Faculdade de Engenharia da Universidade do Porto

Licenciatura em Engenharia Electrotécnica e de Computadores

Reconhecimento óptico de furos em alvos de tiro desportivo na ISR

Relatório do Estágio Curricular da LEEC 2006

Pedro André da Costa Avelino

Orientador na FEUP: Professor Doutor Armando Sousa

Orientador no ISR: Professor Doutor Armando Araújo

Financiamento FCT/POCI

União Europeia

Agosto de 2006

Governo da República Portuguesa

Ciência.Inovação 2010

621.3(047.3)/
LEEC
2006/AVEp

103

Faculdade de Engenharia da Universidade do Porto

Licenciatura em Engenharia Electrotécnica e de Computadores

Reconhecimento óptico de furos em alvos de tiro desportivo na ISR

Relatório do Estágio Curricular da LEEC 2006

Pedro André da Costa Avelino

Orientador na FEUP: Professor Doutor Armando Sousa

Orientador no ISR: Professor Doutor Armando Araújo

Financiamento FCT/POCI

União Europeia

Agosto de 2006

Governo da República Portuguesa

Ciência.Inovação 2010

104945
24 02 10

Resumo

Este trabalho tem como objectivo procurar classificar alvos de tiro desportivo de uma forma alternativa às soluções actualmente existentes. Não tem como meta final chegar a uma solução imediata, mas sim servir de base a futuros estudos e investigação sobre este modelo.

O modelo final idealizado, será o de um sistema que funcione em tempo real e que esteja implementado no terreno (stand de tiro). Este modelo é composto por uma câmara digital enterrada apontadada ao alvo, que a cada disparo dado pelo atleta capte uma fotografia e a envie para um computador. Na tranferência será usado o "picture transfere protocol" que é um protocolo desenvolvido pela "International Imaging Industry Association" que permite a transferência de imagens entre máquinas fotograficas digitais e computadores ou outros dispositivos remotos. No computador através de um software desenvolvido a imagem é processada e os alvos são classificados.

O trabalho desenvolvido apenas se debruçou sobre uma das partes deste modelo, a parte da análise de imagem e desenvolvimento do software de reconhecimento e classificação dos furos, e tem como objectivo implementar uma solução barata simples e mais rápida que a existente actualmente.

O programa usado para implementação do algoritmo é o Matlab. O Matlab é uma ferramenta poderosa no processamento de imagem, possuindo uma vasta biblioteca de funções nesta área.

Sendo assim o trabalho estende-se desde a captação da imagem ao processamento da mesma seguindo basicamente os seguintes passos:

- A imagem é adquirida a partir de uma máquina fotográfica digital, com resolução não inferior a 6 mega pixeis.
- Ao contrário do que acontece no modelo idealizado, todas as imagens analisadas neste trabalho são adquiridas num plano paralelo ao do alvo, devendo este encontrar-se centrado na imagem.
- De todos os objectos presentes na imagem apenas nos interessam identificar os anéis do alvo e os furos provocados pelos projecteis.
- A classificação de cada um dos furos será dada pela distância de cada um deles ao centro do alvo. Este centro é determinado a partir da identificação de cada um dos anéis.

No final a robustez do algoritmo é testado usando diversas imagens.

Índice de Conteúdos

1 Introdução

1.1 Apresentação da Instituição de Estágio ISR-Porto

O presente trabalho foi desenvolvido no Instituto de Sistemas e Robótica do Porto.

O ISR é uma instituição de I&D que se dedica à elaboração de sistemas automáticos para a indústria e outras aplicações (http://www.fe.up.pt/isrp/).

1.2 Organização e Temas Abordados no Presente Relatório

O relatório encontra-se dividido segundo o seguinte plano:

- **Análise do problema com algum detalhe**:

  Nesta secção é dada uma visão geral de como foi pensado e organizado o trabalho nos seguintes pontos:

  - Aquisição da imagem
  - Os alvos objectos de tratamento de imagem
  - Resoluções minimas exigidas à imagem adquirida
  - Método de identificação dos furos na imagem
  - Matlab como ferramenta de trabalho

- **Tecnologia actualmente existente**

  Nesta secção é dada a conhecer as tecnologias actualmente existentes para a resolução do problema em comum.

- **Descrição detalhada da solução proposta**

  Nesta secção é dada uma visão pormenorizada da solução final encontrada para o desenvolvimento do algoritmo.

- **Défices do algoritmo e melhoramentos a realizar**

  Nesta secção é dado a conhecer tudo aquilo que poderá ser melhorado na construção do algoritmo.

- **Algoritmo desenvolvido e código Matlab**

Nesta secção encontra-se a explicação resumida do funcionamento do algoritmo, e o código desenvolvido em código Matlab.

- **Avaliação de resultados e conclusões**

Nesta secção encontra-se todas as imagens testadas, e os resultados obtidos durante o teste.

- **Bibliografia**

2 Análise do problema com algum detalhe

Aquisição da imagem

As condições em que é feita a aquisição da imagem revelou-se de extrema importância e exige alguns cuidados.

Para se captar uma imagem em boas condições para ser processada, é importante evitar a presença de sombras e brilhos que adulterem aquilo que realmente é desejado identificar na imagem, já que sombras e brilhos introduzem objectos estranhos à imagem.

A imagem é adquirida através de uma câmara digital com resolução não inferior a seis megapixeis. Esta foi a resolução mínima calculada, de forma a ser possível retirar o máximo de informação da imagem para que se obtenha o máximo de precisão possível nas medições que irão ser feitas posteriormente. Como um dos objectos essenciais a identificar na imagem são os anéis, e como estes têm apenas algumas décimas de milímetro de espessura, é imperativo que na imagem eles apresentem algumas unidades de píxel de espessura.

A câmara deve encontrar-se num plano o mais paralelo possível em relação ao do alvo, de forma a minorar distorções verticais e horizontais da imagem.

Os alvos objectos de tratamento de imagem

Os alvos são feitos de cartão de diferentes texturas, sendo uns mais rugosos do que outros, contendo mais ou menos imperfeições. Assim sendo cada um deles exige à partida diferentes tipos de tratamento de imagem. Uma destas operações é a forma como irá ser implementada a remoção dos furos do alvo onde é necessário minimizar os danos causados ao cartão pelo impacto do projectil.

Consoante a prova de tiro a que se destina, em cada alvo será necessário identificar e classificar de 3 a 5 furos.

Cada alvo possui dimensões oficiais próprias, tendo todos em comum o mesmo número de aneis (11 aneis ao todo) concêntricos num mesmo ponto. Este ponto, que é o centro do alvo, é um dos pontos essenciais a identificar.

Tirando a diferença de raio entre o anel de pontuação 10 e de pontuação 9, todos os outros anéis possuem uma diferença constante de raio de anel para anel. Irá ser através desta medida real em milímetros e da medida obtida pelo algoritmo em pixeis, que se vai calcular a resolução dada pelo algoritmo.

Resoluções mínimas exigidas à imagem adquirida

Diferentes tamanhos nos alvos exigem diferentes resoluções à imagem adquirida. Cada alvo irá exigir uma resolução mínima para que na altura da classificação dos furos na imagem não sejam cometidos erros nas medições.

Durante o trabalho irão ser usados 3 alvos diferentes: tiro com carabina a 50 metros e tiro com pistola a 10 e 25 metros. O primeiro problema que surgiu foi o de determinar a resolução minima exigida à fotografia para se obter resoluções que garantam medições sem erros, ou seja em que cada anel apresente pelo menos 1 pixéis de espessura.

O alvo mais pequeno (tiro com pistola a 10m) é o menos exigente. Mede 80x80 mm sendo a espessura do anel de 0.1mm logo a resolução mínima necessária é de 800x800 pixeis. O alvo de tamanho intermédio (tiro com carabina a 50 metros) mede 250x250 mm e exige resolver 0.1mm (espessura do anel) em 154.4mm (diametro do anel maior) exigindo por isso uma resolução de 1544x1544 píxeis . O alvo maior e portanto mais exigente em resolução, necessita de resolver 0,2 mm em 500 mm, ou seja, exige uma resolução mínima de 2500x2500 píxeis. A escolha de uma máquina digital com 6mega píxeis de resolução mostra-se teoricamente eficaz, podendo até no caso dos alvos mais pequenos obter-se um anel com mais do que 1 pixel de espessura.

