontais, verticais e diagonais).
3. Os coeficientes de detalhes possuem as informações necessárias para localizar os vértices. A informação contida nestes é fundida através da soma dos quadrados dos coeficientes, ou seja, é criada uma imagem de fusão em que cada *pixel* é calculado como a soma dos quadrados dos coeficientes de detalhes dos *pixels* correspondentes nas imagens de detalhes. Quanto mais acentuado o vértice, maior será o valor da intensidade dos *pixels* na imagem de fusão.
4. Sobre a imagem de fusão, aplica-se uma limiarização com limiar fixo, resultando no conjunto de vértices das formas.

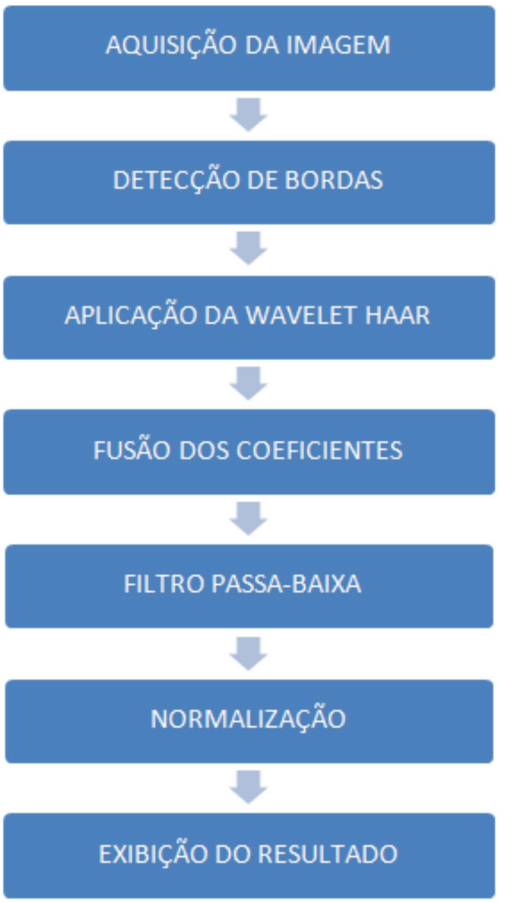
3 Método proposto: detector de vértices Freitas-Costa

Este trabalho propõe uma modificação do método proposto por Carneiro (2007), o método Freitas-Costa, que melhora a eficiência da detecção dos vértices. Ele é composto pelas seguintes etapas:

1. Aplicação do detector de bordas
2. Aplicação da Transformada Wavelet Haar
3. Aplicação da fusão dos coeficientes de detalhes
4. Aplicação de um filtro passa-baixa
5. Normalização do resultado após filtragem
6. Limiarização do resultado normalizado

O diagrama de blocos do algoritmo proposto neste trabalho é mostrado na Figura 12.

Figura 12 - Diagrama de blocos do algoritmo proposto.



4 Metodologia

Este trabalho realiza uma comparação do método de detecção de vértices proposto com métodos tradicionais de detecção de vértices através de curvatura, morfologia matemática e o Método Wavelet proposto por Carneiro (2007). Nesta seção, é descrita a metodologia proposta para a realização da comparação das técnicas de detecção de vértices.

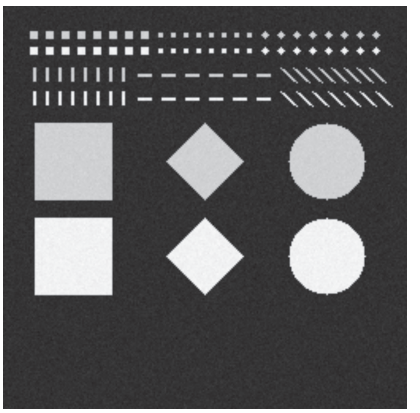
4.1 Imagens utilizadas

Nesta subseção, são apresentadas as imagens utilizadas nos testes das simulações. Foram utilizadas três imagens neste trabalho. A primeira, mostrada na Figura 13, é uma imagem de baixa complexidade, um objeto com contraste máximo, cujos vértices são formados apenas pelo encontro de duas retas horizontais.

