FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Sistema para pontuação em tempo real de um alvo de tiro desportivo

José Francisco de Alpoim Fernandes

VERSÃO DE TRABALHO

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Orientador: Professor Armando Luís Sousa Araújo

27 de Julho de 2016

Resumo

O crescimento exponencial observado nas últimas décadas nos campos da eletrónica e da programação, tem vindo a provocar o aumento da quantidade e qualidade das aplicações práticas desenvolvidas e, inversamente, o decréscimo do custo destas tecnologias. Estes fatores promovem o acréscimo paralelo do número de áreas beneficiadas por este desenvolvimento tecnológico como é o caso do processamento de imagem, uma das áreas abordadas nesta dissertação. O desenvolvimento de métodos de processamento sucessivamente mais poderosos, acompanhado pelos progressos no campo da captação de imagem, tem permitido o aumento da influência deste ramo da engenharia.

Uma das áreas que começa a fazer uso dos benefícios trazidos pelas aplicações baseadas em processamento de imagem é a deteção de impactos. A utilização deste tipo de tecnologias permite aos clubes de menor envergadura financeira, e não só, desfrutar de sistemas de pontuação que conjugam boa precisão com custos financeiros inferiores.

Ao longo deste documento,irá ser apresentado o desenvolvimento de um sistema para a pontuação em tempo real de um alvo de tiro desportivo baseado em visão computacional. São descritas todas as etapas do seu desenvolvimento bem como as tecnologias e metodologias aplicadas. Foram implementadas e comparadas diversas funções destinadas ao tratamento e processamento de imagem, realizados testes de desempenho das mesmas, bem como da qualidade de diferentes câmaras até se atingir a implementação final. Por fim, procede-se à análise dos resultados obtidos que atingiram os requisitos de precisão mínimos definidos.

Palavras Chave: Processamento de Imagem, Deteção de Impactos, Pontuação Automática, *Raspberry Pi*, *OpenCV*, Linguagem C/C++

Abstract

The exponencial growth observed in the last decades in the fields of eletronics and programming, has caused the rise in quantity and quality of pratical applications which are being devoloped and, inversely, the fall of the cost of these technologies. These factors promote the paralel increase of areas beneficially affected by this tecnhological development as is the case of image processing, one of the areas covered in this dissertation. The development of successively more powerful processing methods, joined by the progresses in the field of image capture, has allowed the growth in influence manifested by this engeneering branch.

One area which is starting to make use of such benefits brought forward by image processing applications is impact detection. The use of this kind of technologies allows smaller clients, economically speaking, to take pleasure in the use of precise and financially accessible scoring systems.

Throughout this document, the development of a real time automatic scoring system based on computer vision is described. All stages of its development as well as the methodologies and technologies used are also described. Different image processing functions were implemented and compared, their performances tested and the quality of different cameras was studied until the final implementation was reached. Finally the results which fulfilled the minimum precision requisites are analysed.

Keywords: Image Processing, Impact Detection, Automatic Scoring, *Raspberry Pi*, *OpenCV*, C/C++ Programming Languages

Agradecimentos

Gostaria de agradecer, em primeiro lugar, o apoio dado pelo meu orientador, o Professor Armando Araújo, ao longo desta dissertação bem como durante as diferentes ocasiões, durante os anos que cá passei, em que tive o prazer de ser seu aluno.

Agradeço também ao pessoal do laboratório I002 pelas longas horas que passámos juntos e pelo apoio incansável oferecido. Pedro, Fábio, Ricardo, Tiago, Rafael , Afonso, Gabriel e claro o Pedro Galvão, Rosa, Russo e Quim.

Gostaria também de deixar um grande obrigado a quem me ajudou ao longo desta aventura quer com sugestões técnicas quer com apoio moral. Obrigado Pires, Sebe e Joni.

Finalmente gostaria de dar um especial obrigado à minha família que ao longo dos anos tiveram a paciência de aturar as minhas asneiras e me deram a oportunidade de obter este grau de formação. Muito obrigado!

Francisco Alpoim

"Engineers like to solve problems.
If there are no problems handily available, they will create their own problems"

Scott Adams

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

ISSF	*Internation Shooting Sport Federation*
SBC	*Single Board Computer*
M	Mega
K	*Kilo*
mm	Milímetros
cm	Centímetros
m	Metros

Capítulo 1

Introdução

Atualmente, a deteção e classificação de impactos em alvos é utilizada para diversos fins e aplicações tal como competições nacionais e internacionais de tiro desportivo, treino policial e militar e em diversas infraestruturas dedicadas ao disparo de armas como é o caso das carreiras de tiro. A deteção e avaliação de impactos é realizada com base num conjunto limitado de técnicas disponíveis. Uma delas é a classificação manual, em que a pontuação do impacto é realizada através da avaliação visual por um operador humano. Outra técnica utilizada é a deteção acústica que se baseia no cálculo do tempo de propagação das ondas sonoras causadas pelo impacto do projétil no alvo. Este tempo de propagação é medido por um grupo sensores que em conjunto efetuam uma triangulação de forma a determinar a posição do impacto.

A utilização dos métodos de deteção de impactos usados atualmente trazem, como seria de esperar, vantagens e desvantagens associadas. No caso da deteção manual, a principal vantagem é sobretudo económica, dado a não utilização de equipamento especializado. No entanto a sua utilização implica limitações temporais devido ao elevado número de alvos a classificar. Um impacto é, geometricamente, uma área no alvo e não um ponto. Para efetuar uma pontuação precisa do tiro, é necessário determinar a distância do impacto ao centro do alvo. Dado que a área, devido às características físicas associadas ao disparo, é irregular, tornando-se muitas vezes extenuante classificar todos os alvos associados a uma prova. Por outro lado, num evento onde é realizado mais que um disparo, existe a possibilidade dos impactos se sobreporem. Neste caso a deteção do centro da área de um impacto torna-se ainda mais complexa devido ao fato de apenas existir visibilidade parcial da mesma.

No que toca aos meios acústicos, uma grande parte dos problemas associados à precisão das deteções é eliminado. Em contra partida, a implementação destes sistemas traz consigo custos relativamente elevados dado que este equipamento especializado requer, em grande parte dos casos, um operador profissional. Para além disso, o uso de técnicas acústicas requer uma modificação dos alvos como a instalação de uma câmara acústica e toda a eletrónica associada trazendo consigo um atravancamento de material. Os sensores utilizados, devido à sua sensibilidade e proximidade com os impactos, terão de ser bem protegidos devido à possibilidade de um projétil os poder danificar. Estas condições fazem com que este tipo de sistemas seja pouco favorecido por consumidores

de baixo poder financeiro. Dado isto, o desenvolvimento de um sistema de deteção automático em tempo real baseado em visão torna-se uma alternativa atraente quando comparado com as soluções em vigor atualmente, já que concilia o facto de ser economicamente acessível com um desempenho técnico satisfatório. Para além disso, os métodos baseados em visão trazem consigo a vantagem de utilizar *software* e *hardware* computacionais que são comodidades cujo preço tem vindo a descer acentuadamente ao longo das últimas décadas.

Note-se que existem sistemas de classificação baseados em visão, mas não em tempo real. Nestes sistemas os alvos são inseridos manualmente numa máquina que determina a informação. Esta é fornecida num visor LCD ou passada visa comunicação série para um PC. Um exemplo deste tipo de sistemas é a máquina *Rika easy score*[9] ou a *DISAG RM-III Universal*[10].

1.1 Objetivos

O objetivo da dissertação será o desenvolvimento de um sistema capaz de devolver, em tempo real, a pontuação de alvos de tiro desportivo das modalidades P10 e C10 que cumpram os regulamentos exigidos pela ISSF (*International Shooting Sport Federation*)[11] , pelo que estes não podem nem devem sofrer alterações significativas que invalidem os critérios definidos por esta organização. Tal identificação deverá ser realizada a partir de visão por computador. O sistema deve permitir a visualização, em tempo real, de vários dados, tais como a pontuação atribuída a cada impacto, a pontuação total e o ponto médio dos disparos efetuados. Posteriormente, o algoritmo de identificação de impactos deverá ser exportado e implementado num computador de placa única que irá fazer o processamento e avaliação do tiro com o auxílio de uma câmara com resolução adequada. Será relevante referir que o sistema deverá ser robusto e economicamente acessível e o processamento associado preciso e rápido.

Capítulo 2

Revisão Bibliográfica

Neste capítulo pretende-se realizar uma revisão do conhecimento científico e técnico existente na área do processamento de imagem na qual a deteção de impactos se insere tais como os métodos ou algoritmos frequentemente utilizados. Dado que tal deteção tem como objetivo final a determinação da pontuação de disparos realizados com base nas regras da modalidade de tiro desportivo ISSF, irá ser feita uma breve exemplificação do que esta disciplina consiste.

2.1 Tiro Desportivo

O tiro desportivo ISSF é uma modalidade com relevância histórica dado ser uma modalidade olímpica desde a primeira edição moderna dos Jogos Olímpicos em 1896. A sua prática visa testar o atirador quanto à velocidade de manejo e precisão com uma arma de fogo. É uma modalidade com inúmeras disciplinas dado a grande variedade de distâncias de disparo e de tipos, ou calibres, de arma utilizados. Tal como a generalidade das modalidades existentes, encontra-se sujeita a um conjunto de regras bem definidas. A organização responsável pela regulação e definição das regras e normas em vigor é a ISSF[8].

