COMPARAÇÃO DE DIFERENTES MÉTODOS DE RASTREAMENTO DE CARACTERÍSTICAS DA PUPILA HUMANA

Alessandro G.C. Dias, Rafael A. da Silva, Antônio C. P. Veiga, Milena B. P. Carneiro
Universidade Federal de Uberlândia, Programa de Pós Graduação em Engenharia Elétrica, Uberlândia – MG
E-mail: alessandrogcd@gmail.com, rafaelaugustosilva@outlook.com, acpveiga@gmail.com, milenabueno@yahoo.com

***Resumo* – O diâmetro da pupila contrai-se e dilata-se em resposta a diferentes estímulos fisiológicos. Esses movimentos apresentam interesse clínico podendo ser indicadores de condições anômalas dos sistemas visual e nervoso humanos. Diversos algoritmos têm sido desenvolvidos com o intuito de efetuar o rastreamento da posição e tamanho da pupila no método denominado pupilometria. Este trabalho apresenta comparações de precisão e tempo de processamento entre diferentes sistemas de rastreamento pupilar de forma a identificar propostas eficazes de otimização do algoritmo.**

***Palavras-Chave* – Canny; Processamento de vídeo; Rastreamento; Transformada de Hough.**

COMPARISON OF DIFFERENT TRACKING METHODS FOR CHARACTERISTICS OF HUMAN PUPIL

***Abstract* - The pupil is constantly performing movements of contraction and dilation in different conditions. These movements have clinical interest, being able to diagnose anomalous conditions in the human vision and nervous systems. Several algorithms have been developed in order to perform tracking of the position and size of the pupil in a method known as pupillometry. This paper presents comparisons of accuracy and processing time between different pupil tracking systems in real time.**

***Keywords* – Canny; Hough transform; Tracking; Video processing.**

I. INTRODUÇÃO

A pupila realiza movimentos de acordo com as condições às quais está submetida. Tais movimentos podem ser contrações (mioses) ou dilatações (midríases) estimuladas pelo sistema nervoso parassimpático e simpático respectivamente. Fatores como o uso de drogas, respiração, batidas do coração, patologias, idade, cor da íris, nível de consciência, entre outros, podem causar modificações na dinâmica pupilar [3].

XII CEEL – ISSN 2178-8308
13 a 17 de Outubro de 2014
Universidade Federal de Uberlândia – UFU
Uberlândia – Minas Gerais – Brasil

Devido a interesses clínicos na determinação de características pupilares, a pupilometria dinâmica, que consiste na análise dos reflexos pupilares ao estímulo luminoso, tem sido abordada cientificamente e sistemas automáticos desenvolvidos de modo a realizá-la em tempo real.

Os movimentos pupilares consistem basicamente em acomodação, reflexo pupilar à luz (PLR) e *hippus*. Uma variação anormal no movimento de acomodação pode ser resultado de anormalidade no sistema nervoso. A pupilometria analisa os movimentos pupilares através da mensuração de diversos componentes: Amplitude máxima (diferença entre tamanho inicial e mínimo durante o PLR), latência, velocidade de contração e dilatação, tamanhos máximo e mínimo da pupila.

Sistemas de rastreamento denominados *Active Object Tracking* [5] utilizam sensores ou transmissores para marcar o objeto a ser rastreado e são considerados invasivos, mais adequados para aplicação a sistemas com condições controladas. Dessa forma o rastreamento passivo, método abordado neste trabalho, é geralmente preferível na realização da pupilometria apesar de acarretar em maior complexidade do mesmo.

O rastreamento passivo utiliza alguns critérios de representação de modo a determinar características tais como forma, posição, cor e textura. Torna-se mais complexo em casos em que o objeto de interesse apresenta estrutura deformável e padrões de cor variáveis como na situação de rastreamento da pupila, que reage muito rapidamente a variações na iluminação [6].

Nesse âmbito, existem trabalhos [7][8] que utilizam métodos diversos, entre eles a Transformada de Hough, de rastreamento da pupila em tempo real mas que não efetuam o rastreamento do tamanho, somente da posição. Outras publicações existentes efetuam o rastreamento da posição da pupila em tempo real [9][10] utilizando-a somente para determinar a direção do olhar, não apresentando interesse para o presente trabalho.