Figura 13 - Imagem de baixa complexidade.

A segunda, apresentada na Figura 14, é uma imagem de teste com complexidade intermediária, possuindo objetos de diferentes tonalidades de cinza e de bom contraste entre o fundo e seus vértices, formados pelo encontro de retas horizontais com verticais e encontro de retas diagonais. O círculo possui vértices devido à natureza digital do contorno.

Figura 14 - Imagem de média complexidade.



Fonte: Parks e Gravel, 2009.

A terceira, apresentada na Figura 15, é uma imagem de alta complexidade, possuindo objetos com uma grande variação de tamanho e tonalidades de cinza, contraste variável entre o fundo e vértices no quadrado e losango.

Figura 15 - Imagem de alta complexidade.

4.2 Simulink

O Simulink é uma ferramenta de simulação, modelagem e análise de sistemas dinâmicos. Através da combinação de vários diagramas de blocos, o Simulink é capaz de simular sistemas de pequeno e grande porte, além de se integrar com funções desenvolvidas em Matlab (TAYLOR *et al*., 2004).

A estrutura modular do Simulink permite o agrupamento de modelos dentro de hierarquias que proveem uma visão geral do sistema e uma fácil manutenção de componentes e sistemas complexos. O Simulink utiliza um ambiente gráfico baseado em diagramas de blocos que suportam diferentes operações, como funções aritméticas, entrada e saída de dados, funções de transferência, modelos de estado de espaços, dentre outras (KALAGASIDS *et al*., 2007).

A partir de pequenos blocos, um conjunto de bibliotecas de várias áreas pode ser desenvolvido. Cada biblioteca é formada por vários sub-blocos ou submodelos que representam uma função específica de uma área específica. A partir do arranjo de vários desses blocos na tela e através das conexões dos blocos com algumas variáveis e constantes, é possível simular vários sistemas de equações (TAYLOR *et al*., 2004).

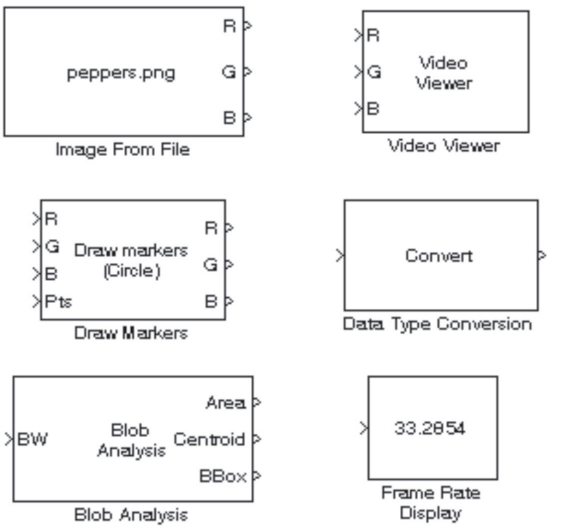
Com o uso do Simulink, um desenvolvedor se preocupa mais com a implementação do modelo físico do projeto, esquecendo das questões de discretização e integração entre blocos, possibilitando assim um menor tempo para simular a eficiência de um algoritmo, para em seguida ser implementado em um sistema específico (KALAGASIDS *et al*., 2007).

Um pacote muito importante para possibilitar a simulação de algoritmos de visão artificial é o pacote de processamento de vídeo e imagem. Ele possui um conjunto de funções úteis para a comunidade de processamento de imagens, composta por funções de aquisição, filtragem, análise e extração de características e exibição de resultados. Esse pacote é capaz de adquirir vídeos compatíveis (AVI e WMV) e imagens (BMP, PNG, TIF, JPEG) que podem ser processados por um conjunto de blocos (MATHWORKS, 2006).

