Recorte circular de cenas computacionais aplicado na separação de íris humana

Acrísio J. do Nascimento Jr.
acrisio@catalao.ufg.br
Departamento de Ciências da Computação - UFG - CaC, 75705-220, Catalão, GO, Brasil
Alex F. de Araújo
Aledir S. Pereira
Norian Marranghello
Jonathan Rogéri
fa.alex@gmail.com
aledir@ibilce.unesp.br
norian@ibilce.unesp.br
jonrogeri@hotmail.com
Departamento de Ciências da Computação e Estatística - IBILCE - UNESP,
15054-000, São José do Rio Preto, SP, Brasil
C. A. M. Barbosa
barbosa@camb.eng.br
João Manuel R. S. Tavares
tavares@fe.up.pt
Departamento de Engenharia Mecânica (DEMec), Faculdade de Engenharia da Universidade do Porto (FEUP) / Instituto de Engenharia Mecânica e Gestão Industrial (INEGI),
Porto, Portugal

Resumo. *O reconhecimento de padrões da íris humana é um processo não invasivo e que permite a extração de características individuais. A localização e separação da íris do restante da imagem é fundamental para o bom desempenho dos métodos desenvolvidos nesta área. O isolamento da íris costuma ser um processo que exige a varredura da imagem e a comparação da localização de cada um de seus pixels. Como esta possui formato arredondado, o recorte de cenas computacionais, com janela de corte circular, pode ser aplicado nas imagens de íris para fazer este isolamento. No entanto, encontrar os limites de uma janela de visualização é um processo exigente computacionalmente, principalmente para estuturas de dados do tipo bitmap, que devem ser processadas pixel a pixel. Na tentativa de reduzir o processamento para tornar viável a aplicação na separação de íris, este trabalho usa os conceitos de hashing e a equação da circunferência para definir os pontos da imagem original bitmap pertencentes a uma área de visualização circular, sem a necessidade de percorrer todos os pixels da mesma. Apresenta-se o resultado da aplicação deste método para a separação da íris humana em imagens bitmap, sem a necessidade de comparar a localização de todos os seus pixels.*

Palavras-chave: *Clipping, recorte, reconhecimento por íris*

1. Introdução

O reconhecimento de padrões da íris humana vem se destacando como uma das principais técnicas de biometria na atualidade, principalmente por ser um processo não invasivo e pelo fato da íris apresentar uma grande quantidade de características únicas de cada pessoa.

Nas pesquisas atuais pode-se encontrar técnicas para obtenção de resultados significativos de reconhecimento. Entretanto, quase todas seguem o mesmo padrão de processos: obtenção da imagem, localização da íris, normalização da imagem, extração de características e comparação. Um problema comum durante o processo de localização é a separação da íris do restante da imagem. Neste estudo, foi utilizado o *clipping* (ou recorte) com janela de visualização circular para remover os pixels externos à região da íris. O *clipping* é uma técnica aplicada, principalmente durante o *pipeline* de visualização gráfica [Foley et al. (1990)], e possui a função de eliminar regiões de uma cena que não fazem parte da área de interesse (ou área visível). Por este processo, consegue-se criar uma nova imagem onde as regiões externas à íris são transformadas na cor branca deixando a íris em evidência.

Encontrar os limites de uma *viewport* é um dos processos mais exigente computacionalmente do estágio de *clipping* do *pipeline* de visualização gráfica [Wu et al. (2006)], [Zhang and Sabharwal (2002)]. O processamento torna-se mais exigente quando aplicado a imagens matriciais (como do tipo *bitmap*). Neste caso, cada pixel deve ser analisado individualmente, comparando sua localização em relação à janela de corte.

Seguindo os conceitos da estrutura de busca *hashing*, pode-se usar a equação da circunferência para montar um vetor, chamado aqui de vetor de limites, para encontrar o contorno de um plano de recorte circular, evitando comparar os pixels inválidos e aumentado assim a eficiência do método de processamento da cena.

A estrutura *hashing* possui uma abordagem de busca diferente das listas, filas e árvores. Ao contrário do que ocorre nas demais estruturas, onde a busca por um determinado elemento é feita através de um percurso, comparando cada elemento visitado, no *hashing* o acesso ao conteúdo ocorre por meio de uma chave única a qual é obtida através de uma função matemática chamada função *hashing*. A busca ocorre através do uso de tabelas de indexaçãao, onde cada posição é associada a uma chave. Desta forma, ao conhecer as chaves torna-se possível acessar as posições desejadas na tabela sem precisar percorrê-la [Drozdek (2000)].