2.2 Processamento de Imagem

O ser-humano poderá ser descrito como uma "máquina"complexa com uma capacidade de interpretar e observar o mundo em redor de forma extremamente eficaz. Um dos sistemas que a constituem é a perceção visual que é capaz de adquirir, processar e interpretar informação tais como cores, formas, texturas ou movimento. Com o desenvolvimento das tecnologias associadas à computorização e sensorização, um dos maiores desafios encarados pelos engenheiros de diversas áreas passa pela implementação destas características humanas em máquinas. A vertente da engenharia responsável por esta implementação associada à visão é o processamento de imagem, que nos tempos correntes é utilizado para o benefício de várias áreas.

- Inspeção visual automática

O processamento de imagem torna-se bastante útil na identificação de falhas em componentes de sistemas eletrónicos ou eletromecânicos. Geralmente, componentes deste tipo com falhas geram uma energia térmica superior que poderá ser evidenciada numa imagem de infravermelhos. A análise desta imagem é posteriormente realizada de forma a identificar os componentes em questão. A indústria metalúrgica também beneficia da utilização de técnicas de análise de imagem na deteção de irregularidades na superfície de objetos metálicos numa linha produção.[12]

- Interpretação de cenas por sensorização remota

  A existência de recursos naturais de cariz mineral ou hidrológico num determinado local do globo poderá ser averiguada remotamente com recurso a sensores colocados em satélites ou aeronaves que posteriormente transmitem esse sinal para processamento. A ideia baseia-se no simples facto de diferentes materiais apresentarem diferentes tonalidades. Tal fenómeno poderá ser usado para medir, por exemplo, a percentagem de gelo numa calota polar ou a própria existência de água sobe a forma líquida noutro planeta.[12]

- Biomedicina

  Este ramo da medicina utiliza diversos dispositivos baseados em imagem com o intuito de realizar um diagnóstico médico do interior do corpo humano. Entre as tecnologias utilizadas destacam-se os raios-X, a tomografia computadorizada, ilustrada na figura 2.1 e os ultrassons.[12] Será então facilmente verificável os benefícios que o tratamento e análise de imagem poderão oferecer a esta disciplina médica tais como:

  1. Localização de objetos ou regiões como os diferentes órgãos do corpo
  2. Realizar a medição de objetos de interesse como tumores
  3. Interpretação da constituição dos objetos em questão para diagnóstico

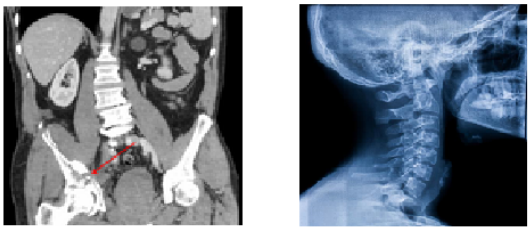

Figura 2.1: Exemplos de imagens raio-X (direita) e de uma tomografia computadorizada (esquerda)[1][2]

- Defesa

  A aplicação de técnicas de processamento de imagem para fins de vigilância é uma área de estudo de importância relevante. Existe uma constante necessidade de realizar monitorização quer de fronteiras terrestres quer marítimas através de meios aéreos por exemplo. No

caso da patrulha de fronteiras marítimas, as técnicas de processamento de imagem poderão ser utilizadas na localização de formações navais ou de embarcações isoladas. O estudo posterior das imagens captadas permite a identificação de parâmetros relevantes como o tamanho das embarcações, a distância a que se encontram da costa e até mesmo a velocidade a que deslocam.[12]

- Compressão de imagens e vídeo

  A compressão de imagem e vídeo é uma área em constante crescimento que continua a verificar desenvolvimentos tecnológicos que desempenham um papel importante no sucesso das comunicações e aplicações multimédia. Apesar do custo do armazenamento de ter decrescido significativamente nas últimas décadas, os requisitos para esse mesmo armazenamento têm crescido de forma exponencial à medida que a qualidade das imagens e vídeo tem aumentado. A título comparativo, veja-se o exemplo da televisão digital. O requisito para armazenar um sinal de alta definição com resolução de 1280 x 720 a 60fps é de 1250 Megabits por segundo. A transmissão direta de imagens ou vídeo com este nível de qualidade sem a aplicação de compressão é uma tarefa de dificuldade notória. No entanto, tanto nas imagens estáticas como nas imagens de vídeo, existem redundâncias na informação visual que poderão ser aproveitadas pela área do processamento da imagem. Estas redundâncias surgem do facto de muitas vezes, numa região homogénea de uma imagem, existirem pixeis vizinhos com variações pouco significativas no seu valor que são indistinguíveis para o olho humano. Da mesma forma, dois *frames* consecutivos numa sequência de vídeo são bastantes similares e apresentam redundâncias temporárias. O objetivo das técnicas de processamento de imagem aplicadas a esta área baseiam-se na redução ou remoção destas redundâncias visuais de forma a representar *frames* de imagens com um número de bits significativamente menor e desta forma reduzir os requisitos de armazenamento e de largura de banda.[12]

Outra das áreas de aplicação das tecnologias de processamento de imagem corresponde à deteção de impactos que será discutida na secção seguinte.

2.3 Deteção de Impactos

A deteção da localização de impactos em alvos para fins de avaliação é uma área relativamente recente e em desenvolvimento constante. Atualmente são usados diversos métodos de deteção, cada um com vantagens e desvantagens características. As técnicas mais relevantes atualmente em vigor correspondem à deteção manual e à deteção acústica. Os métodos de deteção manual, baseados em operadores humanos, ainda têm uma grande relevância devido às dificuldades que advêm da utilização dos métodos acústicos como preço elevado de aquisição e manutenção de equipamentos. Mais recentemente começam a surgir os métodos baseados em visão que, aproveitando o desenvolvimento explosivo das tecnologias de processamento de imagem e a diminuição dos custos de *hardware* e *software*, procuram oferecer uma solução economicamente mais viável sem perder a precisão e qualidade necessárias.

2.3.1 Métodos de Deteção Manuais

Os métodos de deteção manuais fazem uso de um operador humano de forma a avaliar a posição do impacto no alvo. É, como seria de esperar, o método mais antigo e por conseguinte simples. No entanto é ainda utilizado em larga escala. O princípio é básico e consiste em que um juiz realize uma deteção visual do disparo. Em caso de dúvida o mesmo recorre a dispositivos óticos e de medição que calculam a distância do impacto ao centro do alvo de forma a atribuir uma pontuação. Tal como foi anteriormente referido, este tipo de deteção está sujeito a limitações sobretudo no que toca ao tempo de classificação. A avaliação de impactos sobrepostos é outro fator que dificulta a avaliação dado que nesta situação a área de um ou mais impactos não se encontra totalmente visível.

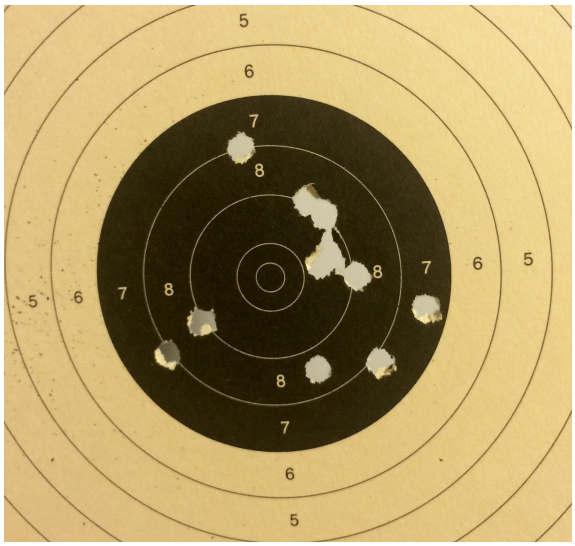

Figura 2.2: Alvo oficial com diveros impactos

A figura 2.2 pretende ilustrar a ocorrência de impactos muito próximos entre si e as dificuldades inerentes à determinação das suas pontuações individuais.

2.3.2 Métodos de Deteção Baseados em Acústica

O aparecimento de métodos alternativos à deteção manual surgiu como forma de minimizar os erros inerentes às avaliações humanas. Os métodos acústicos são um deles e surgiram com os avanços tecnológicos conseguidos na área da acústica. Neste método, a posição do impacto no alvo é determinada com recurso ao cálculo do tempo de propagação das ondas sonoras que o mesmo causa. Estas ondas são detetadas por um grupo de sensores que em conjunto efetuam uma triangulação de forma a calcular a distância relativa do ponto de impacto. A figura 2.3 pretende ilustrar graficamente a posição de um impacto relativamente aos três sensores utilizados e a triangulação resultante.