4.3 Blocos utilizados

A seguir, são descritos os blocos próprios do Simulink utilizados na elaboração deste trabalho, mostrados na Figura 16.

Figura 16 - Blocos prontos utilizados nas simulações (imagem retirada da interface do Simulink).



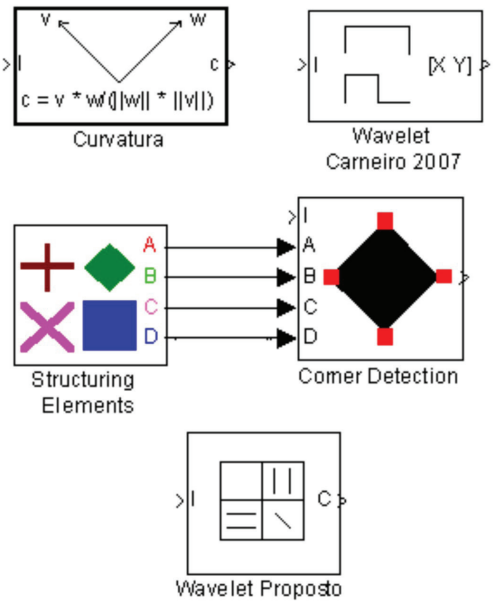
Fonte: Mathworks, 2009.

- **Image From File:** tem a função de ler uma imagem salva e disponibilizá-la para o sistema de simulação no formato de cores RGB.
- **Video Viewer:** tem a função de mostrar numa tela uma imagem ou *frame* passados para ele.
- **Draw Markers:** desenha marcadores coloridos numa imagem. No caso deste trabalho, ele é utilizado para desenhar marcadores nos pontos em que foram detectados vértices pelos algoritmos estudados.
- **Data Type Conversion:** realiza a conversão do formato dos dados.
- **Blob Analysis:** extrai características como área, perímetro, centro de massa, dentre outros, para todos os objetos presentes em imagens segmentadas.
- **Frame Rate Display:** responsável pela determinação da taxa de *frames* processados por segundo, o que é importante para avaliar a velocidade de processamento do algoritmo.

4.4 Blocos desenvolvidos

Durante a elaboração deste trabalho, os blocos já existentes no Simulink não são suficientes para suprir as necessidades da implementação. Dessa forma, foram criados blocos novos exclusivos para este trabalho, os quais são mostrados na Figura 17 e descritos logo em seguida.

Figura 17 - Blocos desenvolvidos para realização de testes neste trabalho.



Fonte: Mathworks, 2009.

- **Curvatura:** bloco responsável pela detecção de vértices através do cálculo da curvatura, utilizando equação 4. Essa função necessita de uma imagem binarizada para localizar os vértices.
- **Wavelet Carneiro 2007:** realiza a implementação do Método *Wavelet* proposto por Carneiro (2007).
- **Structuring Elements:** bloco contendo os quatro elementos estruturantes utilizados.
- **Corner Detection:** bloco que realiza o fechamento assimétrico em si a partir dos elementos estruturantes fornecidos pelo bloco "Structuring Elements".
- **Wavelet Proposto:** realiza a implementação do método de detecção de vértices proposto neste trabalho.

5 Resultados

5.1 Testes em imagens

Nesta seção, são mostrados os resultados experimentais obtidos utilizando as diferentes abordagens implementadas. Para fins de exibição dos resultados, os vértices são desenhados como círculos pretos, os objetos são mostrados em tons de cinza e o fundo é colorido de branco. Inicialmente, na Figura 18, é mostrado o resultado para a imagem de baixa complexidade (Figura 13).