Foram estudados alguns métodos para *clipping* de cenas e, na tentativa de obter um procedimento com menos processamento, foi aplicado a equação da circunferência como função *hashing* para definir o contorno da *viewport* circular aplicada no recorte de imagens de íris humana, para redução do conjunto de pontos processados para extrair as características destas.

Este artigo está organizado da seguinte forma: a seção a seguir descreve os conceitos básicos das técnicas e ferramentas usadas para recorte em cenários virtuais e separação da íris do restante da imagem original. A seção 3 apresenta a formulação do problema, abordando o recorte circular e a descrição do método de recorte proposto, bem como algumas restrições que devem ser observadas para manter válida a definição de *hashing* e a seção 4 descreve os testes realizados e alguns resultados obtidos. As conclusões são apresentadas na seção 5.

2. Conceitos básicos

O *clipping* é aplicado em Computação Gráfica para retirar linhas e superfícies que localizam-se fora da área de visível [Hearn and Baker (1986)]. Esta técnica tem por objetivo acelerar a representação da cena, encontrando os vértices que interceptam o contorno do plano de recorte que limitam as superfícies e eliminando todos os pontos e superfícies que localizam-se fora da região visível.

O *clipping* também pode ser utilizado em conjunto com o sistema de câmera para ampliar uma porção específica da cena. O método responsável por esta seleção e ampliação é chamado de *windowing* [Harrington (1987)]. Muitos algoritmos de recorte foram desenvolvidos considerando *viewport* retangulares como em [Sutherland and Hodgman (1974)], [Greiner and Hormann (1998)] e [Zhang and Sabharwal (2002)]. Na seção a seguir apresenta-se alguns destes métodos.

2.1 Recorte de cenas computacionais

Sutherland foi o pioneiro no desenvolvimento de *clipping* de cenas e, juntamente com Hodgman propuseram um método para recorte de superfície utilizando uma *viewport* retangular [Sutherland and Hodgman (1974)]. Assim, desenvolveram um algoritmo composto por quatro etapas. Em cada etapa, considera-se uma lado da janela de corte retangular, e efetua-se o recorte dos objetos da cena em relação a este lado, de modo que ao terminar de executar estes passos, apenas as fatias visíveis dos polígonos são exibidas. Para isto, são feitos testes de localização considerando os vértices dos polígonos e o contorno da janela de recorte para garantir que apenas as regiões externas sejam descartadas [Hearn and Baker (1986)].

Diferente de Sutherland e Hodgman, Weiler e Atherton não abordam a superfície como sendo um conjunto sequencial de vértices. Para estes autores a cena é formada por uma série de polígonos e para determinar quais partes são visíveis, sendo efetuadas divisões recursivas da cena [Foley et al. (1990)]. Este algoritmo é um dos mais gerais já desenvolvidos para o *clipping* de superfícies e pode ser utilizado para recorte tanto de polígonos convexos quanto côncavos [Hearn and Baker (1986)]. Sua idéia básica é determinar a visibilidade de um polígono em relação a outro através das operações de interseção, união e diferença (operações de conjuntos).

Um método de *clipping* que considera a *viewport* circular foi proposto por Wu, Huang e Han [Wu et al. (2006)]. Este método efetua o recorte de segmentos de parábola fazendo uso das características geométricas destes segmentos e de sua localização em relação à janela de recorte. Tomando *x* e *y* como as coordenadas dos pontos pertencentes à circunferência de raio *r*, e considerando as equações da circunferência com centro na origem ($x^2 + y^2 = r^2$) e da parábola ($y = ax^2 + bx + c$, com a $\neq$ 0), bem como o segmento da parábola localizado no intervalo [p,q], com (p $<$ q), o algoritmo faz um pré-processamento eliminando todos os segmentos de parábolas totalmente externos à *viewport*, usando as condições presentes na Tabela 1, e a seguir prossegue com os cálculos das interseções.

Condição	**Localização da parábola**
c - b^2 / 4 a $\geq$ r	Totalmente acima da *viewport*
c - b^2 / 4 a $\leq$ -r	Totalmente abaixo da *viewport*
q $\leq$ -r	Totalmente à esquerda da *viewport*
q $\geq$ r	Totalmente à direita da *viewport*

Tabela 1: Condições para o pré-processamento do método proposto por Wu, Huang e Han, adaptado de [Wu et al. (2006)].

Com a etapa do pré-processamento, esta abordagem economiza o cálculo de interseções para os segmentos totalmente invisíveis. Este método é eficiente para recorte de segmentos de parábolas usando *viewport* circulares e também usando planos de corte retangulares, mas ainda não consegue trabalhar com recorte de polígonos.