Método de identificação dos furos na imagem

O segundo problema é como identificar uma zona furada no cartão do alvo correspondente ao furo causado pelo projectil. A maneira mais fácil encontrada foi colocar por detrás do alvo, aquando da altura da aquisição da imagem, uma superficie que se destaque das restantes cores padrão do alvo. O vermelho mostrou-se após várias experiências a côr mais favorável, já que é uma das componentes em RGB e causa maior contraste com as restantes cores originais do alvo. Para então se proceder à detecção dos furos na imagem basta procurar na imagem todos os pixeis de cor vermelha.

Matlab como ferramenta de trabalho

O software desenvolvido vai ser implementado em matlab. O matlab é uma ferramenta poderosa no tratamento de imagens possuindo uma biblioteca vasta de funções de auxilio ao processamento de imagem. Possibilita igualmente a construção de interfaces gráficos. Apesar de todas estas potencialidades na análise de imagem o matlab é igualmente lento nos processos, e como um pré-requisito para este trabalho é o baixo tempo de processamento há que escolher bem as funções a utilizar, ou implementar funções alternativas que produzam o efeito desejado em menos tempo de processamento.

3 Tecnologia actualmente existente

Este trabalho tem como objectivo servir de início a um estudo que permita no futuro implementar um sistema de classificação automática de alvos de tiros desportivo, uma solução que possa vir a ser implementada no terreno, coisa que actualmente ainda não existe.

Esta solução tem que ser economicamente viável, de grande fiabibilidade e que seja uma aplicação em tempo real.

Os únicos alvos que reconhecem automaticamente as pontuações são os alvos electrónicos. Estes alvos simulam o uso dos alvos analisados neste trabalho, e calculam a respectiva pontuação através de um sistema electrónico próprio, que se baseia na triangulação do som do impacte do projectil no alvo.

Actualmente o único sistema de pontuação existente para os alvos em cartão, é uma máquina onde são inseridos os alvos e funciona por scannerização dos mesmos (figura 1). Para classificar cada uma das diferentes modalidades, este aparelho exige à partida que se especifique o tipo de prova e o número de furos existentes no cartão sendo o processo de classificação lento.

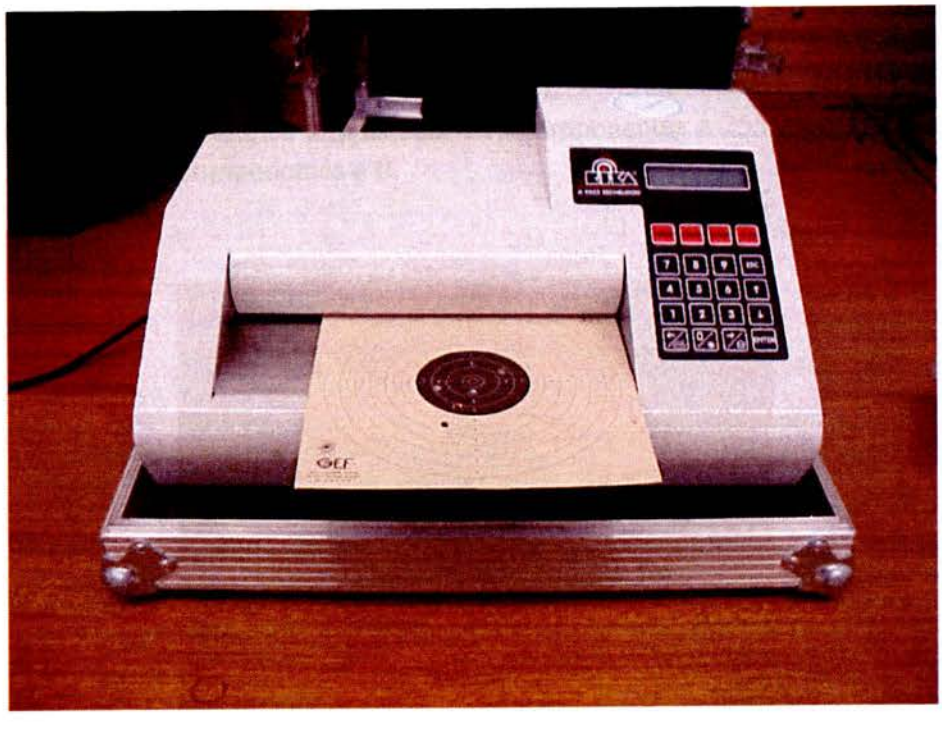

Figura 1.

4 Descrição detalhada da solução proposta

Neste capitulo do relatório, é feita a descrição detalhada do algoritmo construído, bem como a explicação das decisões tomadas aquando da sua construção.

Para uma melhor deste trabalho ser atingido, dividiu-se o algoritmo basicamente em 4 partes distintas:

- Identificação dos furos na imagem adquirida.
- Cálculo do centro dos furos.
- Identificação dos anéis do alvo na imagem e cálculo dos respectivos centros.
- Classificação de cada tiro no alvo.

Identificação dos furos na imagem adquirida:

Uma imagem é uma matriz a três dimensões, em que cada ponto dessa matriz representa um pixel na imagem. Cada dimensão que compõe a matriz principal, é por sua vez uma matriz que representa as componentes RGB da imagem; componente vermelha, verde e azul respectivamente. Os valores dos pontos dessas matrizes variam entre os valores 0 e 255. Por exemplo, um pixel a cor branca terá as suas três componentes a 255 e um pixel a preto terá por sua vez as suas três componentes a 0.

Figura 2.

Nos alvos existem basicamente apenas duas cores. Não são um branco e um preto puro mas são duas cores perfeitamente distintas, que podemos classificar como cor clara e cor escura. As componentes em RGB das zonas mais claras e das zonas mais escuras são bastante distintas, sendo por isso fácil a aplicação de um "threshold" para binarização da imagem.

Todos os alvos possuem a zona mais escura no centro do alvo. Nessas zonas os anéis são da cor mais clara, sendo o restante alvo inversamente preenchido a fundo claro e a anéis escuros.

Na imagem adquirida além destas duas cores encontra-se a cor vermelha introduzida da altura da aquisição da imagem com já foi descrito. Um pixel a vermelho na imagem terá uma componente forte em R e uma componente fraca ou de valor nulo em G e B.

Como o objectivo é retirar da imagem apenas os píxeis que sejam classificados como vermelhos ignorando todos os outros, foi decidido criar uma imagem de igual dimensão à original onde todos os píxeis identificados como vermelhos ficaríam a preto e os restantes píxeis da imagem a branco.

Para isso basta retirar ao valor máximo que cada pixel pode tomar, ou seja, a 255 a sua componente em R, e somar ao valor obtido as suas componentes em G e B. Nesta imagem obtida são claramente visíveis a escuro as zonas a vermelho que desejamos identificar (figura 3).

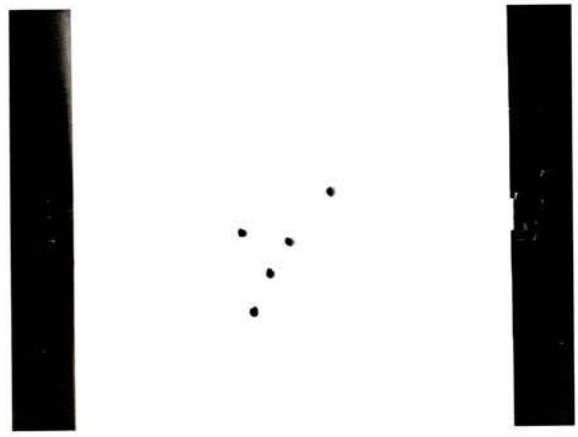

Figura 3.

Após obtida esta imagem, é possivel observar que se trata de uma imagem em escala de cinzento, e não uma imagem a preto e branco como desejado. Para se obter uma imagem a preto e branco basta aplicar a toda a imagem um "threshold" apropriado que através de várias experiências o valor de 180 mostrou-se eficaz.