O objetivo deste trabalho é realizar comparações de desempenho em diferentes sistemas de rastreamento de posição e tamanho da pupila de modo a estabelecer relações entre a acurácia e tempo de processamento, resultantes das diferentes abordagens realizadas. Tal comparação visa, a partir da eficácia de propostas já implementadas, identificar possibilidades de aperfeiçoamento do algoritmo de pupilometria.

Os dois sistemas comparados possuem processos similares na aquisição das imagens, pré-processamento e técnicas utilizadas na determinação das características pupilares. Foram, entretanto, desenvolvidos em linguagens de

programação diferentes e possuem particularidades que serão salientadas nos resultados comparativos.

II. DESCRIÇÃO DO SISTEMA I

Este sistema [1] foi desenvolvido através da ferramenta Matlab® e consiste na busca por um círculo com raio entre 15 e 70 pixels. A busca por circunferências é precedida de um pré-processamento das imagens de modo a prepará-las para a detecção adequada das características e otimização do processo. O processo completo do sistema I é realizado consistindo nas etapas mencionadas a seguir:

A. Conversão para Escala de Cinza

As imagens obtidas pelo dispositivo neste sistema são coloridas, consistindo em uma matriz de três camadas, cada camada sendo uma cor do modelo RGB. A conversão para escala de cinza possibilita a redução dos dados para 2 dimensões sem comprometer sua utilidade para os objetivos do trabalho. Dessa forma consegue-se reduzir o gasto de processamento de cada imagem nas etapas subsequentes.

B. Detecção de bordas – Método de Canny

Bordas são limites entre regiões com níveis de cinza diferentes e são detectadas nesta situação através do método de Canny. O objetivo principal da aplicação deste método é transformar o que não for borda em fundo da imagem (valor 0) otimizando a utilização posterior da Transformada Circular de Hough que efetua a busca por circunferências. A imagem resultante do método de Canny é ainda suavizada, eliminando-se pixels esparsos.

C. Detecção de circunferências – Transformada Circular de Hough

A transformada circular de Hough (TCH) tem o objetivo de localizar circunferências em uma imagem, e sua utilização para a presente situação permite o retorno de características da circunferência encontrada correspondente à pupila, como raio e posição do centroide.

Esta etapa representa a parte mais importante do processo e corresponde à maior parte do gasto de processamento. O método funciona através da criação de uma matriz de votos, que é utilizada para encontrar os círculos após a passagem do algoritmo.

Após a localização das características da circunferência desejadas, os dados dos resultados da busca são armazenados em um documento de texto para posterior análise.

III. DESCRIÇÃO DO SISTEMA II

Desenvolvido no *Microsoft Visual C# 2010 Express*® [16] em todas as etapas (da aquisição de imagens à obtenção dos resultados), esse software [2] efetua pré-processamento utilizando o conjunto de bibliotecas do *AForge.NET* [18]. O pré-processamento é similar, mas possui etapas suplementares em relação ao sistema anterior.

A. Conversão para escala de cinza

A conversão de RGB para escala de cinza, mesmo processo realizado no sistema anterior, é realizada pelo filtro *Grayscale.Common.Algorithms.BT709*.

B. Equalização do Histograma

Em certos casos a transformação de uma imagem para escala de cinza pode resultar em perdas significativas de detalhes em seus objetos. A equalização do histograma produz uma imagem mais uniforme, realçando o contraste entre as bordas dos objetos [11][12]. Esta etapa foi realizada através da chamada da função *HistogramEqualization*.

C. Binarização

Diferentemente do primeiro sistema, a binarização foi efetuada com o intuito de reduzir o número de bordas a serem localizadas na etapa subsequente. Esta etapa converte a imagem equalizada em uma imagem com 2 grupos de pixels, os de frente (valor 1) e os de fundo (valor 0) de acordo com um limiar de decisão definido [11]. Efetuado pelo filtro *Threshold*.

D. Detecção de bordas - Método de Canny

A função *CannyEdgeDetector* efetua a busca de bordas pelo algoritmo de Canny, similarmente ao que é realizado no sistema I (descrito na subseção II.B).

E. Detecção de circunferências – Transformada Circular de Hough

A fim de realizar a busca pela pupila, a transformada de Hough chamada por *HoughCircleTranformation* é modificada de forma a procurar por valores de raio específicos e diminuir significantemente o tempo de processamento.

O sistema II buscou a otimização na determinação das características da circunferência localizada, definindo duas diferentes situações com processos distintos: uma primeira situação onde a primeira imagem ou círculo não é encontrado em mais de três imagens consecutivas e uma segunda situação onde foi encontrado círculo em um intervalo de até três imagens anteriores.