2.2 Processo de localização da íris

O processo de identificação da íris acontece após a aquisição da imagem do olho. Ele é responsável por localizar a íris dentro da imagem do olho em questão. A região da íris pode ser representada por dois círculos, um que tem a fronteira com a esclerótica e outro que divide a íris com a pupila. Normalmente são utilizados operadores para detecção de círculos, onde os mais conhecidos são o operador íntegro-diferencial de Daugman e a transformada de Hough [Daugman (2004)].

O operador íntegro-diferencial, dado na equação (1) representa o valor do pixel de coordenadas *x,y* da imagem de um olho. Neste operador procura-se sobre o domínio *(x,y)* da imagem pelo valor máximo da derivada parcial em relação ao raio *r*, da integral normalizada do contorno da imagem ao longo de um arco circular ∂s de raio *r* e coordenadas do centro (x_0, y_0) [Daugman (2004)]:

$$max \left| G_\sigma(r) \frac{\partial}{\partial r} \oint_{r,x_0,y_0} \frac{I(x,y)}{2\pi r} ds \right| . \tag{1}$$

A transformada de Hough é um método padrão para identificar formas que possuem fórmulas definidas, tais como círculos em imagens digitalizadas. Essa transformada define um mapeamento entre o espaço da imagem *(x,y)* e o espaço de parâmetros *(c, d, r)*. Na equação 4, os termos *c* e *d* formam o centro da circunferência e *r* o raio da mesma. Para esse mapeamento, o espaço de parâmetros é discretizado e representado na forma de uma matriz de inteiros, onde cada posição da matriz corresponde a um intervalo no espaço real de parâmetros. Procuram-se todos os círculos que passam pelo ponto fixo. Esta equação traduz um 'cone' no espaço que é fixado pelos parâmetros. Deve-se então acumular todos esses cones no espaço tridimensional e buscar um pico máximo da acumulação. Se o acúmulo na célula correspondente é alto, então a célula é escolhida. Como exemplo, na Figura 1 tem-se em (a) diversas circunferências de vários tamanhos, onde procura-se detectar os círculos de raio igual a 20. Em (b) tem-se uma matriz acumuladora, sendo que a maior intensidade de incidência corresponde à circunferência de raio 20.

$$(x - c^2) + (y - d^2) = r^2 \tag{2}$$

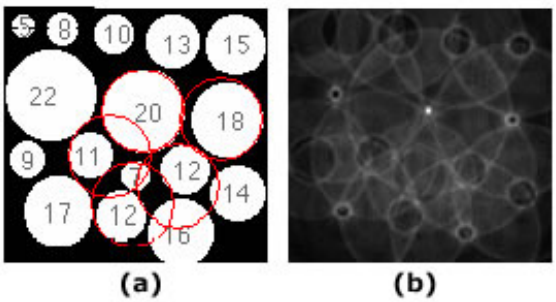

Figura 1: Exemplo da aplicação da transformada de Hough, adaptada de [Chavez (2007)].

2.3 Filtros morfológicos

Morfologia matemática compreende a área que estuda propriedades topológicas e estruturais dos objetos a partir de uma imagem, tendo como objetivo descrever quantitativamente as

estruturas geométricas. Funciona como uma técnica na concepção de algoritmos de processamento digital de imagens, contendo ferramentas básicas como detectores de bordas e filtros morfológicos. Os filtros morfológicos exploram as propriedades geométricas dos sinais e suas máscaras são denominadas de elementos estruturantes. Estes elementos devem apresentar valores *0* (zero) ou *1* (um), de modo a considerar ou não,o pixel correspondente à posição da matriz [Gonzalez et al. (2003)].

2.4 Conversão de sistema de cores

Dentro da área de processamento de imagens pode-se destacar vários sistema de cores, onde o mais conhecido é o RGB (*Red*, *Green* e *Blue*). Neste modelo cada cor é formada a partir das componentes espectrais primárias de vermelho, verde e azul. Este é baseado em um sistema de coordenadas cartesianas, mostrado na Figura 2. Pode-se observar as coordenadas das cores vermelho, verde, azul, ciano, magenta e amarelo. O preto está na origem *(0,0,0)* e o branco em *(1,1,1)*, e na diagonal do cubo ligando estes pontos temos o degradê de níveis de cinza. As imagens no modelo RGB consistem de três planos planos de imagens independentes, cada um associado a uma cor primária.