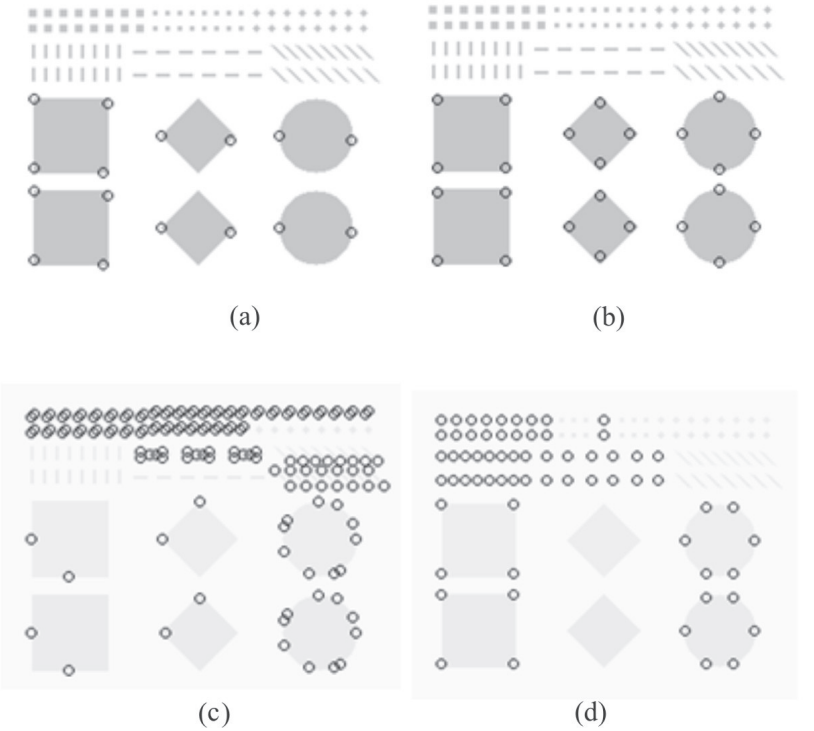
Figura 18 - Resultado da detecção de vértices para a imagem de baixa complexidade utilizando: a) curvatura, b) morfologia matemática, c) proposto por Carneiro (2007) e d) método proposto.

(a) (b) (c) (d)

Os algoritmos de Wavelet proposto por Carneiro (2007) e de curvatura, conforme ilustrado respectivamente nas Figuras 18(c) e 18(a), possuíram um bom desempenho, localizando os vértices mais importantes da forma, contudo, não detectando o vértice formado no encontro dos dois quadrados.

Já o algoritmo de fechamento assimétrico e o método Freitas-Costa possuem a melhor detecção, conforme ilustram as Figuras 18(b) e 18(d), localizando com perfeição os 8 vértices presentes na imagem.

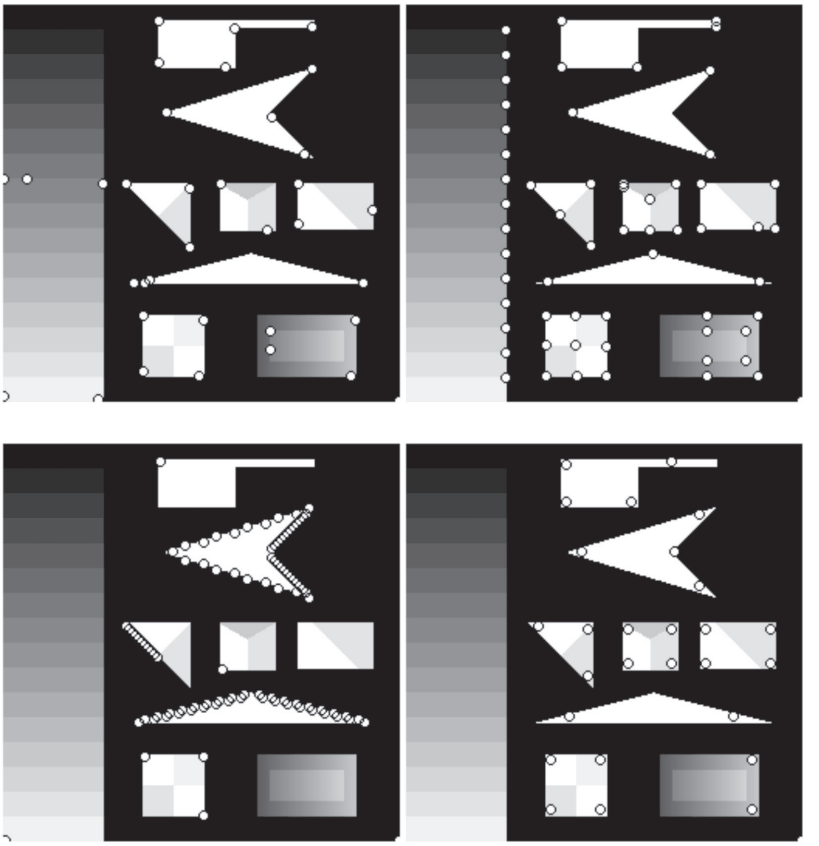
Figura 19 - Resultado da detecção de vértices para a imagem de média complexidade utilizando: a) curvatura, b) morfologia matemática, c) proposto por Carneiro (2007) e d) método proposto.