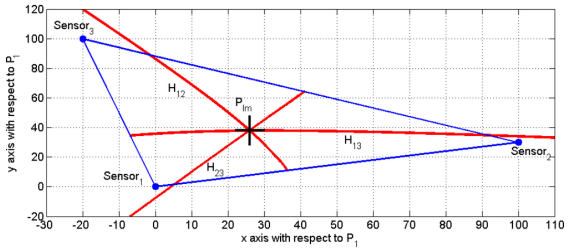

Figura 2.3: Triangulação da posição de um impacto baseada em acústica[3]

A utilização de métodos acústicos traz consigo vantagens no que toca à melhoria no tempo da avaliação da pontuação do impacto mas envolve maiores gastos dado que a aquisição de sistemas deste tipo requer instalação e manutenção por parte de mão de obra especializada. O sistema em si requer também que haja modificação dos alvos de forma a poder ser implementado (câmara acústica) e os sensores e eletrónica utilizados requerem proteção devido à proximidade física com os impactos.

2.3.3 Métodos de Deteção Baseados em Visão

Com o desenvolvimento das áreas de processamento de imagem e micro-controladores, que nas últimas décadas verificaram um crescimento explosivo, paralelo ao da computação e micro-eletrónica, surgiu a possibilidade de aplicar as vantagens que este crescimento e desenvolvimento tecnológico permitem, à área da deteção e avaliação de impactos. Este tipo de sistemas procura conjugar as vantagens dos métodos manuais (economicamente viáveis) com os métodos acústicos (precisos) sendo uma alternativa com futuro e que começa agora a ser seriamente estudada. O princípio de funcionamento é baseado na aquisição de uma imagem, conjunto de imagens ou vídeo e o seu correspondente processamento onde normalmente se aplicam algoritmos específicos, como por exemplo algoritmos de deteção de formas geométricas, de forma a determinar a posição do impacto. De seguida serão abordados de forma mais aprofundada alguns dos algoritmos mais relevantes e utilizados na deteção de impactos. A figura 2.4 ilustra um sistema de classificação de pontuação baseado em visão e os principais elementos que o constituem.

O processamento de imagens é uma área onde os resultados obtidos são relativamente dependente das condições de luminosidade onde o sistema se insere. Como tal os sistemas baseados em visão são geralmente acompanhados de sistemas de iluminação próprios ou então são implementados em localizações com luminosidade controlada. Por oposição, existe a possibilidade de implementar este tipo de sistemas ao ar livre onde não existe controlo das condições de iluminação dos alvos, sendo para tal necessário implementar algoritmos próprios de forma a lidar com este tipo de situações.

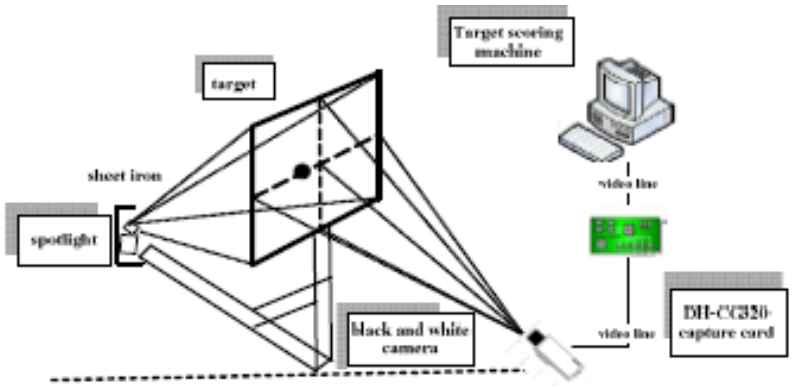

Figura 2.4: Exemplo de um sistema de baseado em visão[4]

2.3.3.1 Transformada de *Hough*

A transformada de Hough é uma técnica normalmente utilizada no campo do processamento digital de imagem com o intuito de detetar formas geométricas tais como linhas retas, curvas, círculos, elipses e até o contorno de polígonos. Tem como princípio a representação de um conjunto de pontos definidos inicialmente num espaço euclidiano num outro espaço permitindo a deteção de pontos que constituem uma determinada figura geométrica[13].

Figura 2.5: Deteção de círculos numa imagem usando a transformada de Hough [5]

Na figura 2.5 pode ser visualizada a deteção do contorno circular presente na imagem, representado a verde bem como o seu ponto central um quadrado laranja.

2.3.3.2 Deteção de Borda - *Edge Detection*

A deteção de borda é uma técnica utilizada em processamento de imagem digital que tal como o nome indica é utilizada para encontrar a borda ou fronteira de um ou mais objetos dentro de uma imagem. Como poderá ser observado na figura 2.6 esta técnica de processamento procura áreas da imagem onde existam descontinuidades de luminosidade de forma a identificar as zonas de fronteira[14].

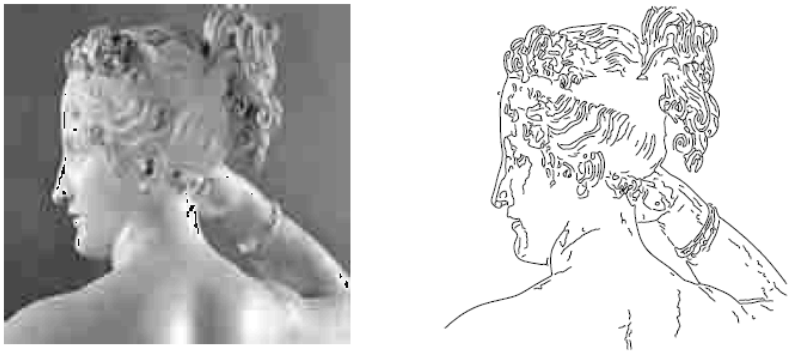

Figura 2.6: Aplicação da técnica *Edge Detection*(direita) a uma imagem digital [15]

2.3.3.3 Filtro de Mediana - *Median Filter*

O filtro de mediana é utilizado como forma de combater o ruído inerente à digitalização de uma imagem. Esta técnica calcula a mediana do valor dos píxeis que rodeiam um determinado píxel de forma a decidir se este é representativo dos seus arredores. Se a técnica decidir que tal não se verifica (ou seja, se for considerado ruído) o valor do píxel é substituído pela mediana daqueles que o rodeiam procurando desta forma suavizar a imagem [16].

2.3.3.4 *Image Thresholding*

O *thresholding* é um dos métodos de segmentação de imagem mais simples. A segmentação de imagem é uma tarefa essencial na área do processamento de imagem e visão computacional. É utilizada com o intuito de dividir ou, segmentar, uma imagem num determinado número de regiões facilitando a sua análise e/ou tratamento. O resultado desta técnica de processamento é uma imagem binária cujo os pixeis apresentam o valor de zero (preto) ou um (branco). A figura 2.7 pretende ilustrar a aplicação deste método.

Figura 2.7: Aplicação da técnica de*Image Thresholding*(direita) a uma imagem digital

2.3.3.5 Retificação Geométrica

A retificação geométrica é uma técnica de manipulação digital de imagem utilizada para corrigir o efeito de distorção espacial introduzido pelo desfasamento axial de uma câmera, por exemplo, em relação ao objeto captado. O objetivo é introduzir uma distorção inversa para compensar a distorção original e desta forma obter uma imagem digital o mais próxima possível da original.

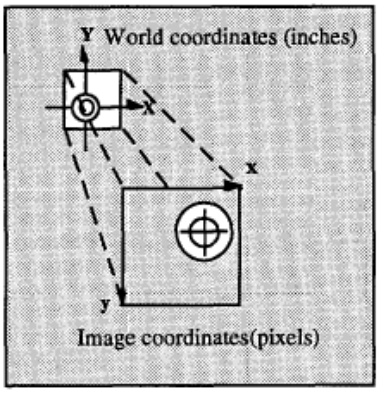

Figura 2.8: Exemplificação do desfasamento axial entre câmera e objeto captado [17]

Capítulo 3

Sistema de Deteção de Impactos

Como já foi mencionado no capítulo 1.1 deste documento, o sistema desenvolvido deverá ser uma alternativa viável aos demais sistemas de deteção de impactos existentes atualmente em vigor. Para o bom funcionamento do sistema torna-se importante utilizar material bem adequado e capaz de lidar com as especificações técnicas impostas de uma forma eficaz. Neste capítulo será debatida e exposta a escolha do material utilizado ao longo do desenvolvimento do projeto.

3.1 *Hardware*

3.1.1 Computador de Placa Única - SBC

Um computador de placa única, também conhecido como SBC do inglês *single board computer*, é como o nome sugere um computador onde todos os componentes necessários ao seu funcionamento se encontram numa placa de circuito impresso. Devido ao seu tamanho reduzido, é um componente bastante utilizado em sistemas de controlo e medida. O computador de placa única será o elemento central do sistema de deteção e terá de efetuar o processamento e tratamento da imagem proveniente da câmara. A escolha deste componente recaiu sobre o facto de que o processamento e algoritmia têm de ser realizados com rapidez já que a nível competitivo é de grande importância que possam ser analisados vários disparos num intervalo de tempo reduzido. Tendo em conta os fatores em cima enunciados, a escolha do SBC recaiu sobre o *Raspberry Pi*. Para além de possuir uma velocidade de processamento satisfatória, o facto de possuir uma câmara nativa dedicada com possibilidade de adaptação através do uso de um *shield*, tornaram a escolha do *Raspberry* bastante atrativa.

3.1.2 Câmara

Este componente do sistema desempenha o papel de captar o alvo onde os impactos ocorrem, tirando *snapshots* periódicos para de seguida transmitir a informação visual para que o SBC faça o processamento e tratamento da imagem.