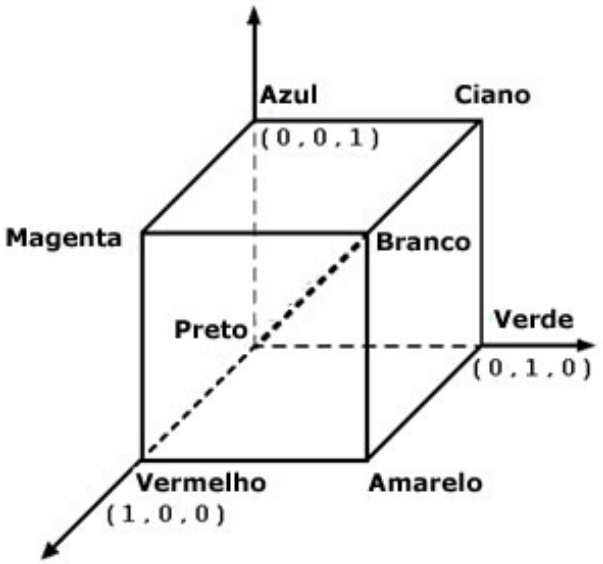

Figura 2: Espaço de cores RGB.

Além dos sistemas que tratam cores, pode-se realizar a conversão da imagem em tons de cinza. Uma imagem em níveis de cinza corresponde a uma representação onde cada pixel assume um valor em um intervalo entre *0* (zero) e *255*, onde o valor *0* (zero) é associado ao preto, o valor *255* corresponde ao branco, e os valores intermediários constituem os tons de cinza. Neste trabalho, realizou-se a transformação da imagem originalmente em RGB para uma imagem em níveis de cinza, onde é levada em consideração a intensidade das três camadas do modelo RGB para a formação da intensidade do pixel na imagem em níveis de cinza.

2.5 Equalização de histograma

O histograma é a representação da imagem em um gráfico, onde o eixo horizontal apresenta valores de níveis de cinza e o eixo vertical indica a quantidade de pixels em cada um dos níveis de cinza desta imagem. A equalização de um histograma é a distribuição da quantidade de pixels em um nível de cinza. Essa técnica é utilizada para se obter alguns resultados como

clarear ou escurecer uma imagem e aumentar o contraste entre os seus níveis de cinza.

2.6 Operadores de gradiente

Operadores de gradientes são os filtros utilizados em processamento de imagens digitais para identificação de bordas em imagens. Estes retornam as bordas de acordo com mudanças na propriedade física ou espacial de superfícies iluminadas. Há vários métodos baseados nestes operadores, como Canny, Roberts e Sobel [Gonzalez et al. (2003)]. Esses algoritmos permitem separar os objetos do fundo, possibilitando a caracterização do formato de objetos, sua área, dentre outros.

3. Formulação do problema

3.1 O recorte circular

O problema do *clipping* de cenas computacionais usando *viewport* circular, pode ser dividido em três principais categorias: recorte de linhas (ou curvas), recorte de superfícies e recorte de imagens matriciais. Esta última categoria, é mais usada no processamento de imagens, onde as cenas são manipuladas na forma de matriz, com cada posição desta representando um pixel da imagem [Gonzalez et al. (2003)].

O recorte de linhas (ou curvas) consiste em determinar os pontos de interseção existentes entre elas e a circunferência, considerando apenas o segmento que liga estes pontos visível. Na Figura 3 apresenta-se um exemplo de *clipping* aplicado em uma cena formada por semi-retas, onde na coluna (a) tem-se a cena original, com as linhas externas e internas à *viewport* exibidas e na coluna (b) pode-se ver a mesma cena apenas com os segmentos internos processados.

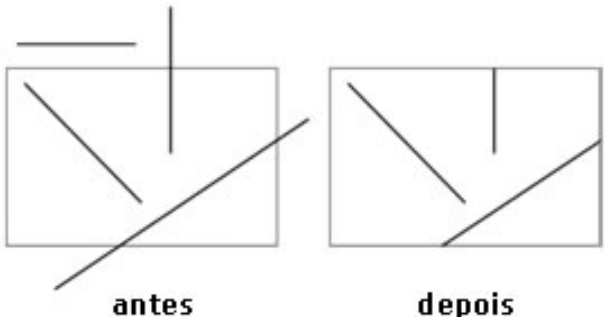

Figura 3: Exemplo de recorte de linhas.

Uma superfície (ou polígono) pode ser divida em uma série de segmentos de retas e curvas. Portanto, o *clipping* de superfícies pode ser obtido a partir da aplicação do recorte de linhas em cada aresta que pertence ao polígono [Sutherland and Hodgman (1974)]. Para as imagens matriciais, isso pode ser feito de duas formas. Uma é através da segmentação da cena e extração dos objetos, aplicando então o *clipping* de superfícies e na sequêcia um algoritmo de preenchimento para reestabelecer as cores e texturas dos objetos visíveis. Para evitar o uso dos processos de preenchimento e de segmentação (que é complexo), outra forma de recortar cenas matriciais é a partir da comparação da distância entre seus pontos e o centro da *viewport*.