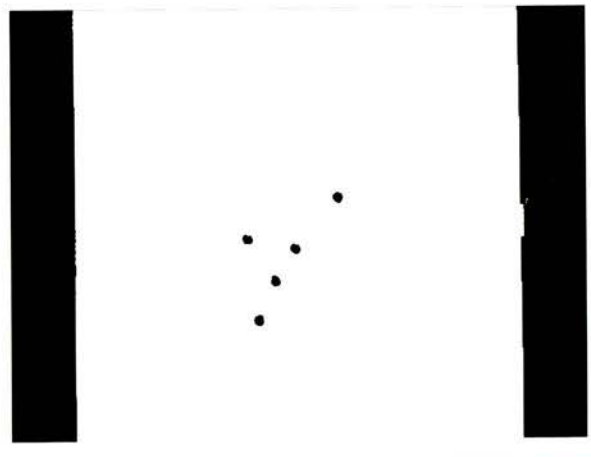

Figura 4.

Código matlab:

% alvo --- imagem original adquirida

*% furos(:,:) = ((255 - alvo(:,:,1) + alvo (:,:,2) + alvo(:,:,3)) > 180) * 255;*

Na figura 2 temos a imagem desejada, onde se podem observar os 5 furos prefeitamente distinguidos dos restantes píxeis do alvo, e onde é possível igualmente ver as duas faixas laterais a preto que representam o fundo igualmente vermelho onde o alvo foi colocado aquando da aquisição da imagem.

Uma vez identificados os furos, é necessário agora calcular o centro de cada um deles. Os furos possuem uma forma arredondada. Se aproximarmos cada um deles a uma circunferência obtemos o centro desejado. Ao aproximarmos o furo a uma circunferência é cometido um erro, mas graças à elevada resolução da imagem adquirida este erro é desprezável, já que cada furo possui um número muito elevado de píxeis, sendo o centro calculado extremamente próximo do centro ideal.

Para calcular uma circunferência bastam 3 pontos, mas se identificarmos a orla de cada furo, podemos usar todos os pontos que a constituem para o calculo do centro minorando assim o erro.

Para calcular a orla foram usadas inicialmente algumas funções de "edge detection" que o matlab possui. Estas funções são baseadas no calculo de derivadas, logo de processamento muito lento. Como um dos objectivos deste trababalho é o processamento rápido do algoritmo construído, ao longo do trabalho irão ser evitadas quaisquer operações de derivação, optando-se por processos alternativos que se demonstrem igualmente eficazes.

Como estamos na presença de imagens binárias, ou seja, uma imagem contituída apenas por pixeis com valores de 0 e 1, preto 0 e branco 1, podemos aplicar-lhes as operações lógicas elementares AND, OR, etc.

A solução adoptada e igualmente eficaz à de "edge detection", foi a utilização de operações de dilatação, ou seja, em cada furo identificado é acrescentado em redor do seu corpo mais um pixel. Ao dilatarmos a imagem original num pixel e fazendo um XOR entre esta nova imagem e a imagem original, obtemos as orlas de todos os furos (figura 5).

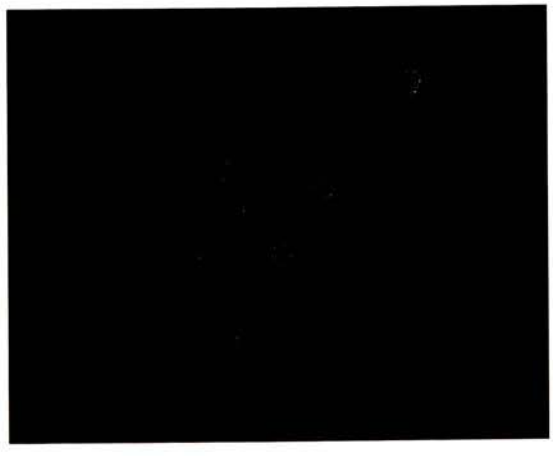

Figura 5.

Uma vez detectadas as orlas dos vários furos, procedemos ao cálculo dos respectivos centros. O método consiste no varrimento da imagem, da esquerda para a direita, até que seja detectado o primeiro branco, ou seja, um 1 na imagem. Relembramos que a imagem tem o fundo a preto, ou seja a 0, e apenas as orlas dos furos se encontram a 1. Portanto qualquer pixel a 1 encontrado na imagem pertence a uma orla.

Uma vez encontrado um ponto pertencente a uma orla, aplica-se a função (bwtraceboundary) de conectividade do matlab. Esta função, a partir deste ponto inicial, procura o ponto vizinho seguinte que se encontra a si ligado, e percorre sucessivamente toda a orla do furo até retornar ao ponto inicial, guardando num vector todas as coordenadas dos pontos por onde passa. Através deste vector de coordenadas e aplicando a equação da circunferência é estimado o centro de cada furo.

Código Matlab

```
contour = bwtraceboundary(BW7, [row, col], 'S', 8, inf);

abc=[contour(:,2)  contour(:,1)  ones(length(contour(:,2)),1)] \ [-(contour(:,2).^2+contour(:,1).^2)];

    a = abc(1); b = abc(2); c = abc(3);

    xc = -a/2;   % calculo d a localização do centro(xc,yc)

    yc = -b/2;
```

Ao longo do teste deste algoritmo, verificou-se que os furos não tinham um raio inferior a 40 pixeis, logo para uma economia de tempo de processamento, é possível varrer a imagem em vez de linha a linha, varre-la por exemplo de 10 em 10 linhas, tornando o algoritmo muito mais rápido.

Outro ponto que se levanta é que mesmo varrento a imagem de 10 em 10 linhas, são naturalmente encontrados ao longo do varrimento pontos que pertencem a uma orla já detectada. Para evitar que esta orla seja novamente percorrida e recalculado o seu centro foi implementado o seguinte método :

- Foi criado um vector de coordenadas de centros calculados, contendo no seu interior apenas o valor [0,0]. Este ponto não pertence à imagem, já que qualquer matriz inicia-se no ponto de coordenadas [1,1], ou seja, [0,0] será um ponto auxiliar inicial.

- Aquando da identificação do primeiro ponto pertencendo a uma orla e posterior cálculo do seu centro, é calculada a distância a todos os pontos contidos no vector de centros (que inicialmente apenas contem as coordenadas [0,0]). Se esta distância for maior que um raio médio de um furo, que se situa nos 25 pixeis, então estamos presentes a uma orla ainda não percorrida, que dará origem a um centro novo que será acrescentado ao vector de centros.

- Este método é repetido sucessivamente e sempre que um ponto de uma orla é detectado, é calculada a sua distância a todos os pontos guardados no vector de centros calculados. Se a distância mínima de todas as distâncias calculadas for maior que um raio médio, é considerado orla nova, ou seja, furo de centro ainda não calculado.

Exemplo de aplicação da função:

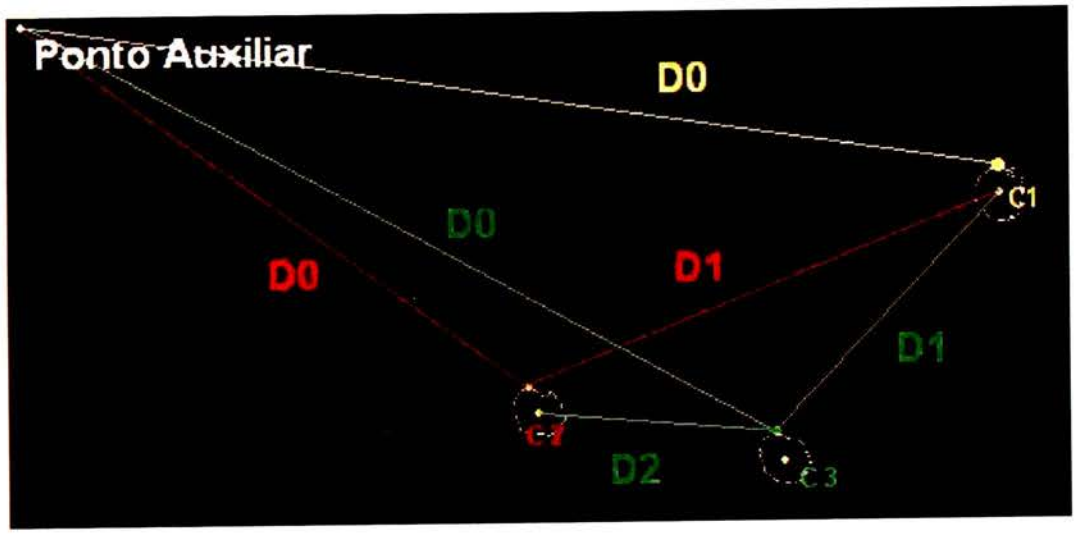

Figura 6. (ordem de detecção dos furos (furo novo): amarelo, vermelho e verde. Onde D0, D1, D2 representam distâncias calculadas aos pontos existentes no vector de centros [0,0],C1 2 C2)

Este método permite que sempre que um ponto de uma orla seja encontrado, a orla não seja percorrida e o seu centro determinado, o que iria acontecer inúmeras vezes. Por exemplo uma linha da imagem que atravesse uma orla de um furo, e que toque em dois dos seus pontos, a orla desse furo iria ser percorrida duas vezes, acontecendo o mesmo com o seu centro o que iria tornar o algoritmo muito mais lento.