A câmara está indiretamente sujeita, do ponto de vista técnico, a algumas limitações, nomeadamente no que toca à resolução mínima que lhe é exigida. Os alvos em que irão decorrer os testes têm dimensões variáveis de acordo com o calibre da arma que, de igual modo, irá influenciar a distância de disparo.

No caso dos alvos de 10 metros de carabina, C10, estes possuem um diâmetro de 45.5 mm, ou seja, um raio de 22.75 mm desde o centro até ao anel de pontuação mais exterior (figura 3.2). Assim, de forma a determinar a pontuação do disparo com uma resolução decimal é necessário dividi-lo em cem partes iguais. Logo teremos de resolver 0.2275 mm por pixel, caso contrário um cálculo decimal preciso não será possível.

A câmara nativa do *Raspberry*, representada na imagem 3.1, cumpre os requisitos necessários, dispondo de uma resolução de 5M pixeis (dimensão de imagem 2594x1944 pixeis).

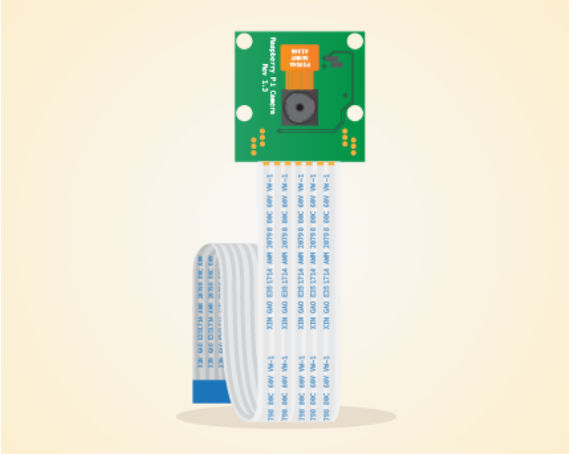

Figura 3.1: Câmara nativa do SBC *Raspberry Pi* [6]

Após testes ao seu desempenho, verificou-se que apesar de a câmara do *Raspberry Pi* apresentar uma resolução suficiente para a determinação das pontuações dos disparos, o facto de não possuir *zoom* ou foco regulável faz com que a sua utilização não seja a mais adequada. Isto porque para obter imagens com nitidez aceitável, a distância ao alvo deverá ser muito grande. Isto faz com que a imagem, para além do alvo, capte uma área sem interesse desperdiçando assim uma grande quantidade de pixeis.

Como alternativa, foi utilizada uma câmara compatível com qualquer uma das versões de *Raspberry* existentes, com uma resolução idêntica à câmara nativa deste SBC (5M px) mas com a possibilidade de regulação da distância de focagem.

3.1.3 Alvos

Os alvos que irão ser utilizados para teste correspondem a alvos oficiais da caetgoria P10 e C10, que seguem as normas impostas pela *ISSF*. Os alvos encontram-se dimensionados para a prova de pistola de ar e carabina a 10m. A figura 3.2 representa um dos alvos em questão.

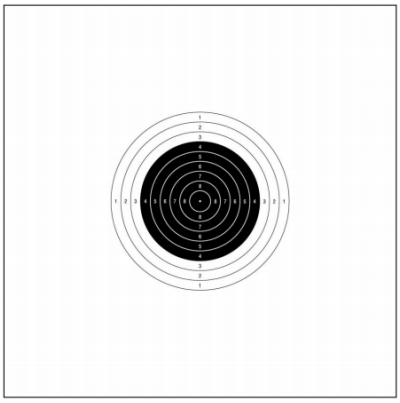

Figura 3.2: Alvo de categoria P10

As tabelas 3.1 e 3.2 ilustram as dimensões associadas aos alvos da categoria C10 e P10 respetivamente.

10 Ring	0.5 mm	(± 0.1 mm)	5 Ring	25.5 mm	(± 0.1 mm)
9 Ring	5.5 mm	(± 0.1 mm)	4 Ring	30.5 mm	(± 0.1 mm)
8 Ring	10.5 mm	(± 0.1 mm)	3 Ring	35.5 mm	(± 0.1 mm)
7 Ring	15.5 mm	(± 0.1 mm)	2 Ring	40.5 mm	(± 0.1 mm)
6 Ring	20.5 mm	(± 0.1 mm)	1 Ring	45.5 mm	(± 0.1 mm)
Inner ten: when the white ring(dot) has been shot out completely					
Black from 4 to 9 rings = 30.5 mm (± 0.1 mm)					
Ring thickness: 0.1 mm to 0.2 mm					
Minimum visible size of target card: 80 mm x 80 mm					

Tabela 3.1: Distâncias dos diversos anéis ao centro do alvo na categoria C10 [8]

10 Ring	11.5 mm	(± 0.1 mm)	5 Ring	91.5 mm	(± 0.5 mm)
9 Ring	27.5 mm	(± 0.1 mm)	4 Ring	107.5 mm	(± 0.5 mm)
8 Ring	43.5 mm	(± 0.2 mm)	3 Ring	123.5 mm	(± 0.5 mm)
7 Ring	59.5 mm	(± 0.5 mm)	2 Ring	139.5 mm	(± 0.5 mm)
6 Ring	75.5 mm	(± 0.5 mm)	1 Ring	155.5 mm	(± 0.5 mm)
Inner ten: 5.0 mm ± 0.1 mm					
Black from 7 to 10 rings = 59.5 mm (± 0.5 mm)					
Ring thickness: 0.1 mm to 0.2 mm					
Minimum visible size of target card: 170 mm x 170 mm					

Tabela 3.2: Distâncias dos diversos anéis ao centro do alvo na categoria P10 [8]

3.2 *Software*

Em paralelo com o *hardware* utilizado, a implementação correta do código ou algoritmos de processamento de imagem é uma componente fulcral do projeto. A escolha das bibliotecas e

ambientes de desenvolvimento mais adequados para a tarefa em mãos é um passo que não deverá ser desprezado dada a sua importância e relevância ao longo do desenvolvimento desta dissertação.

3.2.1 *OpenCV*

Existem atualmente várias bibliotecas destinadas a tornar o processamento de imagem acessível e eficaz. De entre as existentes o *OpenCV* foi a escolha efetuada pelo facto de suportar diversas linguagens de programação como C++, C, Java ou Python. A sua compatibilidade com diversos sistemas operativos como o Linux, na qual o *Raspberry Pi* se baseia, tornam o uso desta biblioteca bastante atrativo [18].

3.2.2 *Visual Studio*

O ambiente de desenvolvimento da algoritmia escolhido recaiu no *Visual Studio*[19]. Esta plataforma de programação é amplamente utilizada na criação de aplicações para *Windows*, *Android* e *iOS* disponibilizando inúmeras ferramentas úteis como *debugging* avançado e suporte de diversas linguagens de programação. Para além demais, oferece compatibilidade com a biblioteca *OpenCV*.

Capítulo 4

Algoritmia

A escolha e implementação dos algoritmos de processamento de imagem mais adequados ao projeto é uma etapa de importância vital para o bom funcionamento do sistema. Neste capítulo pretende-se apresentar os algoritmos e funções utilizadas e ao mesmo tempo explicar o seu funcionamento.

Após a análise e estudo da revisão bibliográfica acerca de algumas das funções ou algoritmos mais utilizados, foi elaborado um plano para a estrutura do programa bem como para a linguagem de programação mais adequada a ser implementada. Uma das primeiras decisões a ser tomadas foi a utilização da biblioteca de processamento de imagem *OpenCV* que traz a vantagem de ser compatível com os sistemas operativos *Windows* e *Linux* e que para além disso encontra-se disponível para o *Raspberry Pi*. Esta biblioteca dispõe de interfaces em C, C++, *Python* e Java.

A linguagem de programação na qual o código foi implementado corresponde à linguagem C/C++ que foi escolhida em detrimento à linguagem *Python*. Isto porque apesar de ambas serem suportadas pela biblioteca *OpenCV*, a primeira apresenta a vantagem de obter velocidades de processamento mais rápidas em relação à segunda.

4.1 Estrutura do Código

Nesta secção é oferecida uma representação visual da sequência de etapas realizadas ao longo da implementação do código conjuntamente com uma descrição e explicação de cada uma destas mesmas etapas.

4.1.1 Descrição da Estrutura

1. **Inicialização de Variáveis**

   Tal como o nome indica, nesta etapa do código são inicializadas as variáveis globais que irão ser utilizadas ao longo do programa bem como a inclusão dos ficheiros necessários ao funcionamento da biblioteca de processamento de imagem *OpenCV*.