Para a segunda imagem de teste, o algoritmo de curvatura conseguiu identificar todos os vértices do quadrado, conforme ilustrado na Figura 19(a). Contudo, o método não consegue localizar os vértices inferiores e superiores do círculo e losango. Além disso, os vértices são localizados apenas nas formas de maior área. Isso é causado devido ao fato de o cálculo da curvatura necessitar que o contorno do objeto possua um número mínimo de pontos, de modo que seja possível encontrar os dois vetores utilizados para calcular a curvatura de contornos discretos, como explicado na seção 2.3.

Conforme ilustrado na Figura 19(b), o algoritmo de fechamento assimétrico conseguiu identificar os vértices dos maiores objetos presentes na imagem. Já o algoritmo proposto por Carneiro (2007) possuiu um resultado ruim, não localizando os dois vértices nos quadrados e os dois vértices no losango, e detectou vários vértices inexistentes no círculo, conforme mostrado na Figura 19(c). Na Figura 19(d), é ilustrado o resultado obtido pelo método Freitas-Costa. Este possui um desempenho intermediário entre o algoritmo de curvatura e o algoritmo de fechamento assimétrico, não localizando os vértices no losango, contudo, em aplicações nas quais a forma é conhecida, os parâmetros utilizados na detecção de vértices podem ser otimizados. Uma vantagem dos métodos de fechamento assimétrico e proposto em comparação ao algoritmo baseado em curvatura é que eles não necessitam de uma prévia segmentação, dessa forma, o último é dependente de segmentação, e caso esta não seja satisfatória, os vértices não serão localizados.

Figura 20 - Resultado da detecção de vértices para a imagem de alta complexidade utilizando: a) curvatura, b) morfologia matemática, *wavelets* c) proposto por Carneiro (2007) e d) método proposto.



Por fim, o resultado obtido para a imagem de alta complexidade é mostrado na Figura 20. Diferentemente das imagens de baixa e média complexidade, para essa imagem, os vértices localizados são desenhados como círculos brancos, com bordas pretas sobre a imagem original.

O resultado do algoritmo de curvatura é ilustrado na Figura 20(a). Este apresentou um bom resultado, localizando eficientemente a maior parte dos vértices presentes nos objetos mais claros, contudo, deixando de detectar vértices dentro das formas, bem como em regiões mais escuras. Isso ocorre devido à dependência da segmentação através da técnica de limiarização. O algoritmo de segmentação automática considera como fundo as regiões escuras e as claras como objeto, e sobre estas são detectados os vértices.

O resultado obtido pelo algoritmo de fechamento assimétrico é ilustrado na Figura 20(b). Esse método possui a melhor detecção e foi capaz de localizar eficientemente a maior parte dos vértices presentes, não localizando os vértices em regiões com baixo contraste e côncavas.