3.2 Uso dos conceitos de *hashing*

Na tentativa de reduzir o custo computacional para buscar os pontos de uma imagem que são internos a uma janela de visualização circular, neste trabalho adota-se o conceito de *hashing*. Esta estrutura caracteriza-se por permitir a busca de um conjunto de dados a partir de um conjunto de chaves, usando uma função matemática, evitando assim a execução de operações de

comparação. Assim, tomando o conjunto de valores inteiros entre *0* (zero) e *r* (com *r* sendo o raio da *viewport* circular) como as chaves de busca, a equação da circunferência de centro no ponto *(a, b)*, retorna dois valores iguais em módulo para cada chave. Considerando apenas o módulo destes valores, tem-se uma função *hashing*, chamada aqui de *h*:

$$(x - a^2) + (y - b^2) = r^2, \tag{3}$$

aplicando raiz quadrada em ambos os lados da equação tem-se:

$$\begin{aligned} (y - b)^2 &= r^2 + (x - a)^2, \\ y &= b + \sqrt{r^2 - (x - a)^2}, \\ h(x) &= b + \sqrt{r^2 - (x - a)^2}. \end{aligned} \tag{4}$$

com $0 \leq x \geq r$.

A equação *h(x)* define o intervalo polar de *0*$\sim\pi$ e aplicando-a, obtem-se uma sequência de dados armazenados em um vetor *v* (com *x* elementos), chamado de vetor de limites, que é usado para definir os pontos da cena que pertencem à região visível no primeiro quadrante do sistema de coordenadas.

Usando a propriedade de que a circunferência é um polégono simétrico, tem-se para cada coordenada *x*, duas ordenadas *(y e -y)* e para cada *x* positivo tem-se seu espelho do lado negativo, ou seja, para todo *x* maior que *0* (zero) e menor que *r*, existe um *-x* menor que *0* (zero) e maior que *-r*.

Portanto, espelhando-se o primeiro quadrante em relação ao eixo *x*, obtém-se o quarto quadrante e, espelhando estes dois em relação a *y*, tem-se a circunferência completa. Desta forma, tem-se o diagrama de fluxo do algoritmo de recorte circular na Figura 4. Após a criação do vetor de limites no passo 1 do algoritmo, o quadrante 1 da janela de visualização é preenchido usando os pontos, cujas ordenadas pertencem aos vetores auxiliares y_i, com $0 \leq i < r$. Estes vetores são preenchidos com os valores inteiros do intervalo de *0* (zero) a *h(*x_i*)*. Assim, para cada *i* encontra-se, sem a necessidade de efetuar comparações, uma linha vertical limitada pelo contorno da janela de corte. Os passos 3 e 4 do algoritmo, efetuam os espelhamentos em torno do eixo *x* e *y*, respectivamente. Para este processamento, o centro da *viewport* e da íris se coincidem. Portanto, as coordenadas dos pontos que formam os quatro quadrantes visíveis da circunferência podem ser usadas para localizar a íris na imagem original.

O método de recorte adotado recebe como entradas o raio da *viewport* e as coordenadas do centro desta. Usando o raio da janela de corte, o vetor de limites é montado definindo uma aproximação do contorno da janela de corte. No exemplo da Figura 5, a circunferência que contorna a *viewport* possui raio igual a *7*. Portanto, a região pertencente ao quadrante 1 é formada por *7* vetores auxiliares (y_i com $0 \leq i < r$, representados em níveis de cinza), na vertical, cujos tamanhos são definidos por *v*.