A primeira situação leva à realização da busca por circunferências considerando-se a imagem completa, e o segundo caso resulta em um corte da imagem em 2,5 vezes o diâmetro previamente encontrado antes de realizar o pré-processamento, de forma a reduzir o espaço de busca da circunferência e o tempo de processamento.

IV. COMPARAÇÃO DO DESEMPENHO DOS DOIS SISTEMAS

A comparação dos resultados obtidos softwares considerados foi realizada analisando-se dois diferentes parâmetros: tempo de processamento e precisão na determinação da posição e raio da pupila.

A ambos os sistemas foi submetido o mesmo conjunto de imagens em cada teste sob as mesmas condições e otimizações, de modo a se obter medidas válidas de comparação. Todos os testes foram realizados em um computador com processador Intel Core i5® de segunda geração e 4GB de memória RAM.

A. Teste de Tempo de Processamento

Para realizar o teste em relação ao tempo de processamento foram utilizadas imagens obtidas pelo processo de aquisição do sistema I. O conjunto de imagens foi submetido ao sistema I [1], desenvolvido em Matlab® e ao sistema II [2], desenvolvido em C#, de modo a demostrar a diferença no tempo de processamento.

O conjunto de 349 imagens foi obtido iluminando-se o olho com um LED da cor vermelha, cuja intensidade de iluminação começa em 2% e passa a 40% no quadro 237. Essas imagens foram obtidas por uma *webcam* da marca Logitech® a uma taxa média de 21 quadros por segundo e possuem dimensão de 329 por 245 pixels.

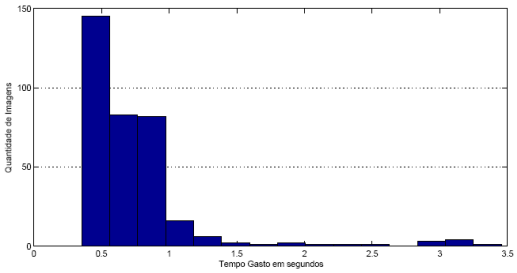

Fig. 1. Distribuição de frequência do tempo gasto no processamento de cada imagem no sistema I.

Para o sistema I, foram encontrados círculos em 96% das imagens. O tempo de processamento de cada imagem está entre 300 e 3500ms, com uma média de 759ms, como observado na Figura 1. Já para o sistema II, foram encontradas circunferências em todas as imagens e o tempo de processamento de cada imagem está na faixa entre 200 e 400ms, com uma média de 343,5ms, o que pode ser constatado na Figura 2. Percebe-se que o sistema II desenvolvido em C# gasta menos da metade do tempo de processamento do sistema II, desenvolvido em Matlab®.

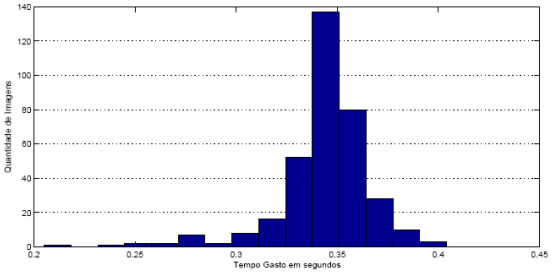

Fig. 2. Distribuição de frequência do tempo gasto no processamento de cada imagem no sistema II.

B. Teste de Precisão na determinação da posição e raio da pupila

O teste de precisão na determinação das características pupilares foi realizado em duas situações. O primeiro teste foi efetuado sob as mesmas condições do teste de tempo de processamento, ou seja, utilizando as mesmas 349 imagens. No segundo teste da precisão foram utilizadas imagens do banco de dados de íris CASIA [22], pertencentes ao conjunto *CASSIA-Iris-Interval*, totalizando 122 imagens. Nas duas situações foi analisada a diferença encontrada entre os dois softwares.

Na primeira configuração é utilizado o mesmo conjunto de imagens do teste de tempo de processamento. Pode-se observar pela Figura 3 que as posições das circunferências encontradas pelos dois sistemas são muito próximas. O gráfico da Figura 4 exibe os valores dos raios para os dois sistemas que, assim como os valores das posições das circunferências, são valores bastante próximos.