O resultado obtido pelo método proposto por Carneiro (2007) é ilustrado na Figura 20(c). Ele possui uma grande quantidade de falsas detecções, como nos objetos triangulares, e pequenas oscilações do contorno geraram a detecção de vários vértices, tendo também perdido vários vértices, como no objeto mais alto da figura.

O resultado obtido pelo método Freitas-Costa é mostrado na Figura 20(d). Ele apresentou uma boa detecção dos vértices, tendo conseguido identificar boa parte dos vértices presentes na imagem e nos mais diferentes tipos de objetos, exceto nos objetos com um contraste menor com o fundo.

5.2 Tempo de processamento

Todas as simulações realizadas neste trabalho foram feitas em um *notebook* DELL Precision M65, com processador Intel Core2Duo T7200, com 2GHz, 3Gb de Ram, DDR2 667, utilizando o sistema operacional Windows XP SP3 e o Matlab 2006b.

Na Tabela 1, é mostrada a quantidade de *frames* por segundo de cada algoritmo estudado neste trabalho e para cada imagem utilizada.

Tabela 1: Taxa de *frames* por segundo para os algoritmos estudados.

	Nome	**Simples**	**Média**	**Alta**
Imagem	Tamanho	128x128	256x256	256x256
	Número de obj. relevantes	1	6	8
Método	Curvatura	21,3679	2,0916	2,0781
	Morfologia	29,7063	9,5510	9,6899
	Carneiro (2007)	26,6667	7,2727	6,8073
	Proposto	31,9477	9,2765	8,0373

Para o caso da imagem de baixa complexidade, o algoritmo que apresenta melhor performance é o algoritmo Freitas-Costa, com uma taxa de processamento de quase 32 *frames* por segundo (fps), mas seguido bem de perto pelo algoritmo de morfologia (taxa de 29,7 fps) e pelo algoritmo proposto por Carneiro (2007) (taxa de 26 fps). O algoritmo de curvatura apresentou um resultado bem inferior aos dos outros, com uma taxa de processamento de apenas 21,3 fps.

Para a imagem de complexidade média, o algoritmo de morfologia passa a ser o mais rápido, com uma taxa de 9,5 fps, quase igual à do algoritmo Freitas-Costa, que foi de 9,2 fps. O algoritmo Wavelet de Carneiro (2007) fica um pouco aquém, com uma taxa de 7,2 fps, enquanto o algoritmo de Curvatura apresentou novamente um resultado bastante inferior aos dos outros, com uma taxa de processamento de 2,09 fps.

Para a imagem de maior complexidade, o algoritmo de morfologia é novamente o mais rápido, apresentando uma taxa de processamento de 9,6 fps, ou seja, levemente maior que a taxa da imagem de complexidade média. O algoritmo Freitas-Costa apresenta uma leve redução de processamento em relação ao teste anterior, ficando com uma taxa de aproximadamente 8 fps. O algoritmo de Carneiro (2007) também apresenta uma diminuição de processamento (6,8 fps) e o algoritmo de curvatura mantém praticamente o mesmo resultado do teste anterior, ou seja, uma taxa de processamento de aproximadamente 2 fps.

6 Conclusão

Neste trabalho, é apresentado o estudo de métodos tradicionais da literatura de processamento de imagens para detecção de vértices e é proposto um algoritmo para localização de pontos críticos baseados na Transformada Wavelet.

Os resultados mostram que o algoritmo de detecção de vértices por curvatura é complexo computacionalmente, podendo ser implementado em sistemas em tempo real, limitando-se ao maior objeto presente na imagem. A detecção é dependente da segmentação através da técnica de limiarização e, para possuir um bom desempenho, é necessário o conhecimento prévio do objeto a ser segmentado.

Além disso, os resultados mostram que o método proposto por Carneiro (2007) é capaz de localizar os vértices apenas na imagem de baixa complexidade, resultando em uma grande quantidade de falsas detecções.