Finalmente após o varrimento da imagem completa, é guardado o vector dos centros calculados, que será obviamente indispensável para que no futuro sejam calculadas as distâncias de cada centro dos furos ao centro do alvo e dadas as classificações.

Cálculo do centro do alvo

O centro do alvo irá ser determinado a partir dos anéis que compõe.

Todos os alvos possuem 11 anéis centrados no mesmo ponto. O objectivo é usar todos os anéis no calculo do centro do alvo.

Graças à elevada resolução obtida na imagem adquirida, cada anel possui uma espessura em píxeis que nos permite tirar a sua orla interior e exterior, duplicando assim o número de anéis usados para o cálculo do centro do alvo.

Antes de proceder à detectção dos anéis no alvo, é necessário preparar a imagem. Em algumas provas de tiro, o cartão à volta do furo fica bastante danificado como é possível ver na imagem seguinte (figura 7).

Figura 7. (Este é um alvo correspondente à prova de tiro com carabina a 50 metros, e é o alvo estudado onde se verificam mais danos causados pelo projectil no rebordo da zona furada)

Estes danos provocados no cartão são especialmente prejudiciais quando se situam por cima de um anel.

Após uma binarização da imagem podemos ver que o cartão rasgado fica a branco, confundindo-se com um anel (figura 8). Ao percorrermos a orla de um qualquer anel atingido por um furo, somos obrigados a percorrer igualmente a orla do furo e são cometidos erros na estimativa dos centros de cada anel, já que são acrescentados pixeis não pertencentes ao anel.

Figura 8. (imagem do binarizada)

Figura 9. (Este é um alvo correspondente à prova de tiro com pistola a 10 metros, é possível verificar que neste caso o projectil não causa tantos danos no rebordo da zona furada)

A solução será retirar os furos da imagem. Para isso basta dilatar o furo de forma a que este tape a zona de cartão rasgado.

Como é obvio além de serem retirados os furos também são apagadas as zonas pertencentes a anéis. É impossível reaver os píxeis pertencentes aos anéis danificados pelo furo, mas ao menos estes não irão prejudicar o cálculo do centro de cada furo.

Caso um furo esteja situado sobre um anel e na zona de varrimento da imagem, este anel não é detectado, o que ao longo dos vários testes se mostrou ser um mal menor, já que o centro criado por um anel afectado por um furo produz um centro bastante desviado do real.

Para então se obter uma imagem limpa de furos foi necessário seguir alguns passos:

- Apenas é possível proceder à dilatação de um corpo que esteja com o valor lógico 1 num fundo a 0. Como a nossa imagem binária contém os furos a preto em fundo branco, é necessário primeiro negar a imagem (figura 10 b)) e só depois proceder à sua dilatação (figura 10 c)). Após a dilatação, a imagem é novamente negada e é aplicada a operação lógica AND entre imagem que contém os furos dilatados com a imagem do alvo binarizada.

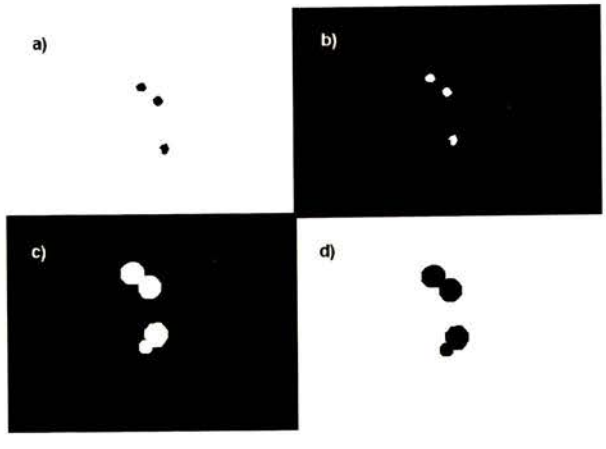

Figura 10.

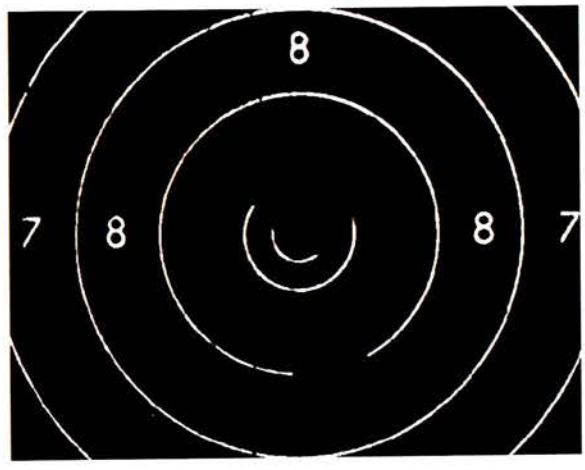

Figura 11. (imagem obtida após a serem retirados os furos)

De notar que a operação de dilatação dos furos para remoção do rebordo rasgado, varia em número de unidades consoante o tipo de prova de tiro, já que os danos causados nos cartões dos alvos pelos projecteis não são iguais em todos os alvos como já foi dito anteriormente.

Após retirados os furos podemos passar ao varrimento da imagem e detecção de anéis.

Detecção dos anés e cálculo dos respectivos centros

Antes de procedermos à detecção dos anéis no alvo, há certos elementos que temos de ter em conta, para que quando iniciarmos o varrimento da imagem, como por exemplo não sejamos prejudicados por objectos os números de pontuação existentes no alvo.

Figura 12. (disposição nos números de pontuação no alvo)

Todos os alvos possuem entre os anéis números que identificam a pontuação de cada tiro. Inicialmente optou-se por uma estratégia de os retirar da imagem. Mas a função que implementava esse processo demonstrou-se demasiado morosa no seu processamento sendo então essa opção posta de parte.

Assim sendo, estes números devem ser pura e simplesmente evitados aquando do varrimento da imagem, já que são objectos que pertencem à imagem mas não interessam ser analisados, podendo causar problemas na futura identificação dos anéis. Estes números possuem a mesma cor dos anéis independentemente da zona em que se encontram do alvo, ou seja, são claros na zona escura e escuros na zona clara, podendo ser confundidos com um possível anel aquando do varrimento da imagem. A melhor forma de os evitar, é varrer a imagem numa direcção em que eles não se encontrem. Como os números se encontram dispostos vertical e horizontalmente no alvo, varrendo a imagem na diagonal estes são evitados.

Outro problema é como tornar possível identificar a totalidade dos anéis existentes no alvo para o calculo do centro, já que estes possuem cores diferentes e após uma binarização da imagem uns aparecem a branco e outros a preto?

O ideal será então transformar a imagem do alvo, numa imagem binária, onde a totalidade dos anéis do alvo estejam a branco em fundo preto ou vice versa. Como implementar isto?

Para uma maior segurança em termos de precisão no cálculo do centro do alvo, procede-se à detecção da orla interna e externa de cada anel. Para isso seguiram-se passos semelhantes ao dados para o cálculo das orlas dos furos, ou seja, através de operações de dilatação e de operações lógicas sobre imagens.

Como os alvos possuem duas partes distintas anéis claros em zona escura e anéis escuros em fundo claro, vamos ter que fazer a detecção dos anéis em duas partes distintas, ou seja, detectar os anéis claros e depois os escuros e juntar a totalidade dos anéis numa só imagem.

A primeira operação sobre a imagem é a binarização e é feita com um "threshold" apropriado, determinado experimentalmente e testado nas várias imagens com sucesso (figura13.).

Figura 13.