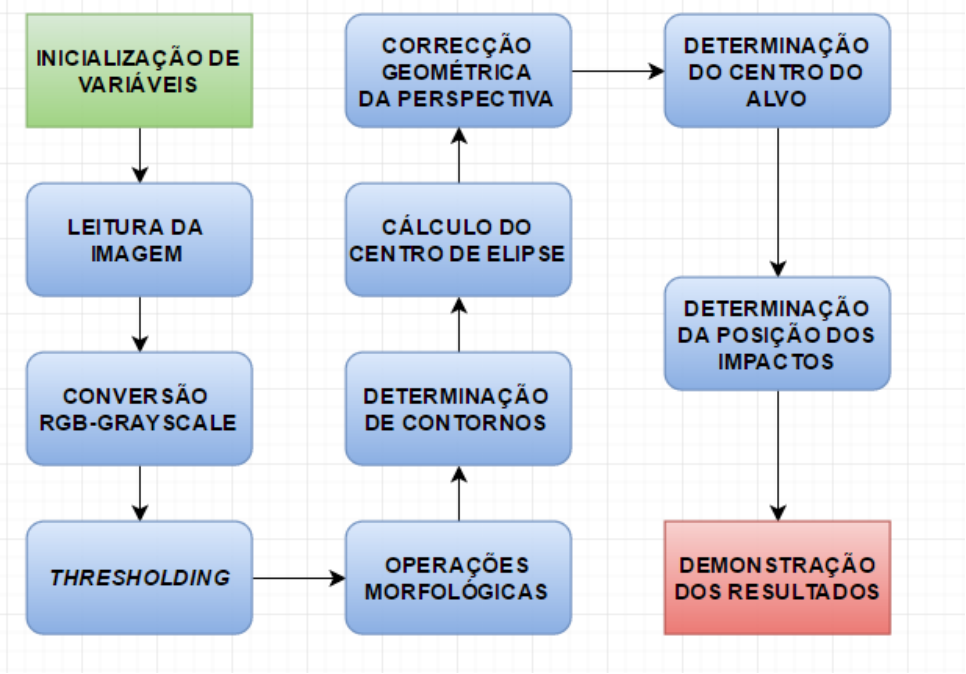

Figura 4.1: Diagrama sequencial das funções implementadas

É também nesta etapa realizada a criação de algumas das matrizes que irão alocar as imagens ao longo do programa.

2. **Leitura da Imagem**

   A biblioteca *OpenCV* oferece a possibilidade de realizar a leitura de imagens através da função *imread*[20]. Esta função carrega uma imagem sobe a forma de uma matriz, permitindo o seu processamento no decorrer do programa. Caso não seja possível carregar a imagem (devido à inexistência do ficheiro em questão ou formato inválido) a função retorna uma matriz nula.

3. **Conversão RGB-*Grayscale***

   Aqui é feita a conversão da imagem do modelo de cores RGB para *grayscale*. Neste modelo os pixeis que constituem a matriz da imagem apenas retêm informação relativa à sua intensidade que pode variar entre 0(preto) e 1(branco), o que resulta numa imagem em tons de cinza. Esta transformação irá facilitar a aplicação das funções de morfologia e de *thresholding* que irão ser descritas de seguida.

4. ***Thresholding***

   Após a conversão da imagem para o modelo *grayscale* a imagem é segmentada com recurso à função *threshold*[21] resultando numa imagem binária tal como foi descrito em 2.3.3.4. Esta função atribui um de dois valores ao campo de intensidade dos pixeis da imagem.

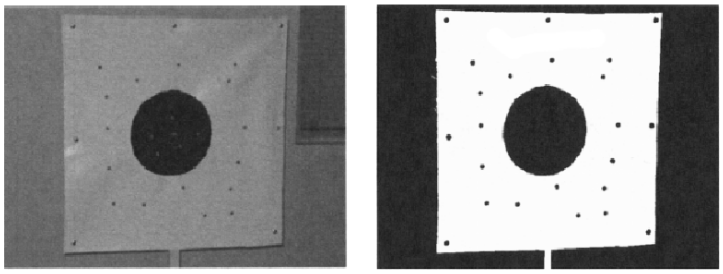

Figura 4.2: Segmentação dos pixeis da imagem pela função *threshold*[7]

Pela análise da imagem 4.2 é possível verificar que o pixel na posição (x,y) da matriz da imagem é sujeito a uma condição de segmentação. Se o seu valor de intensidade for superior a um valor de referência, ou seja, ao valor de *threshold*, o seu novo valor corresponde ao máximo permitido (representado visualmente como branco). Caso contrário, o seu valor passará a ser igual a zero o que corresponde à cor preta.

Esta operação terá influência para o funcionamento otimizado da função de deteção de contornos descrita no item 6.

5. **Operações Morfológicas**

   O recurso às operações de morfologia é um passo cuja a importância não poderá ser desprezada em diversas aplicações de processamento de imagem. A biblioteca *OpenCV* disponibiliza duas funções de morfologia - *erode* e *dilate*[22]. O objetivo destas funções é remover o ruído, ou perturbações, inerentes à maior partes das imagens sendo por isso de grande utilidade no tratamento de imagem digital. O efeito destas funções sobre uma imagem digital poderá ser observado na figura 4.3.

Figura 4.3: Função *dilate* (centro), *erode* (direita) e original (esquerda)

Após a aplicação destas duas operações, é utilizada a função *medianblur*[23] com o intuito de suavizar a imagem. Os efeitos da aplicação desta função foram discutidos previamente

em 2.3.3.3.

6. **Determinação de Contornos**

   Nesta fase do código pretende-se isolar os contornos da imagem, neste caso do alvo, com o intuito de tornar a zona geométrica definida pelo alvo mais pronunciada.

   A função utilizada para este fim é *findcontours*[24] cujo o método de funcionamento passa por guardar os contornos presentes na imagem num vetor de pontos. No entanto, devido a fenómenos de distorção causados pela lente da câmara, notou-se que o rebordo do alvo apresentava uma pequena curvatura. Como tal, foi testado o uso de duas funções de determinação de contornos alternativas: *approxPolyDP*[24] que permite fazer a delimitação da zona de fronteira do alvo tendo em conta a curvatura existente e *convexHull*[24] capaz de detetar defeitos de convexidade presentes numa curva e corrigi-los. A figura 4.4 ilustra o resultado obtido pela utilização da função *approxPolyDP* na determinação dos contornos de um alvo representados pela linha azul.

   A implementação destas duas funções irá ter influência, mesma que pequena, nos resultados finais obtidos que irão ser comparados e discutidos no próximo capítulo.

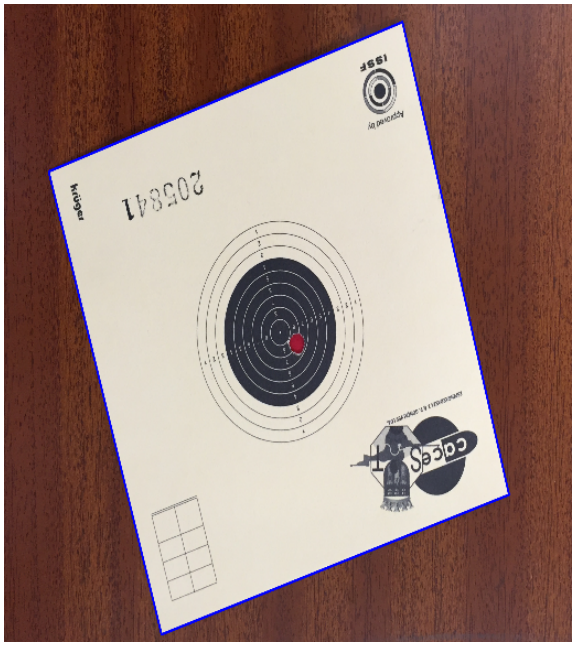

Figura 4.4: Deteção dos contornos do alvo

7. **Cálculo do Centro de Elipse**

   Após a determinação dos contornos da imagem, pretende-se identificar quais destes contornos são os relevantes para o que se pretender realizar. Neste caso, o objetivo passa por encontrar o centro geométrico do centro do alvo, que devido aos fenómenos de distorção geométrica causados pelo inevitável ângulo entre a lente da câmara e o alvo, não será um

círculo como seria de esperar, mas sim uma elipse. O cálculo do centro desta elipse será um ponto relevante para o próximo passo do programa definido no item 8.

8. **Correção Geométrica da Perspetiva**

   Para corrigir os efeitos de distorção geométrica mencionados anteriormente, foram aplicadas as funções *getPerspectiveTransform* e *warpPerspective*[25]. A aplicação deste par de funções é vital para o funcionamento correto do cálculo da pontuação dos disparos, já que sem elas a localização dos impactos bem como o centro do alvo não poderão ser calculados realisticamente.

   A utilização deste par de funções requer como entrada quatro pontos da imagem original. Estes pontos foram calculados com base no centro da elipse que constitui o centro do alvo mencionado no item 7. A imagem corrigida é guardada numa nova matriz com dimensões de largura e comprimento idênticas estando preparada para a determinação dos impactos e centro do alvo.

4.2 Determinação da Pontuação dos Disparos

A determinação do centro do alvo bem como da posição dos impactos são as duas etapas mais complexas no que toca ao número de passos que as constituem, merecendo especial atenção. A implementação conjunta das duas irá resultar na obtenção da pontuação do disparo que irá corresponder à distância relativa entre o impacto e o centro do centro do alvo.

Pretende-se agora demonstrar a estrutura interna destas duas etapas.

4.2.1 Determinação do Centro do Alvo

À semelhança do passo referido em 3 da subsecção anterior, a função para a determinação do centro do alvo começa por aplicar uma transformação do espaço de cores da imagem, desta vez de RGB para HSV.

O espaço de cores HSV, cujas iniciais representam as três variáveis que usa para representar cor: matiz, saturação e valor (ou brilho) (*Hue, Saturation, Value*), é de bastante utilidade na área do processamento de imagem facilitando a segmentação de uma imagem pelas cores nela contida. Em *OpenCV* é possível realizar esta transformação aplicando a função *cvtColor*[21].