A modificação proposta por este trabalho foi capaz de melhorar o resultado do método proposto por Carneiro (2007), possuindo um bom desempenho para as diversas imagens estudadas, mas não conseguiu resultados melhores que o algoritmo de fechamento assimétrico.

Este possui a melhor detecção de vértices, conseguindo localizar com a maior eficiência do que os outros métodos estudados.

Em termos de complexidade computacional, o método proposto possui um bom desempenho computacional, enquanto o método de fechamento assimétrico possui uma variação grande quanto ao tamanho dos elementos estruturantes utilizados.

Dessa forma, o método proposto associa uma boa detecção de vértices e uma baixa complexidade computacional, sendo útil para aplicações em tempo real, tais como rastreamento de objetos, reconhecimento de movimento, navegação robótica, dentre outras.

Agradecimentos

Os autores gostariam de agradecer à Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP), à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro e ao Laboratório de Engenharia de Sistemas de Computação pelo fornecimento das condições materiais necessárias para o desenvolvimento deste trabalho.

Referências

CARNEIRO, Alex Torquato Souza. *Identificação e localização de pontos críticos em imagens digitais*. 2007. 65f. Trabalho de Conclusão de Curso (Graduação)-Universidade Federal do Ceará, Fortaleza, 2007. Disponível em: <http://www.cgeti.ufc.br/monografias/ALEX_TORQUATO_SOUZA_CARNEIRO.pdf>. Acesso em: 24 jan. 2010.

COSTA, Luciano da Fontoura; CÉSAR JÚNIOR, Roberto Marcondes. *Shape Analysis and classification:* theory and practice. Boca Raton: CRC Press, 2001. ISBN: 0-8493-3493-4.

COSTA, Rodrigo Carvalho Souza. *Inspeção automática de laranjas destinadas à produção de suco, utilizando técnicas de processamento digital de imagens*. 2006. 91f. Trabalho de Conclusão de Curso (Graduação em Tecnologia Mecatrônica)-Centro Federal de Educação Tecnológica do Ceará, Fortaleza, 2006. Disponível em: <http://www.academia.edu/3128419/Inspecao_Automatica_de_Laranjas_Destinadas_a_Producao_de_Suco_Utilizando_Tecnicas_de_Processamento_Digital_de_Imagens>. Acesso em: 03 fev. 2012.

GAO, Xinting et al. Multiscale contour corner detection based on local natural scale and wavelet transform. *Image and Vision Computing*, Newton, v. 25, n. 6, p. 890-898, 2006.

GONZALEZ, Rafael C.; WOODS, Richard E. *Digital image processing*. New Jersey: Pearson Prentice Hall, 2008. ISBN: 0-1316-8728-X.

JOHNSTON, E.; ROSENFELD, A. Angle detection on digital curves. *IEEE Trans. Comp*. Washington, v. C22, n. 7, p. 875-878, 1973.

KALAGASIDS, A. S. et al. The International Building Physics Toolbox in Simulink. *Energy and Buildings*, Amsterdam, v. 39, p. 665-674, 2007.

KITCHEN, L. ROSENFELD, A. Gray level corner detection. *Pattern Recognition Letters*, Amsterdam, v. 1, n. 2, p. 95-102, 1982.

LAGANIÈRE, Robert. Morphological corner detection. In: *Proc. International Conference in Computer Vision*, 1998. New Delhi: Narosa Publishing House, 1998. pp. 280-285. Disponível em: < http://www.site.uottawa.ca/~laganier/publications/coin.pdf>.

MASOOD, Asif; SARFRAZ, M. Corner detection by sliding rectangles along planar curves. *Computers Graphics-UK*, Amsterdam, v. 31, n. 3, p. 440-448, 2007.

MATHWORKS. *Video and Image Processing Blockset*: Userguide. Natick, MA: 3 Apple Hill Drive, 2006. Disponível em: <http://www.mathworks.com/products/computer-vision/index.html>. Acesso em: 14 fev. 2010.