Em seguida aplica-se uma dilatação à imagem. Esta operação tem aqui como principal objectivo evidenciar os anéis a branco que se encontram no fundo preto e eliminar os anéis a preto na imagem. Ao dilatarmos a imagem binarizada, estamos a dilatar todos os brancos da imagem, obtendo-se uma imagem onde desaparecem os anéis a preto e na zona a escuro se evedenciam os anéis a branco como desejado (figura 13).

Caso se verifique que algum anel apresenta quebras, esta operação também possibilita novamente a sua união.

Figura 14.

O tratamento dos aneís a preto é exactamente igual ao tratamento dos anéis a branco apenas se parte da imagem binarizada original negada (figura 15.).

Figura 15.

Com as duas imagens obtidas anteriormente e para as juntar numa só, basta aplicar a operação lógica XOR entre as duas imagens. Voltando a negar a imagem resultante obtemos todos os aneis a branco em fundo preto (figura 16).

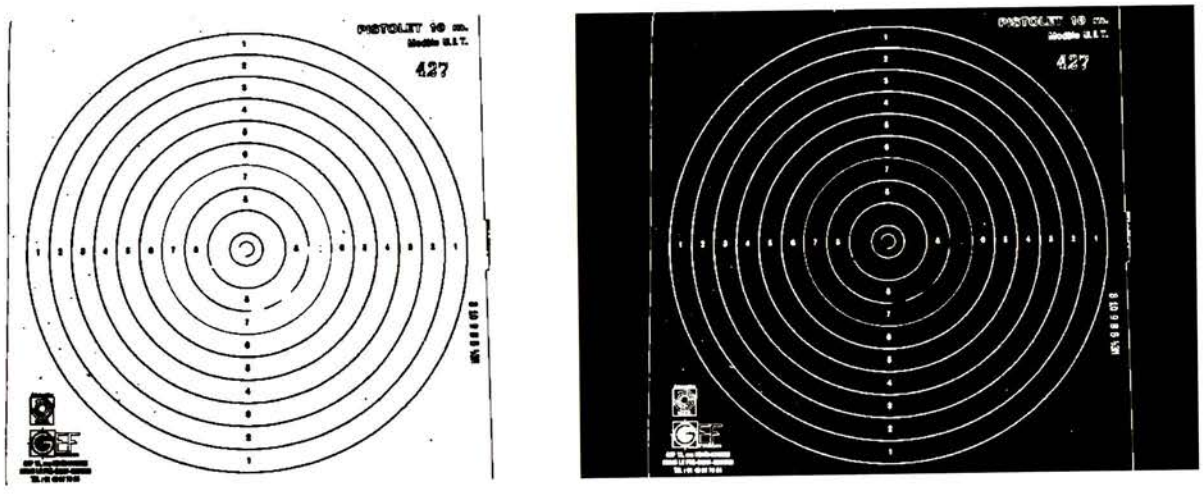

Figura 16.

A obtenção das orlas internas e externas de cada anel segue exactamente o mesmo procedimento que foi utilizado na detecção das orlas dos furos.

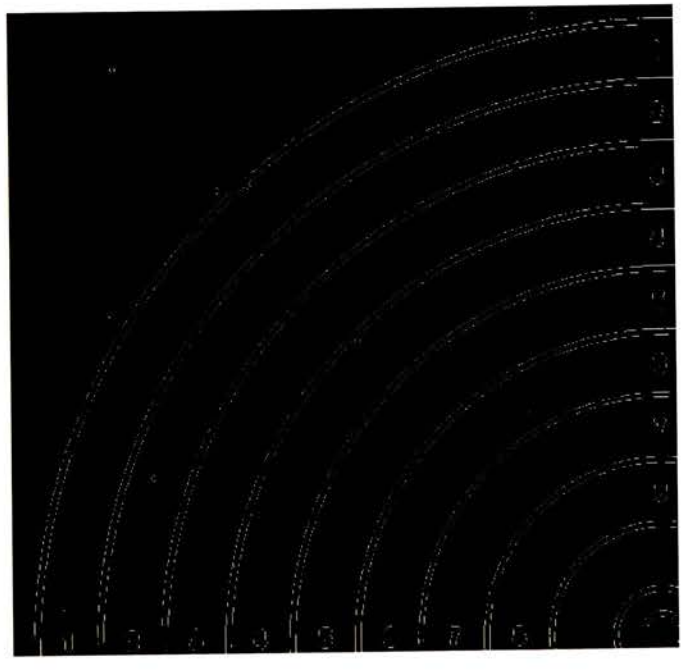

Figura 17.

Após detectadas as orlas internas e externas de cada anel pode-se começar executar o varrimento da imagem.

Como o centro do alvo coincide com o centro da imagem, vamos dividir a altura e largura da imagem por dois e iniciamos o varrimento da imagem nesse ponto. Conforme já foi mencionado o varrimento da imagem será feito de dentro para fora na diagonal de forma a evitar os números existentes no alvo.

Como critério de paragem de varrimento da imagem foi usado o número de linhas varridas, que vão diminuindo à medida que decorre o varrimento da imagem e o número de anéis detectados. Poderíamos ter usado só número de anéis, já que existem 11 anéis em cada alvo, com limites internos e externos o número passa para 22, e que ao fim de encontrados os 22 aneís este parasse de procurar mais, mas se algum anel é atingido por um furo na direcção de varrimento este será ignorado e o número total de aneís detectados nunca chegaría aos 22.

O varrimento é feito na direcção NW de modo a evitar igualmente tanto os números existentes dentro do alvo como as letras e caracteres existentes fora dos alvos.

O método de detecção dos anéis e cálculo dos respectivos centros é similar ao usado anteriormente no cálculo dos centros dos furos. O algoritmo varre a imagem e quando encontra um píxel a branco, supostamente pertencente à orla de um anel, começa a percorrer toda orla tal e qual como anteriormente descrito sendo então calculado o seu centro.

Caso não se trate de uma orla de um anel, ou seja, o objecto encontrado na imagem não produz um centro dentro um raio admissível para centros encontrados, este objecto é descartado e a operação de varrimento é retomada (função "verifica"). Este género de objectos são originados por ruído existente na imagem ou causados por rugosidades ou imperfeições existentes no cartão do alvo.

Este método permite-nos varrer a imagem em qualquer direcção desejada, passando inclusivamente pelos números, mas por uma questão de rapidez de processamento é preferível evitar-los e escolher um caminho limpo destes objectos previsíveis.

À medida que são detectados os aneís e calculados os respectivos centros, os raios de cada um dos anéis e respectivos centros são guardados em dois vectores diferentes. Depois de o vector de centros de aneís completamente preenchido, é calculada uma média entre todos os centros calculados e assim estimado o centro real do alvo.

No vector que possui o valor dos raios internos e externos de cada anel, é feita a média de cada cada par interno e externo de anel e estimado o raio de cada anel.

Cálculo da resolução em milímetros por pixel

Após obtido o vector de raios dos anéis do alvo, é calculado a partir deste, um outro vector que contém as distâncias entre os vários anéis. Tirando o anel de raio mais pequeno existente nos alvos, todas as restantes diferenças entre anéis são constantes (8mm), ou seja, todos os

valores em píxeis existentes neste vector andarão na mesma ordem de grandeza. É a partir de esta distância entre anéis que se vai estimar a resolução obtida.

Mas como já foi referido a detecção de um anel pode falhar ao longo de um varrimento. Então para garantir que neste vector de diferenças em píxeis entre os vários anéis apenas estejam valores na mesma ordem de grandeza, é usado o método dos mínimos quadrados. Da aproximação à recta $y = mx + b$, são retirados do vector todos valores que estejam demasiado longe do pretendido, e com restantes é calculado o valor padrão de diferença entre anéis com a ajuda do valor do erro quadrático.

Para determinar a resolução obtida basta agora apenas dividir os 8mm pelo número de píxeis de diferença entre anéis ou vice versa.

Este valor de resolução vai ser importante depois para obter a classificação de cada furo. A classificação irá ser feita calculando a distância de cada centro de furo ao centro do alvo, subtraída do valor do raio da munição. Este valor da resolução vai ser o valor que nos vai permitir fazer uma conversão de milímetros para píxeis.

Cálculo das classificações

O cálculo das classificações tem como base as medidas oficiais dos vários alvos. Estas estão regulamentadas e publicadas num manual da federação internacional de tiro desportivo (ISSF) "Technical Rules For All Shooting Disciplines".