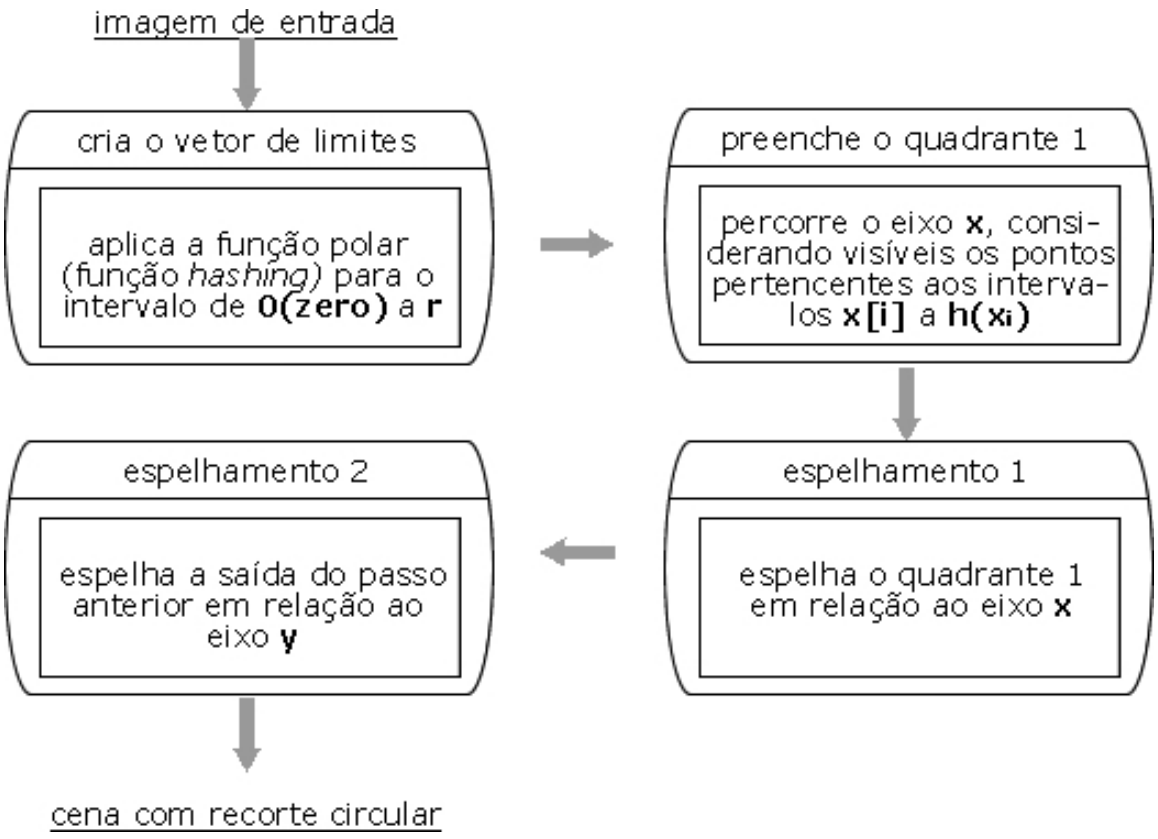

Figura 4: Diagrama de fluxo do método de recorte circular proposto.

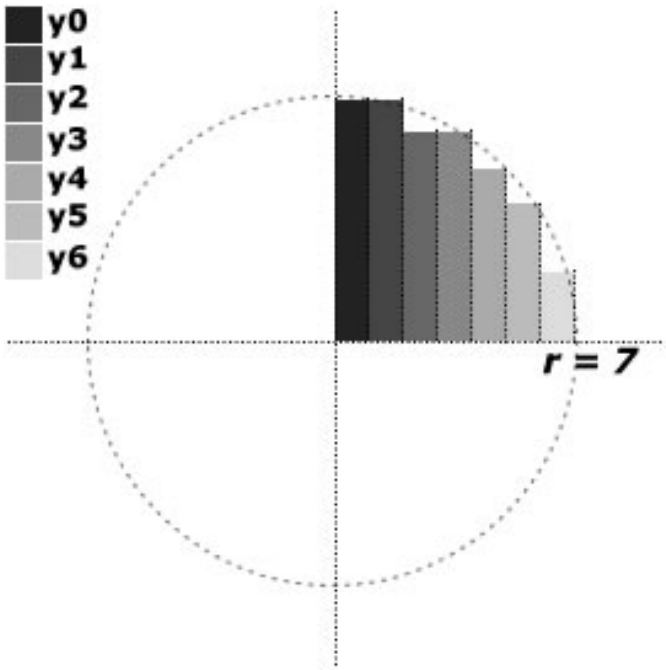

Figura 5: Exemplo do preenchimento do quadrante 1 pelos vetores y_i.

4. Testes realizados

Para a realização dos testes neste trabalho foi utilizado o banco de imagem UBIRIS (disponibilizado pelo Departmento de Ciência da Computação, Universidade de Beira Interior,

Covilhã - Portugal). Este banco é formado por imagens de íris humanas e foi escolhido por apresentar imagens reais, coloridas, com reflexos e oclusão.

Para realização do processo de localização da íris, inicialmente a imagem passa por uma conversão do sistema de cores RGB para tons de cinza, fazendo a equalização do histograma para aumentar o contraste entre a íris e os demais componentes do olho. Após essa etapa, deve ser aplicado o método de detecção de bordas em imagens, para localização da borda externa da íris, que faz fronteira com a esclerótica. Nos testes realizados, o operador de Canny [Canny (1986)] foi o que apresentou os melhores resultados para esta localização quando comparado com os demais operadores de gradiente. Os resultados mais expressivos foram conseguidos usando a média da intensidade da imagem para o valor do limiar e sigma igual a 0.6. Após a aplicação do operador de Canny, utilizou-se a transformada circular de Hough [Gonzalez et al. (2003)], para identificar a circunferência a partir dos traços de bordas conseguidos no processo anterior, como indicado pela Figura 6.