MATWHORKS. MATLAB/SIMULINK. Disponível em: <http://www.mathworks.com>. Acesso em: 15 jan. 2009.

PARKS, Donovan; GRAVEL, Jean-Philippe. *Corner detectors*. 2008. Disponível em: <http://kiwi.cs.dal.ca/~dparks/CornerDetection/index.htm>. Acesso em: 15 jan. 2009.

PASSARINHO, Cornélia Janayna P. et al. Multiscale curvature assessment of postural deviations. *Lecture Notes in Computer Science*, Amsterdam, v. 3138, n. 1, p. 296-303, 2004.

PAULA JÚNIOR, Iális Cavalcante de. *Abordagem wavelet para detecção de cantos em formas*. Dissertação (Mestrado em Engenharia de Teleinformática) – Fortaleza, Universidade Federal do Ceará, 2007.

PAULA JÚNIOR, Ialis C. et al. Multiscale Corner Detection in Planar Shapes. *Journal of Mathematical Imaging and Vision*, New York, v. 45, n. 3, p. 251-263, 2013.

ROSTEN, Edward; DRUMMOND, Tom. Machine Learning for High-Speed Corner Detection. *Lecture Notes in Computer Science*, Berlin, v. 3951, p. 430 – 443, 2006.

SHUI, Peng-Lang; ZHANG, Wei-Chuan. Corner Detection and Classification Using Anisotropic Directional Derivative Representations. *IEEE Transactions on Image Processing,* Charlottesville, v. 22, n. 8, p. 3204-3218, 2013.

TAYLOR, C. J. et al. Macroscopic Traffic Flow Modelling and Ramp Metering Control Using Matlab/Simulink. Environmental Modelling & Software, Amsterdam, v. 19, n., p. 975-988, 2004.

XU, Wanying et al. A Biologically Motivated Corner Detection Method Based on the Oriented Receptive Fields of Simple Cortical Cells. 2010 International Conference on Biomedical Engineering and Computer Science (ICBECS), 2010, Wuhan. *Proceedings*… Wuhan: IEEE, 2010. p. 1-4.

ZHANG, Yong; DONGSHENG, Ji. Adaptive Harris Corner Detection Algorithm Based on B-spline Function. SECOND INTERNATIONAL CONFERENCE ON INTELLIGENT HUMAN-MACHINE SYSTEMS AND CYBERNETICS, 2, 2010, Jiangsu. *Proceedings*... Jiangsu: IHMSC, 2010. p. 69-72.

Sobre os autores

Rodrigo Fernandes Freitas
Tecnólogo em Mecatrônica Industrial pelo Instituto Federal de Educação, Ciência e Tecnologia do Ceará – IFCE. Mestre em Engenharia, área de Engenharia de Teleinformática, pela Universidade Federal do Ceará – UFC.

Rodrigo Carvalho Souza Costa
Engenheiro eletricista pela Universidade Federal do Ceará – UFC. Tecnólogo em Mecatrônica Industrial pelo Instituto Federal de Educação, Ciência e Tecnologia do Ceará – IFCE. Mestre em Engenharia, área de Engenharia de Teleinformática, pela Universidade Federal do Ceará – UFC. Doutorando em Engenharia, área de Engenharia de Teleinformática, pela Universidade Federal do Ceará – UFC.

Paulo César Cortez
Engenheiro eletricista pela Universidade Federal do Ceará – UFC. Mestre em Engenharia, área de Engenharia Elétrica, pela Universidade Federal da Paraíba, Campus II - Campina Grande-PB, atualmente Universidade Federal de Campina Grande - UFCG. Doutor em Engenharia, área de Engenharia Elétrica, pela Universidade Federal da Paraíba, Campus II - Campina Grande-PB, Universidade Federal de Campina Grande - UFCG. Professor associado II do Departamento de Engenharia de Teleinformática da Universidade Federal do Ceará – UFC.