Como cada alvo possui as suas medidas oficiais, para cada alvo foi criada uma matriz contendo os raios dos vários anéis que os constituem. Cada linha da matriz representa um valor de pontuação e está dividida em dez intervalos, já que a pontução final vai ter que ser dada com uma casa decimal de precisão. Por exemplo a primeira linha é composta pela pontuação 10 (pontuação mais alta), a segunda pela 9 e assim sucessivamente até à pontuação 1.

De notar que as pontuações vão desde 1,0 1,1 .....9,8 9,9 até 10. Apesar da linha de pontuação 10 se encontrar igualmente dividida em dez partes, não irá ter casas decimais apenas se encontra assim por uma questão de contrução da matriz de pontuações.

Nesta matriz de pontuções nas várias linhas e colunas encontram-se as distâncias em milímetros entre os vários anéis.

Por exemplo para o alvo de tiro com pistola a 10 metros:

- A linha da matriz correspondente à pontuação 9 começa quando a distância ao centro do alvo é superior a 5,75mm e termina quando esta é igual a 13,75mm, ou seja, 8mm de largura. Como a pontuação irá ser dada com uma casa decimal, esta largura tem de ser dividida em 10 intervalos, ou seja, em intervalos de 0,8 mm de largura.

Após construida a matriz de pontuações basta calcular a distância de cada centro de tiro ao centro do alvo subtraindo o raio da munição, e ver em qual dos intervalos contidos na matriz se encaixa esta distância atribuindo-lhe a respectiva classificação (figura 17).

Figura 17.

5 Défices do algoritmo e melhoramentos a realizar

Um dos défices que este algoritmo apresenta é de estar apenas testado para imagens a 6 mega pixeis. Se forem usadas imagens de uma resolução superior, terão que ser feitas alterações no número de unidades de dilatação a serem efectuadas ao furo para remoção das zonas de rebordo dos furos.

Outro défice na capacidade de remoção dos furos da imagem verifica-se quando estes estão situados sobre a parte clara do alvo onde o algoritmo desenvolvido não os consegue remover. Caso estes não se encontrem sobre um anel, com a ajuda da função "verifica" descrita anteriormente, o resultado final da detecção de anéis e respectivos centros não sofre qualquer tipo de erro, mas se os furos se encontrarem por cima de um anel do alvo, o cálculo do centro do desse anel pode conter erros.

O problema mais complexo para o qual não foi desenvolvida nenhuma função que o possa minorar, é o facto de os furos às vezes se encontrarem sobrepostos ou unidos, e o algoritmo desenvolvido calcular um centro único para os dois furos. O algoritmo apenas tem a capacidade de reconhecer se a zona detectada como furo é efectivamente apenas um furo ou se é mais do que um furo sobreposto. Como já foi descrito anteriormente, é usada uma função de conectividade para percorrer as várias orlas dos furos. Esta função guarda num vector todos os pontos que constituem as respectivas orlas. A partir da dimensão deste vector sabemos qual é o número médio de pontos que constitui o perímetro de cada furo. Caso o valor exceda esse número médio de pontos previsto, trata-se então de mais do que um furo detectado.

Como já foi dito este trabalho é um trabalho que exige continuação e que se apoia muito na experimentação.

Este algoritmo foi desenvolvido segundo uma linha de pensamento, mas o mesmo problema poderá se encarado de diversas formas. Todas as alterações que possam ser feitas de forma a melhorar o desempenho do algoritmo em termos de redução de tempo de processamento serão muito positivas, já que o objectivo do trabalho é evoluir no sentido de uma aplicação em tempo real.

6 Algoritmo Desenvolvido e Código Matlab

--------------------|--Algoritmo construído incluindo pequenas anotações--|------------------------

O Algoritmo encontra-se dividido em 5 partes distintas:

- Identificação do tipo de prova;
- Detecção dos furos no cartão do alvo;
- Calculo do centro geometrico dos furos;
- Calculo do centro geometrico do alvo;
- Atribuição da pontução a cada tiro;

»»»»»»»»»»»»»»»»»»»»»»»»»» identificação do tipo de prova ««««««««««««««««««««««««««««««

0bjectivo:

- Diferentes provas significam diferentes resoluções obtidas (cada alvo "enche" a imagem adquirida a 6megapixeis e encontra-se centrado na imagem)
- consequentemente, obteremos diferentes tamanhos no diâmetro dos furos (importante para o calculo dos centros dos furos)
- diferentes raios de aneis de alvo para alvo

»»»»»»»»»»»»»»»»»»»» »»»»»»»Detecção dos furos no cartão do alvo«««««««««««««««««««««««««««

- Para uma melhor identificação os furos encontram-se a vermelho;

Processos:

- Varrimento da imagem pixel a pixel, linha a linha;
- é criada uma imagem (furos) a duas dimensões, resultante dos seguintes cálculos:
- Ao valor 255 é retirado a componete R (vermelho) de cada pixel e somadas as outras duas componentes G (verde) e B (azul), obtendo-se uma imagem em escala de cinzento, onde os furos aparecem a mais escuro. Procede-se a um threshold apropriado e obtem-se uma imagem de fundo branco(255) com com os furos evidenciados a preto(0);

»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»» Calculo do centro geométrico dos furos ««««««««««««««««««««««««««««««««

Processos:

- Binarização da imagem (furos) ficando os furos a preto e o resto da imagem a branco (furosbw);

- Detecção das orlas dos furos:

- Dilatação de um pixel da imagem binarizada (furosbwd);

- Operação lógica XOR entre as duas imagens XOR(furos,furosbwd) obtendo-se assim as orlas dos diversos furos (furosedge), onde as orlas se encontram a branco e a restante imagem a preto;

- Varrimento da imagem (furosEdge) da direita pa esquerda, de 10 em 10 linhas (furos têm em geral mais de 20 pixeis de diametro) de forma a encontrar o primeiro branco pertencente a uma orla de um furo;

- Cálculo da distância desse pixel branco ao pontos guardados em centro_furos(col,row) [inicializado em (0,0)];

- Se a distância calculada for maior que um valor arbitrado para raio do furo -----> furo novo;

- Toda a orla é percorrida e é calculado o centro do furo pela equação matemática da circunferência;

- Centro calculado guardado no vector centro_furos;

- Se a distância for inferior ao valor arbitrado para raio do furo -----> furo já indentificado;

»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»» Cálculo do centro geométrico do alvo ««««««««««««««««««««««««««««««««

Processos:

- Binarização da imagem original (alvo), com um threshold apropriado (alvobw);

- Remoção dos furos da imagem original;

- Dilatação da imagem (furosbw)----->(furosbwd) ; //remoção do branco (cartão rasgado) em redor do furo

- Operação lógica AND entre a imagem original binarizada e a obtida anteriormente AND(furosbwd,alvobw) imagem obtida (furosbwnew);

- Identificação de todos os anéis do alvo;

- Dilatação em algumas unidades de pixel da imagem (furosbwnew) para unir anéis que possam estar eventualmente partidos (furosbwnewd)

- Esta operação é realizada tanto para aneis brancos em fundo preto, como para aneis pretos em fundo branco. Para trabalhar os aneis pretos cria-se uma imagem auxiliar negada da original aplicando-se a mesma operação de dilatação;

- Através da operação lógica XOR entre a imagem original e imagem auxiliar obtemos uma imagem final com todos os anéis dilatados a branco em fundo preto;

- Para obter as orlas dos anéis identificados basta dilatar de uma unidade toda e imagem e aplicar a operação XOR entre as duas imagens obtendo-se a imagem (aneis_alvoedge);

- Varrer a imagem (aneis_alvoedge) do centro da imagem para fora num ângulo de 45graus, e procurar por um branco pertencente a uma orla de um anel;

- Encontrado o pixel branco de um anel é percorrida toda a orla do anel, e através da equação da circunferencia é calculado o seu centro;

- Cada centro é guardado no vector (alvocentro), e cada raio calculado é guardado no vector (raio);

- Como são calculados os raios dos anéis interiores e exteriores de cada anel, é determinado o anel médio entre os dois que representa o anel real e são guardados em (raios_medios)

- Depois de varrida a imagem completa é calculado um centro médio a partir de todos os centros calculados (alvocent);

- É aqui calculada a resolução obtida dividindo os milimetros reais do raio do anel maior do alvo pelo numero de pixeis obtidos do mesmo anel no programa;

Neste ponto do programa já são conhecidos os centros de cada furo guardados em (centro_furos), e o centro do alvo guardado no vector (alvocent), e o raio de cada anel determinado em (raios_medios) ;

Podemos proceder à classificação de cada tiro;

»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»» Calculo da pontuação obtida em cada tiro ««««««««««««««««««««««««««««

Processos:

- Através do número de aneís identificados, verificar se faltou identificar algum anel; utilizar método dos minimos quadrados;

- Calcular a distância entre a centro do alvo e o centro de cada furo subtraida do raio da munição;

- verificar em que intervalo de valores se encontra a distância calculada.