Localizada a fronteira externa da íris, o passo seguinte é a identificação da borda junto à pupila. Para esse processo, são necessários alguns métodos diferentes, já que em muitos casos a íris escura possui pouco contraste com a pupila. Devido a esse baixo contraste e o reflexo causado pelo flash na aquisição das imagens, o operador de Canny não apresentou bons resultados para localização da pupila. Para solucionar esse problema foi utilizada uma sequência de filtros morfológicos (abertura e remoção), em uma região central da imagem limiarizada, de dimensão 70x70 pixels, uma vez que a pupila fica nesta região em todas as imagens utilizadas.

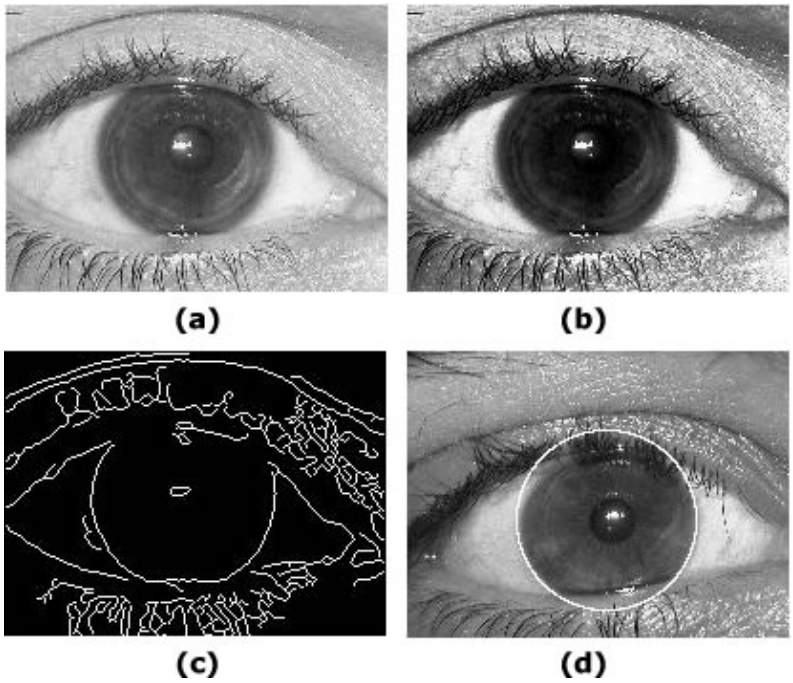

Figura 6: Localização da borda externa da íris: a) imagem original em tons de cinza; b) imagem equalizada; c) bordas detectadas pelo método de Canny; d) fronteira externa localizada.

Inicialmente, os valores inferiores a um limiar, definido manualmente com base nos níveis de cor para cada imagem, foram transformados em branco, para aumentar o constraste da pupila. Após essa fase, foram utilizados os filtros morfológicos de abertura e remoção, e então novamente a transformada de Hough, para encontrar agora uma aproximação o círculo referente à pupila, conforme apresentado na Figura 7.

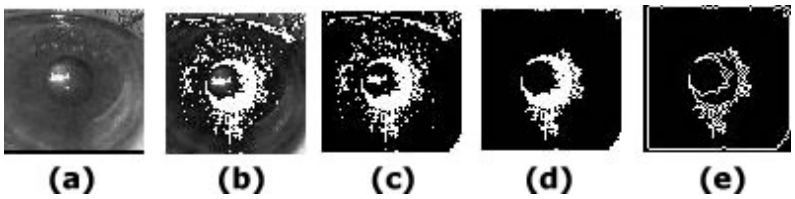

Figura 7: Sequência para reconhecimento da pupila: a) quadrado central da imagem; b) imagem equalizada; c) imagem binarizada; d) imagem após filtro de abertura; e) imagem após filtro de remoção.

O próximo passo para o reconhecimento da íris é a normalização da imagem, onde deve-se transformar a imagem da íris localizada em um retângulo, convertendo-a do plano cartesiano $I(x,y)$ para o plano polar $I(r,\theta)$. Esse procedimento visa diminuir a interferência de oclusões e reflexos na íris.

Para um melhor desempenho da normalização, é necessário isolar a imagem da íris localizada. Com o processamento realizado até este estágio, tem-se uma aproximação das fronteiras da íris e da pupila, suficientes para definir o ponto central da pupila e o raio aproximado das duas circunferências que delimitam a íris. Com esses dois parâmetros é possível utilizar o método de recorte circular usando a função *hashing*, descrito na seção 3, para separar a região da íris do restante da imagem. Foram descartados os pontos externos à janela de visualização circular. Da mesma forma, para remoção da pupila, o mesmo processo foi aplicado, usando o raio da circunferência menor, desta vez removendo da imagem todos os pontos internos à janela de visualização. Obtendo-se apenas a região que contém a íris da imagem, como vê-se nas Figuras 8 e 9, onde as regiões que não pertencem à íris aparecem na cor branca na coluna (b).