Após a conversão cromática é necessário definir o intervalo de valores das três variáveis em cima mencionadas de forma a isolar a cor pretendida através da função *inRange*. No caso dos alvos de tiro desportivo, a cor em questão corresponde à do centro do alvo do alvo que é preto.

Fazendo uso das operações morfológicas de dilatação e erosão 5 e filtro de mediana para suavização, os contornos do centro do alvo são evidenciados.

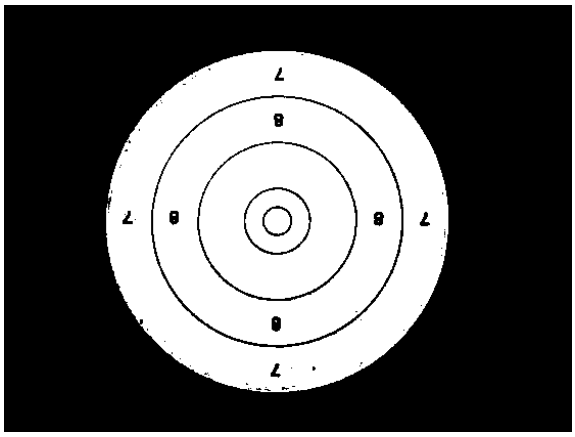

Figura 4.5: Imagem HSV do centro de um alvo

4.2.2 Interpretação da Informação dos Contornos

Após os passos em cima descritos, o objetivo é mais uma vez detetar os contornos da imagem de forma a calcular a posição central do alvo que após a correção de perspetiva corresponde ao centro de um círculo e não de uma elipse.

Existem várias funções que podem ser aplicadas para obter e posteriormente assinalar os contornos, cada uma obtendo resultados aproximados mas não idênticos, o que irá influenciar o resultado final calculado.

Sendo assim, foram analisados e implementados diferentes funções e estratégias e os resultados comparados. Serão agora apresentadas essas mesmas estratégias.

1. ***findContours + Transformada de Hough Circular***

   Mais uma vez é utilizada a função de deteção de contornos apresentada em 6. Os pixeis que constituem os contornos detetados são guardados num vetor de pontos onde cada ponto corresponde à posição de um pixel na matriz da imagem.

Figura 4.6: Transformada de Hough aplicada a um alvo

Após a obtenção dos contornos presentes na imagem é possível utilizar a função *HoughCircles*[26] com o intuito de encontrar os contornos circulares presentes. Caso a função encontre círculos, facilmente se poderá averiguar as variáveis que o identificam como o raio e o centro.

A implementação conjunta das duas funções retornou valores próximos dos que seriam desejados mas não exatos. Como se pode observar na figura 4.6, existe um desvio entre o centro do círculo demarcado pela transformada de *Hough* circular e o verdadeiro centro do alvo. Note-se que o círculo vermelho que pretende detetar a zona negra do alvo está ligeiramente à direita.

2. ***convexHull + boundingRect***

A função *convexHull*[24] é semelhante à *findContours* sendo, no entanto, capaz de detetar defeitos de convexidade presentes numa curva e corrigi-los. O objetivo da estratégia descrita neste ponto é guardar os contornos retornados por *convexHull* e através da função *boundingRect* determinar quais os relevantes para o cálculo do centro do alvo. A função *boundingRect* delimita um contorno através de uma caixa, ou quadrilátero, que o circunscreve, tal como se pode observar na figura 4.7.

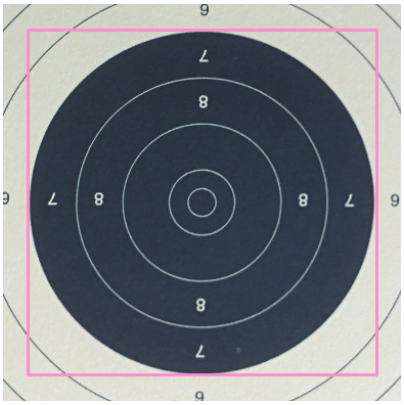

Figura 4.7: Função *boundingRect* aplicada a um alvo

Se a caixa circunscrever o *bullseye* precisamente, os centros destas duas áreas geométricas deverá ser idêntico. Sabendo que função *boundingRect* retorna como parâmetros a posição do vértice superior esquerdo da caixa na matriz da imagem bem como o comprimento e largura do quadrilátero, a localização do seu centro geométrico poderá ser determinada.

$$Centro_x = caixa_x + (caixa_{comprimento}/2) \tag{4.1}$$

$$Centro_y = caixa_y + (caixa_{largura}/2) \tag{4.2}$$

Após o cálculo do centro da caixa, verificou-se que a sua posição, apesar de bastante próxima, não corresponde exatamente à do centro do alvo.

3. ***convexHull + boundingRect + Rotação de Imagem***

   De forma a tentar corrigir o desvio presente na estratégia implementada no ponto anterior, procurou-se realizar a rotação sucessiva da imagem por um fator de 90º e calculando, de cada vez, o centro da caixa. O resultado desta implementação corresponde a 4 pontos distintos. Fazendo uma média da distância entre os 4 pontos esperava-se obter um quinto ponto cuja posição seria idêntica à do centro do alvo.

Figura 4.8: Representação do centro das caixas após rotação

A figura 4.8 pretende apresentar o raciocínio por detrás do cálculo do centro das caixas após rotação da imagem. A vermelho pretende-se ilustrar o centro calculado originalmente, a verde após uma rotação de 90º no sentido anti-horário, a amarelo uma rotação de 180º e a branco uma rotação de 270º. Finalmente após o cálculo média dos 4 pontos descritos obtêm-se o ponto azul, que deverá ser sobreposto ao centro do alvo.

No entanto, após implementação do algoritmo, verificou-se que o centro final determinado não correspondia ao centro do alvo, sendo das estratégias implementadas a que verificou o pior resultado.

4. ***findContours + boundingRect***

   Procurou-se testar o comportamento da função *boundingRect* a partir dos contornos obtidos pela função *findContours*, sendo a ideia por detrás desta estratégia semelhante à implementada anteriormente no ponto 2 desta lista.

Nesta situação, depois da aplicação de *boundingRect* e do cálculo do centro da caixa resultante idêntico ao demonstrado na figura 4.7, verificou-se que este se sobrepunha com bastante precisão ao centro do alvo como se pode observar na figura 4.9.

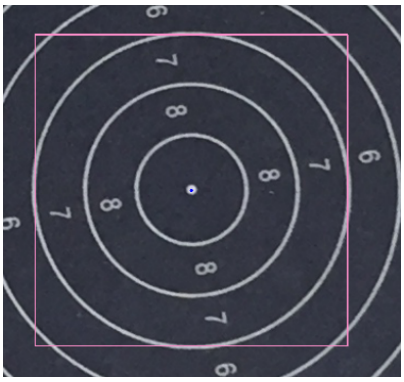

Figura 4.9: Identificação do centro do *bullseye*

O ponto azul assinalado no centro da caixa corresponde ao centro calculado.

Com base nos resultados obtidos pelas diferentes estratégias implementadas, foi escolhida a referida no item 4 por se aquela com os resultados mais promissores.

Encontrando-se o centro do alvo determinado, passa-se agora à determinação do centros dos impactos.

4.3 Determinação da Posição dos Impactos

O início da determinação dos impactos no alvo inicia-se de forma análoga à determinação do centro do alvo na medida em que se aplica novamente uma transformação do espaço de cores *grayscale* para HSV, operações morfológicas e um filtro de mediana de forma a tornar os contornos da imagem nítidos e eliminar irregularidades ou ruído presentes.

A principal diferença reside no facto de por detrás dos impactos presentes no alvo, se ter colocado uma película de cor vermelha de forma a realçar a sua deteção. Devido a esta característica, o intervalo de valores a ser colocado na função *inRange*, que anteriormente(4.5) foi utilizada para isolar a cor preta, deverá ser necessariamente alterado, desta vez de forma a isolar a cor vermelha.

O resultado obtido encontra-se demonstrado na figura 4.10.

Ao ser aplicada a função *findContours* em conjunção com *boundingRect* são determinados os contornos presentes na imagem 4.10. De forma a eliminar possíveis falsas deteções é imposta uma condição de forma a eliminar todos os contornos exceto aqueles cujas as dimensões sejam aproximadamente idênticas ao do calibre da munição utilizada, que para alvos de pistola de ar a 10 metros são de aproximadamente 4.5 mm[27].