----------Código desenvolvido em matlab----------

```
////////////////////////////////////////////// Função vermelho_no_alvo//////////////////////////////////////////////

alvo = imread('pistola10_5.jpg');
furos(:,:) = ((255 - alvo(:,:,1) + alvo (:,:,2) + alvo(:,:,3)) > 180 ) * 255;

////////////////////////////////////////////// Função centro_furos //////////////////////////////////////////////
furosbw = im2bw(furos);
se = strel('disk',1);
furosbwd= IMDILATE(furosbw,se);

edgefuros=xor(furosbw,furosbwd);
size(furos);
ntiros=0;
tiroscentro=[0,0];

for row = 100:10:(dim(1)-100)
   for col = 380:(dim(2)-380)
      if edgefuros(row,col)
         distancia
         if d==0
            ntiros=ntiros+1 ;
            contour = bwtraceboundary(edgefuros, [row, col], 'N', 8, inf);
            if max(length(contour)<4) disp([row,col]);end
           abc=[contour(:,2)contour(:,1)ones(length(contour(:,2)),1)]\[-
 (contour(:,2).^2+contour(:,1).^2)];
            a = abc(1); b = abc(2); c = abc(3);
            tirocentro(1)= -a/2;
            tirocentro(2)= -b/2;
            tiroscentro=[tiroscentro;tirocentro];
         else
```

```
        end
      end
    end
end
clear se furosbwd dim ntiros contour abc a b c
////////////////////////////////////////////////////Função distância ////////////////////////////////////////////////////
dimc=size(tiroscentro);
ii=dimc(1);

for i=1:ii
    aa=tiroscentro(i,2);
    bb=tiroscentro(i,1);
    dist(i) = ((col-bb)^2+(row-aa)^2);
    if dist(i)<medida^2
       d=1;
       return
    end
end

d=0;
return
////////////////////////////////////////////////////// Função detecta_aneis ////////////////////////////////////////////////////
furosbw = im2bw(furos);
alvobw=im2bw(alvo,0.5);

cfurosbw=~furosbw;
se1 = strel('disk',dilatacao);
cfurosbwd = IMDILATE(cfurosbw,se1);

furosbwd=~cfurosbwd;

B1=and(alvobw,furosbwd);
C=~B1;
```

```
se = strel('disk',4);
IM1 = IMDILATE(B1,se);
imshow(IM1)
figure

IM2 = IMDILATE(C,se);

BW4=xor(IM1,IM2);
BW5=~BW4;
se1 = strel('disk',1);
BW6 = IMDILATE(BW5,se1);

BW7=xor(BW6,BW5);
imshow(BW7)
figure

dim = size(BW7);
alvocentro=[];
row=round(dim(1)/2)
col=round(dim(2)/2);

rowi=row;
coli=col;

n_aneis=0;
while (n_aneis < 20 && row>2)
if BW7(row,col) == 1
    contour = bwtraceboundary(BW7, [row, col], 'S', 8, inf);
    abc=[contour(:,2)contour(:,1)ones(length(contour(:,2)),1)]\[-(contour(:,2).^2+contour(:,1).^2)];
    a = abc(1); b = abc(2); c = abc(3);
    xc = -a/2;% calcula a localização do centro e do raio
    yc = -b/2;
```

```
elseif BW7(row+1,col) == 1
    contour = bwtraceboundary(BW7, [row+1, col], 'S', 8, inf);
    abc=[contour(:,2)contour(:,1)ones(length(contour(:,2)),1)]\[-(contour(:,2).^2+contour(:,1).^2)];
    a = abc(1); b = abc(2); c = abc(3);
    xc = -a/2;% calcula a localização do centro e do raio
      yc = -b/2;

      verifica
     else
       col=col+1;
       row=row-1;
       end
end
for i=1:n_aneis/2
  raios_medios(i)=(raios(2*i-1)+raios(2*i))/2;
end
DIM=size(raios_medios)
for i=1:(DIM(2)-1)
  dif_raios_medios(i)=raios_medios(i+1)-raios_medios(i);

end
desvio_padrao=std(alvocentro)
centro_do_alvo=mean(alvocentro)
raio_anel1=max(raios_medios);
clear alvobw cfurosbwd furosbwd  C se IM1 IM2 BW4 BW5 se1 BW6 dim row col rowi coli abc a b c
xc yc i DIM

////////////////////////////////////////////////// Função Verifica//////////////////////////////////////////////////
if abs(sqrt((coli-xc)^2+(rowi-yc)^2))< 50
  n_aneis=n_aneis+1;
  alvocent(1)=-a/2;
  alvocent(2)=-b/2;
  alvocentro = [alvocentro;alvocent];
  raio  = sqrt((alvocent(1)^2+alvocent(2)^2)-c);
```

```
 raios(n_aneis) = raio;
 col=col+3;
 row=row-3;

else
  col=col+1;
  row=row-1;

end
return
////////////////////////////////////////////////////// Função calculo_anel ////////////////////////////////////////////////
SS=size(dif_raios_medios);
a=1;
for i=1:SS(2)
 dist_raios_medios(i)=((abs(dif_raios_medios(i)-mean(dif_raios_medios)) < 10)*dif_raios_medios(i))
   if (dist_raios_medios(i)~=0 )
     dist_raios_medios1(a)=dist_raios_medios(i)
     a=a+1;
   end
 end

 SSS=size(dist_raios_medios1);
intervalo=sum(dist_raios_medios1)/SSS(2)

resolucao=largura_anel/intervalo

clear SS SYS i a SSS SYST
////////////////////////////////////////////////////// Função Pontuacao ////////////////////////////////////////////////////

clear i x y
dim=size(tiroscentro);
ii=dim(1);
score=[0 0 0 0 0];
```

```
x=2;
y=1;
s=9.9;
i=2;
while i<=ii
  i
d=abs(sqrt((centro_do_alvo(1)-tiroscentro(i,1))^2+(centro_do_alvo(2)-tiroscentro(i,2))^2)-
raio_municao/resolucao);
if d > raios_reais(10,11)/resolucao
    score(i-1)=0
    i=i+1;
elseif ((d > raios_reais(1,1)/resolucao) && (d <= (raios_reais(1,11)/resolucao)))
  score(i-1)=10
  i=i+1;

else

  for x=2:10
    for y=1:10
      if ((d > raios_reais(x,y)/resolucao) && (d <= (raios_reais(x,y+1)/resolucao)))
        score(i-1)=s
        a=tiroscentro(i,1);
        b=tiroscentro(i,2);
        text(a,b, sprintf('%2.3f pontos',s ),'Color','y','FontWeight','bold');
        hold on
      else
        s=s-0.1;
      end
    end
  end
  s=9.9;
  i=i+1;
```

end

end

pontuacao_total=sum(score)

clear dim ii a m n d

Melhoramentos feitos ao algoritmo:

Introdução de thresholds adaptativos a cada imagem em vez de o uso de um fixo.

- Esta melhoria permite tornar o algoritmo mais flexível ao uso de imagens captadas com diferentes intensidades de luz.