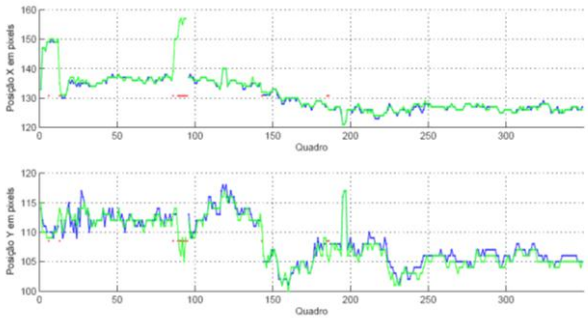

Fig. 3. Posição X e Y do círculo encontrado em cada imagem pelo sistema I (azul) e II (verde), excluídos os círculos não encontrados.

Nas duas figuras a curva de cor azul representa o sistema I (Matlab®), a curva de cor verde representa os resultados obtidos pelo sistema II (C#) e os pontos em vermelho representam as imagens cujos círculos não foram localizados pelo sistema em Matlab (13 imagens).

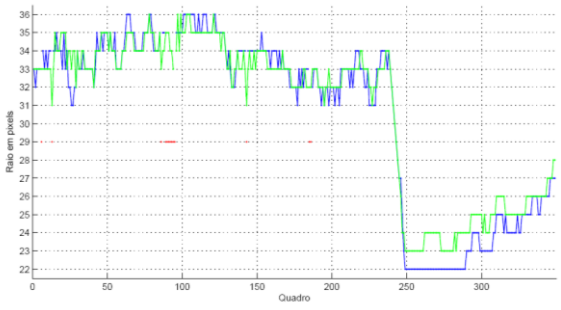

Fig. 4. Tamanho do raio do círculo encontrado em cada imagem pelo sistema I (azul) e II (verde), excluídos os círculos não encontrados.

Já na segunda situação, onde o teste foi realizado com um banco diferente de imagens, obtemos os gráficos das Figuras 5 e 6 que representam respectivamente a distribuição de frequência da diferença do raio e da diferença na distância do centro da circunferência encontrado.

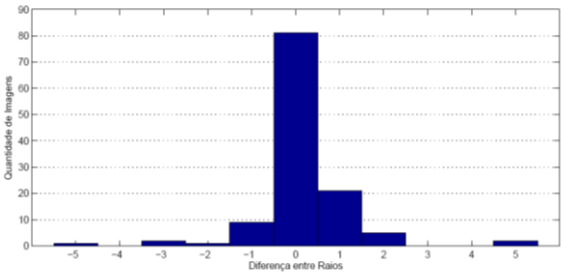

Fig. 5. Distribuição de frequência da diferença de raio encontrado para cada imagem pelos diferentes sistemas.

Chega-se a uma média da diferença de raio encontrado de 0,16 pixels e uma média da distância entre centros de 1 pixel. Pode-se perceber que ambos os sistemas apresentam, portanto, valores próximos encontrados para os parâmetros da pupila desejados.

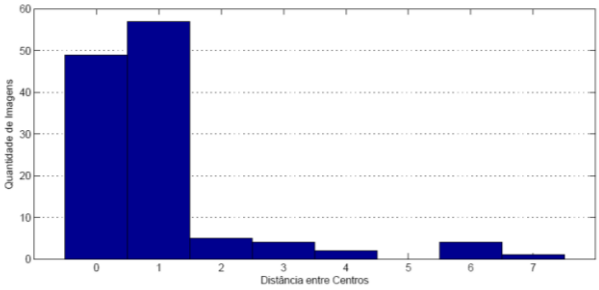

Fig. 6. Distribuição de frequência da distância do centro encontrado para cada imagem pelos diferentes sistemas.

V. CONCLUSÕES

Percebe-se que os sistemas testados possuem resultados similares de precisão na determinação da posição e raio da pupila. É evidente, entretanto, a diferença existente no tempo de processamento obtido nos testes para as duas soluções desenvolvidas.

O sistema desenvolvido em C# obteve performance superior em mais de 50% com relação ao tempo de processamento. Tal diferença pode ser explicada pelas otimizações resultantes de etapas suplementares de pré-processamento (etapas B e C) e diferentes caminhos de execução do algoritmo em situações específicas.

O algoritmo do sistema II obtêm rendimentos ainda superiores se utilizados todos os pré-processamentos implementados [2]. Neste trabalho ambos os sistemas foram submetidos a condições idênticas, de forma a determinar a eficácia e relevância dos aperfeiçoamentos no sistema II em relação ao sistema I, acarretando na limitação de certas características de desempenho existentes no sistema II que obtêm, em condições normais, desempenho ainda superior ao exposto neste artigo.