O próximo passo corresponde ao cálculo da distância entre o centro do impacto detetado e o centro do alvo, determinado em 4. É utilizada a função matemática *norm* que calcula a distância

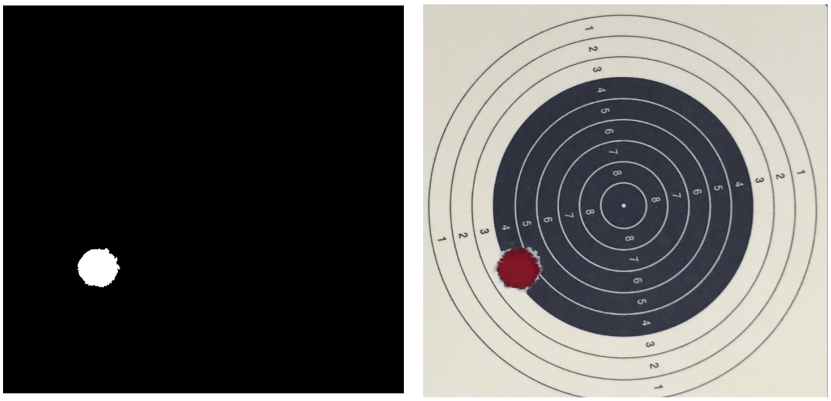

Figura 4.10: Aplicação de HSV para a deteção de vermelho

entre dois pontos na matriz de uma imagem. A essa distância é subtraído o raio do impacto, já que de acordo com os regulamentos da ISSF, o cálculo da pontuação é efetuado de acordo com o ponto da marca deixada pelo disparo mais próximo do centro do alvo.

$$Distancia = (norm(centro_{alvo} - centro_{impacto}) - raio_{impacto}) \tag{4.3}$$

Após a determinação da distância entre este dois pontos, será possível calcular a pontuação do disparo através da implementação de uma função matemática linear.

$$Resultado = 10 - (((2 * Distancia) - 0.5)/5) \tag{4.4}$$

Em baixo encontra-se uma representação gráfica da variação da pontuação em função da distância do disparo.

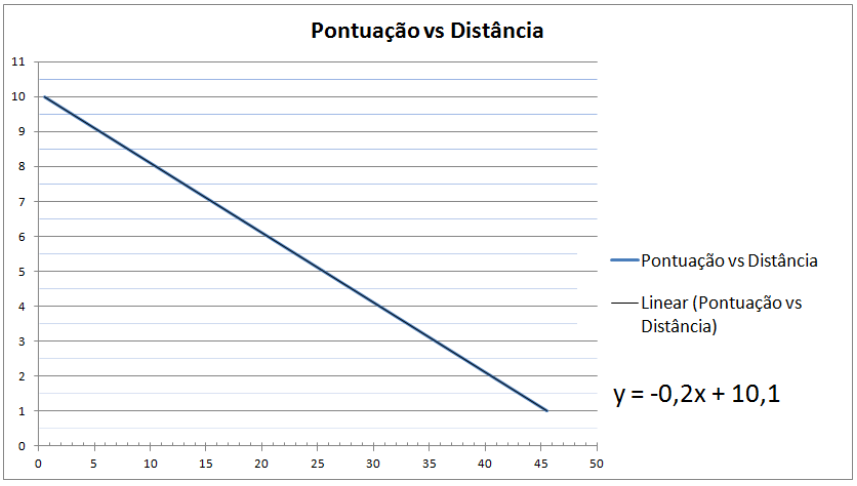

Figura 4.11: Gráfico de variação da pontuação em função da distância

Capítulo 5

Demonstração de Resultados

Neste capitulo pretende-se demonstrar os resultados obtidos pela algoritmia implementada. Faz-se uma interpretação analítica dos mesmos e apresentam-se conclusões tiradas.

5.1 Obtenção dos Resultados

Após a implementação dos diferentes programas, procedeu-se à sua compilação e execução primeiro em computador e numa fase posterior no *Raspberry Pi*. Os resultados são disponibilizados pela linha de comandos das máquinas em questão como poderá ser observado na imagem 5.1.

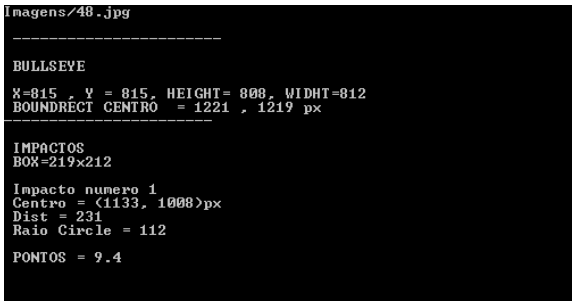

Figura 5.1: Resultados obtidos na janela de comandos

Em conjunto com a linha de comandos é feita a compilação de imagens do alvo em etapas relevantes do programa como é o caso no fim do cálculo do centro do centro do alvo e da posição dos impactos como as figuras 5.2 e 5.3 pretendem ilustrar.

Figura 5.2: Alvo com impacto identificado

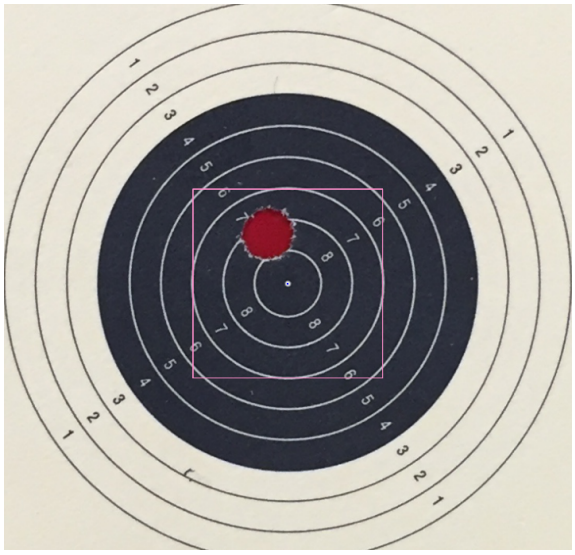

Figura 5.3: Alvo com centro do *bullseye* identificado

5.2 Comparação e Análise - Imagens 8M pixeis

Após a compilação e execução das diferentes versões implementadas, os resultados foram anotados e comparados com os valores esperados, chamados de referência, resultantes da avaliação resultante de uma máquina de pontuação automática.

Em baixo encontram-se as pontuações obtidas pelos diferentes algoritmos que serão comparadas à pontuação de referência. Nesta fase são analisadas as imagens captadas por uma câmara de 8M pixeis de um telemóvel comum.

5.2.1 Correção de Perspetiva - Método 1

Neste método, procura-se determinar os contornos globais do alvo e identificar os quatro cantos do mesmo. Estes quatro pontos serão submetidos às funções *getPerspectiveTransform* e *warpPerspective* de forma a realizar a correção de perspetiva de imagem. Este método traz consigo a desvantagem de necessitar a presença dos quatro cantos do alvo na imagem, facto que nem sempre é garantido.

Como foi referido anteriormente no ponto 6 da subsecção 4.1.1 do capítulo 4, foram testadas duas funções (*approxPolyDP* e *convexHull*) de forma a realizar uma recolha dos contornos, que serão denominadas método 1.1 e 1.2 respetivamente e cujos os resultados poderão ser analisados nas tabelas 5.1 e 5.2.

Comparação de Pontuação Algoritmo vs Referência									
Alvo	1	2	3	4	5	6	7	8	9
Referência	8.9	8.1	5.4	5.2	9.4	10.3	8.8	9.2	10.4
Programa	8.9	8.1	5.3	5.3	9.3	10.2	8.7	9.2	10.4

Tabela 5.1: Pontuação obtida com implementação do método 1.1

Comparação de Pontuação Algoritmo vs Referência									
Alvo	1	2	3	4	5	6	7	8	9
Referência	8.9	8.1	5.4	5.2	9.4	10.3	8.8	9.2	10.4
Programa	8.9	8.1	5.4	5.5	9.3	10.2	8.7	9.2	10.4

Tabela 5.2: Pontuação obtida com implementação do método 1.2

Verifica-se que as pontuações obtidas pelas duas funções são idênticas às de referência em 4 dos 9 alvos testados no caso da tabela 5.1 e 5 dos 9 na tabela 5.2. Verifica-se também que o desvio em relação à referência é geralmente de uma décima exceto para o alvo 4 da segunda tabela em que foi registado um erro de 3 décimas.

5.2.2 Correção de Perspetiva - Método 2

Esta método distingue-se do anterior pelo facto de procurar não os contornos mais exteriores do alvo, mas sim os contornos do centro do alvo. Após a identificação do centro do mesmo, é criado um quadrilátero cujo centro é partilhado com a elipse do centro. As funções utilizadas correspondem à *findContours* em conjunção com *boundingRect*. Finalmente, à semelhança do método descrito em 5.2.1, os quatro vértices do quadrilátero são recebidos pelas funções que irão fazer a correção de perspetiva.

A vantagem deste método (denominado 2) em relação ao primeiro, advém de ser possível corrigir a distorção da câmara sem haver necessidade de detetar a globalidade do alvo.

Alvo	1	2	3	4	5	6	7	8	9
Referência	8.9	8.1	5.4	5.2	9.4	10.3	8.8	9.2	10.4
Programa	8.9	8.0	5.3	5.2	9.3	10.2	8.8	9.4	10.5

Tabela 5.3: Pontuação obtida com implementação do método 2

As pontuações obtidas foram idênticas às da referência em 2 dos 9 alvos testados e o desvio máximo corresponde a uma décima.

Após a análise dos resultados provenientes dos diferentes métodos verificou-se que apesar de ambos os métodos serem viáveis, os métodos 1.1 e 1.2 são mais precisos na estimação da pontuação, obtendo um maior número de pontuações idênticas à referência. O método 2 apesar de menos preciso, traz a vantagem de a etapa de correção de perspetiva não necessitar dos quatro cantos do alvo.