Threshold usado na identificação dos vermelhos na imagem:

Amostragem de pontos vermelhos do lado esquerdo direito da imagem:

```
matrix1=(255-alvo(100,200:20:500,1)+alvo(100,200:20:500,2)+...
alvo(100,200:20:500,3));
```

Amostragem de pontos vermelhos do lado esquerdo direito da imagem:

```
matrix2==(255-alvo(siz(1)-100,200:20:500,1)+alvo(siz(1)-100,200:20:500,2)+...
alvo(siz(1)-100,200:20:500,3));

matrix=[matrix1,matrix2];

thresh=mean(matrix)
furos(:,:) = ((255 - alvo(:,:,1) + alvo (:,:,2) + alvo(:,:,3)) > thresh  ) * 255;
```

Threshold usado na binarização da imagem usando o método de Otsu:

```
alvogray= 0.2990*alvo(:,:,1) + 0.5870*alvo(:,:,2) + 0.1140*alvo(:,:,3);
alvobw = im2bw(alvo,graythresh(alvogray));
```

Para uma mais fácil aplicação do algoritmo, foi desenvolvido uma pequena aplicação gráfica utlilizando igualmente o matlab (figura 18.).

Esta aplicação permite-nos:

- escolher a imagem do alvo que desejamos classificar,
- assinalar através de uma caixa de botões a qual das três provas o alvo se refere,
- observar as pontuações de cada tiro impressas na imagem do alvo,
- observar o somatório das pontuações obtidas na prova,
- observar o tempo de processamento da aplicação.

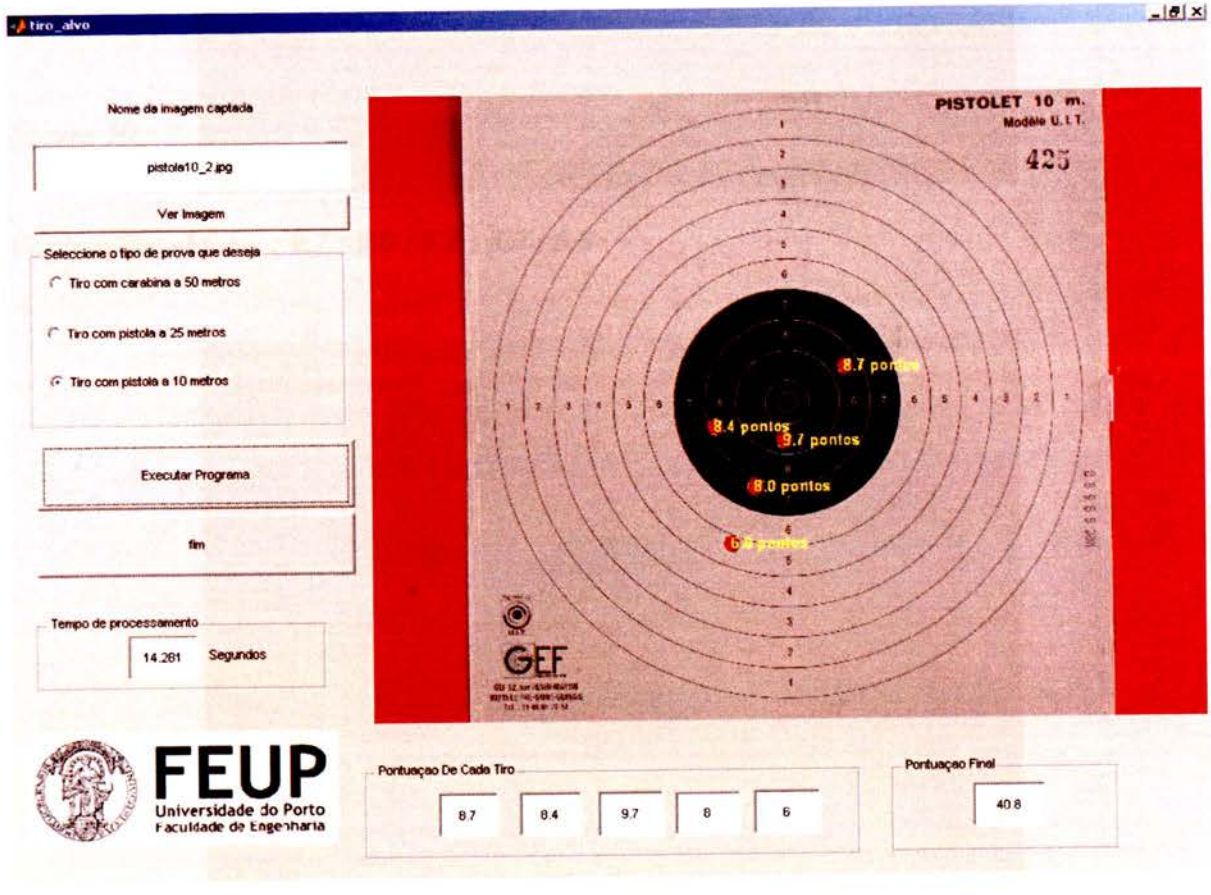

Figura 18.

7 Avaliação de Resultados e Conclusões

Para testar a robustez do algoritmo, foram usadas diversas imagens de alvos de tiro com pistola a 10 metros e de carabina a 50 metros:

Figura 19.

Pontuações obtidas : 9.7 ; 9.0 ; 8.1 ; 8.0 ; 6.8

Figura 20.

Pontuações obtidas : 8.7 ; 8.4 ; 9.4 ; 8.0 ; 6.0

Figura 21.

Pontuações obtidas : 7.1 ; 6.7 ; 8.1 ; 9.6 ; 8.0

Figura 22.

Pontuações obtidas : 6.4 ; 8.7 ; 9.8 ; 8.8 ; 1.0

Figura 23.

Pontuações obtidas : 7.8 ; 9.2 ; 9.5 ; 8.7 ; 6.9

Figura 24.

Pontuações obtidas : 7.4 ; 8.9 ; 10 ; 8.9 ; 8.0

Pontuações obtidas : 10 ; 10 ; 9.1

Figura 25.

Pontuações obtidas : 9.8 ; 9.0 ; 7.4

Figura 26.

A robustez do algoritmo em termos de pontuações em unidades pode ser testada com rigor, já que na realidade são assim que os alvos são classificados. É uma classificação que é dada a olho e qualquer erro que o programa possa dar nas classificações é detectado. A pontuação dada pelo programa desenvolvido com uma casa décimal, não pode ser comprovada que está 100% correcta mas novamente por uma análise a olho, vê-se na globalidade as pontuações dadas pelo programa fazem sentido. Por exemplo quando o programa nos dá uma pontuação de 7,4 7,5 ou 7,6, o rebordo do furo encontra-se a meio da distância entre o anel de pontuação 7 e 8, quando nos dá uma pontuação de 7,8 ou 7,9, o rebordo do furo encontra-se muito perto do anel de pontuação 8 e quando a pontuação de 7,1 e 7,2 o rebordo do furo encontra-se perto do anel de pontuação 7.

Como podemos verificar através dos testes realizados, os resultados obtidos são muito satisfatórios, já que para os 8 alvos em teste apenas três deles apresentaram erros na pontuação assinalados a vermelho nas figuras 20, 23 e 26.

Os primeiros erros foram nos alvos de tiro com pistola a 10metros e foram apenas de uma décima. O segundo erro foi num alvo de tiro com carabina a 50 metros e foi de duas décimas,

Ao todo foram classificados 36 furos apenas errando 2 deles. Isto demonstra que os resultados obtidos são bastante satisfatórios já que não estamos na presença de nenhum erro grosseiro.

Podemos concluir que o algoritmo construido tem condições de continuar a ser trabalhado, e consequentemente melhorado, para que os resultados obtidos possam ser ainda melhores, principalmente em termos de rapidez de processamento actuando sobre as principais funções.

Tempos de processamento das principais funções:

Cálculo dos furos - 1,187seg

Operação Binarização - 0,265

Operação Dilatação - 1,313seg

Operação XOR - 0,25seg

Cálculo centro furos - 0,328seg

Operação Binarização - 0,265seg

Operação Binarização - 1,329seg

Operação Negação - 0,14seg

Operação Dilatação - 0,875seg

Operação Negação - 0,125seg

Operação AND - 0,172seg

Operação Negação - 0,125seg

Operação dilatação - 0,438seg

Operação XOR - 0,203seg

Operação Negação - 0,14seg

Operação Dilatação - 0,141seg

Operação XOR - 0,203seg

Cálculo anéis - 3,204seg

Referências e Bibliografia

Recurso às toolboxes de processamento de imagem que o Matlab nos disponibiliza, bem como aos programas de demosntração existentes.

www.issf-shooting.org

www.wikipedia.org


FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO
BIBLIOTECA
0000104945