AGRADECIMENTOS

Os autores agradecem à Fundação de Amparo à Pesquisa do Estado de Minas Gerais – FAPEMIG, pelo apoio financeiro ao projeto aprovado referente ao edital 01/2012.

REFERÊNCIAS

[1] C. R. Bernadelli, Rastreamento em vídeo das características da pupila. 2011.

[2] A. G. da Costa Dias, Pupilometria dinâmica: uma proposta de rastreamento das características da pupila humana em tempo real. 2013.

[3] G. L. Ferrari, Pupilometria dinâmica: Aplicação detecção e avaliação da neuropatia autonômica diabética e estudo da correlação entre a resposta temporal da pupila ao estímulo visual e a glicemia. Universidade Tecnológica Federal do Paraná. 2010.

[4] V. Ibanez, V. Yano, and A. Zimer, “Automatic pupil size measurement based on region growth”, in Biosignals and Biorobotics Conference (BRC), 2012 ISSNIP, pp. 1-4, IEEE, 2012.

[5] A. C. Bovik. The Essential Guide to Video Processing. Academic Press, 2009.

[6] C. R. Bernadelli and A. C. P. Veiga, Iris motion tracking using feature extraction by shape matching. XIII Symposium on Virtual and Augmented Reality SVR-2011, CD-ROM, May/2011.

[7] M. Soltany, S. Zadeh, and H. Pourreza, “Fast and accurate pupil positioning algorithm using circular hough transform and gray projection,” in Proceedings of International Conference on Computer Communication and Management (ICCCM 2011), 2011.

[8] S. Wibirama, S. Tungjitkusolmun, C. Pintavirooj, and K. Hamamoto, “Real time eye tracking using initial centroid and gradient analysis technique,” in Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on, vol. 2, pp. 1054–1057, IEEE, 2009.

[9] M. Ciesla and P. Koziol, “Eye pupil location using webcam,” arXiv preprint arXiv:1202.6517, 2012.

[10] C. Jian-nan, Z. Chuang, Q. Yan-jun, L. Ying, and Y. Li, “Pupil tracking method based on particle filtering in gaze tracking system,” International Journal of the Physical Sciences, vol. 6, no. 5, pp. 1233–1243, 2011.

[11] R. C. Gonzalez and R. E.Woods, Processamento Digital de Imagens. Pearson Prentice Hall, 2010.

[12] M. S. Nixon and A. S. Aguado, Feature Extraction and Image Processing. Academic Press, 2008.

[13] M. Carneiro, “Reconhecimento deiris utilizando algoritmos gen´eticos e amostragem nao uniforme,”

[14] Simon J. K. Pedersen, Circular Hough Transformation. Alborg University, Vision, Graphics, and Interactive Systems. 2007.

[15] C. Kimme, D. Ballard, and J. Sklansky, “Finding circles by an array of accumulators,” Communications of the ACM, vol. 18, no. 2, pp. 120– 122, 1975.

[16] Microsoft Visual Studio (2012), Visual C# 2010 Express. Acedido em 21 de março de 2012, em: http://www.microsoft.com/visualstudio/en-us/products/2010-editions/visual-csharp-express.
[17] Logitech (2012). Logitech HD Webcam C525. Acedido em 19 de março de 2012, em: http://www.logitech.com/en-us/webcam-communications/webcams/hd-webcam-c525.
[18] AForge.NET, "AForge.NET Framework 2.2.4." http://www.aforgenet.com/news/2012.02.23.releasing framework 2.2.4.html. [acessado em 20-Marc¸o-2012].
[19] AForge.NET, "AForge.NET Framework Documentation." http://www.aforgenet.com/framework/docs/. [acessado em 21-Marc¸o-2012].
[20] Biblioteca MSDN, "Desenvolvimento .NET." http://msdn.microsoft.com/pt-br/library/aa139615. [acessado em 21-Marc¸o-2012].
[21] B. Sahin, B. Lamory, X. Levecq, F. Harms, and C. Dainty, "Adaptive optics with pupil tracking for high resolution retinal imaging," Biomedical optics express, vol. 3, no. 2, pp. 225–239, 2012.
[22] AForge.NET, CASIA Iris Image Database. http://biometrics.idealtest.org. [acessado em 12-Maio-2012]