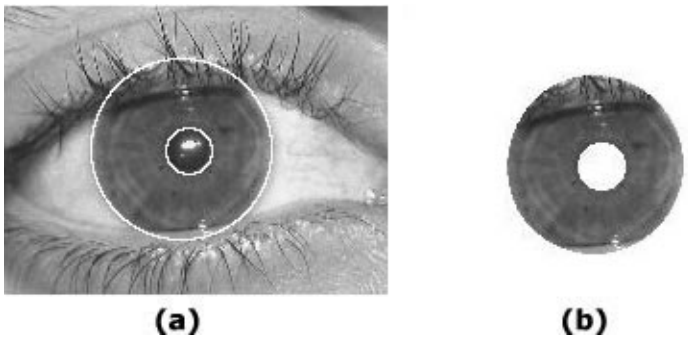

Figura 8: Localização da íris: a) íris localizada; b) separação da íris usando o método de *clipping*.

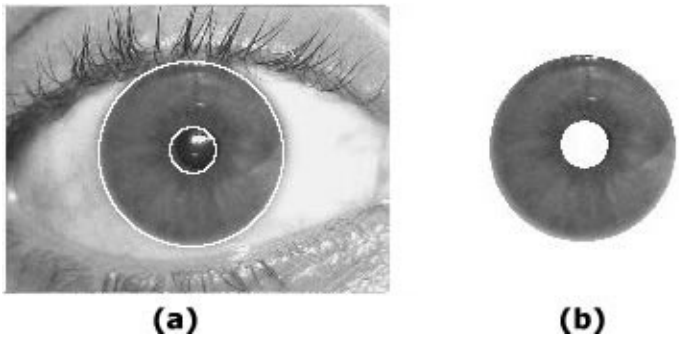

Figura 9: Localização da íris: a) íris localizada; b) separação da íris usando o método de *clipping*.

5. Conclusões

O método de recorte usando os conceitos de função *hashing* mostrou-se promissor e retornou resultados com boa precisão. Utilizando este método duas vezes seguidas, e invertendo as regiões descartadas (região externa para o contorno da íris e a região interna para a pupila) obteve-se resultados animadores para aplicação no processamento de imagens sobre a região da íris humana. A principal vantagem do método de recorte circular proposto, é a não necessidade de verificar os pontos não pertencentes à área visível, o que é obtido devido à busca baseada na estrutura de *hashing*.

Agradecimentos

Os autores agradecem o suporte financeiro da CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) e da FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo).

Referências

Canny, J., 1986. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, n. 6, pp. 679–698.

Chavez, R. F. L., 2007. *Uma proposta para melhoria na eficiência de um sistema de reconhecimento de íris humana*. Tese de Mestrado, Universidade Estadual de Campinas - Faculdade de Eng. Elétrica e Computação. Campinas, SP - Brasil.

Daugman, J., 2004. How iris recognition works. *IEEE transactions on circuits and systems for video technology*, vol. 14, n. 1, pp. 21–30.

Drozdek, A., 2000. *Data Structures and Algorithms in C++*. Brooks/Cole Publishing Co., Pacific Grove, CA, USA.

Foley, J. D., van Dam, A., Feiner, S. K., & Hughes, J. F., 1990. *Computer graphics: principles and practice (2nd ed.)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Gonzalez, R. C., Woods, R. E., & Eddins, S. L., 2003. *Digital Image Processing Using MATLAB*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Greiner, G. & Hormann, K., 1998. Efficient clipping of arbitrary polygons. *ACM Trans. Graph.*, vol. 17, n. 2, pp. 71–83.

Harrington, S., 1987. *Computer graphics: a programming approach, 2nd ed.* McGraw-Hill, Inc., New York, NY, USA.

Hearn, D. & Baker, M. P., 1986. *Computer graphics*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Sutherland, I. E. & Hodgman, G. W., 1974. Reentrant polygon clipping. *Commun. ACM*, vol. 17, n. 1, pp. 32–42.

Wu, Q., Huang, X., & Han, Y., 2006. A clipping algorithm for parabola segments against circular windows. *Computers & Graphics*, vol. 30, n. 4, pp. 540–560.

Zhang, M. & Sabharwal, C. L., 2002. An efficient implementation of parametric line and polygon clipping algorithm. In *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing*, pp. 796–800, New York, NY, USA. ACM.