5.3 Comparação e Análise - Imagens 5M pixeis

Nesta secção são analisados os resultados obtidos pela câmara de 5M pixeis compatível com o SBC *Raspberry Pi* e com possibilidade de focagem manual.

O método de correção de perspetiva utilizado corresponde ao método 2 da subsecção anterior (5.2.2). Na tabela 5.4 encontram-se os resultados retirados de 40 amostras.

Comparação das Pontuações Obtidas com a Referência										
Alvo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Programa	9,4	8,7	9,3	8,3	7,5	7,7	7,2	10,2	9,9	8,9
Alvo	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Programa	9,4	9,7	10,1	5,9	9,4	7,7	inv	6,9	9,2	9,8
Alvo	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
Programa	9,4	9,6	9,5	10,1	9,4	6,8	10,1	9,4	9,5	8,6
Alvo	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**
Programa	8,6	8,7	9	8,1	5,6	5,7	9,3	10,2	8,9	9,2

Tabela 5.4

Os resultados poderão ser visualizados graficamente na figura 5.4.

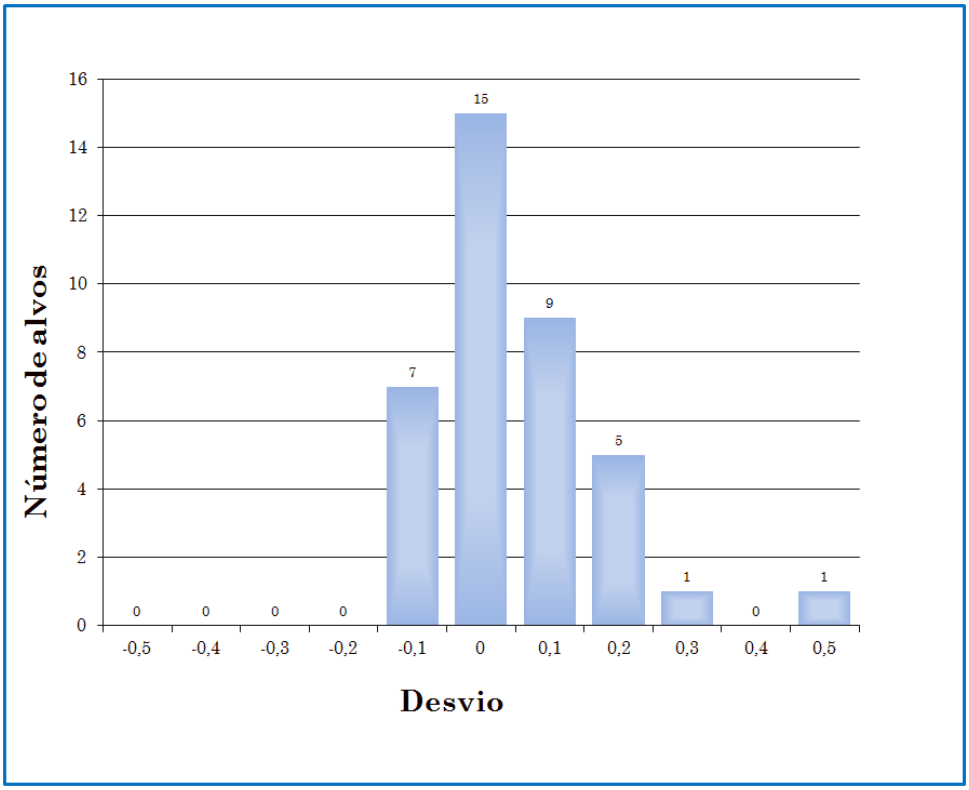

Figura 5.4: Comparação dos resultados obtidos com a referência

É possível constatar que em 15 dos 40 alvos avaliadas, a pontuação atribuída pelo programa foi idêntica à da referência, em 16 ocorreu um desvio de uma décima (quer acima ou abaixo da referência) e nos restantes o desvio corresponde a duas décimas ou mais.

Estes desvios poderão ser explicados devido à instabilidade quer da posição do alvo em relação à câmara quer das condições de luminosidade no momento da captura. Em condições ideais estes fatores deverão ser controlados de forma rigorosa de modo a garantir uma maior consistência por parte do programa de determinação de pontuação.

Capítulo 6

Conclusão e Trabalho Futuro

6.1 Conclusão

Esta dissertação teve como objetivo o desenvolvimento de um sistema de pontuação em tempo real de um alvo de tiro desportivo com o intuito de oferecer a um utilizador uma alternativa viável aos demais sistemas já existentes no mercado. Ao longo deste projeto foi realizado um estudo das opções disponíveis atualmente como sistemas de pontuação manuais, por acústica e por visão computacional e foram analisadas as vantagens e desvantagens de cada um.

Este estudo foi apoiado por uma revisão bibliográfica do conhecimento e tecnologias existentes nas áreas abrangidas por esta dissertação como é o caso do processamento de imagem. Após a análise dos métodos e técnicas mais utilizados foi delineado um plano para a estrutura da programação mais adequada a lidar com o problema em mãos. Devido ao grande número de funções e técnicas de processamento de imagem existentes, esta estrutura foi evoluindo e sofrendo alterações ao longo do projeto. Os resultados de cada alteração foram registados e comparados entre si.

De seguida procedeu-se à migração da algoritmia para o SBC *Raspberry Pi* e mais uma vez foram comparados quer os resultados obtidos quer o desempenho deste componente. Para além disso, foi testada a influência da qualidade das câmaras utilizadas na performance do programa.

Apesar de os resultados terem sido satisfatórios, o sistema poderá ser sujeito a melhorias que serão discutidas na secção seguinte.

6.2 Trabalho Futuro

Como referido, o sistema implementado poderá ser melhorado nalgumas áreas. Em primeiro lugar, a velocidade de compilação e execução do programa ficou abaixo do esperado, o que no caso de aplicações que requerem rapidez representa um problema.

Seria também interessante verificar a influência que uma iluminação por infravermelhos tem na deteção do impacto deixado pelos disparos no alvo.

Bibliografia

[1] X-Ray Exposure: How Safe Are X-Rays? URL: http://www.medicalnewstoday.com/articles/219970.php.

[2] Computed Tomography (CT). URL: https://www.nibib.nih.gov/science-education/science-topics/computed-tomography-ct.

[3] José A. Somolinos, Amable López, Rafael Morales, and Carlos Morón. A new Self-calibrated procedure for impact detection and location on flat surfaces. *Sensors (Switzerland)*, 13(6):7104–7120, 2013. doi:10.3390/s130607104.

[4] Xinnan Fan, Qianqian Cheng, Penghua Ding, and Xuewu Zhang. Design of automatic target-scoring system of shooting game based on computer vision. *Proceedings of the 2009 IEEE International Conference on Automation and Logistics, ICAL 2009*, (August):825–830, 2009. doi:10.1109/ICAL.2009.5262810.

[5] Detecting Circles Using OpenCV. URL: http://www.pyimagesearch.com/2014/07/21/detecting-circles-images-using-opencv-hough-circles/.

[6] Raspberry Pi Camera Module - Geeetech Wiki. URL: http://www.geeetech.com/wiki/index.php/Raspberry{_}Pi{_}Camera{_}Module.

[7] Faizan Ali. Shooting Targets. pages 515–519, 2008.

[8] Internationaler Schiess-sportverband V. International Shooting Sport Federation Official Statutes Rules and, 2013. URL: http://www.issf-sports.org/documents/rules/2013/ISSFRuleBook2013-3rdPrint-ENG.pdf.

[9] RIKA Easy Score 220. URL: http://www.hellotrade.com/rika-sport-gmbh-cokg/target-transport-systems-easy-score-220.html.

[10] DISAG RM-III Universal. URL: http://www.disag.de/produkte/rm-iii-universal/.

[11] ISSF Home Page. URL: http://www.issf-sports.org/.

[12] Ajoy k. Rau Tinku Acharya. *Image Processing: Principles and Applications*. Wiley-Interscience, 2005. URL: https://books.google.com.my/books?id=smBw4-xvfrIC{&}dq=what+is+image+processing{&}source=gbs{_}navlinks{_}s, doi:10.1002/0471745790.

[13] F. Tarsha-Kurdi, T Landes, and P Grussenmeyer. Hough-Transform and Extended Ransac Algorithms for Automatic Detection of 3D Building Roof Planes From Lidar Data. *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007*, XXXVI(1):407–412, 2007.

[14] Matlab Edge Detection. URL: http://www.mathworks.com/discovery/edge-detection.html.

[15] Tony Lindeberg. Discrete derivative approximations with scale-space properties: A basis for low-level feature extraction. *Journal of Mathematical Imaging and Vision*, 3(4):349–376, 1993. doi:10.1007/BF01664794.

[16] Brown Ballard. Computer Vision. chapter 4. 1982.

[17] Bijan G. Mobasseri. Automatic target scoring system using machine vision. *Machine Vision and Applications*, 8(1):20–30, 1995. doi:10.1007/BF01213635.

[18] Itseez. OpenCV Homepage, 2015. URL: http://opencv.org.

[19] Visual Studio. URL: https://www.visualstudio.com/.

[20] OpenCV - Image file reading and writing. URL: http://docs.opencv.org/3.1.0/d4/da8/group{_}{_}imgcodecs.html{#}ga288b8b3da0892bd651fce07b3bbd3a56{&}gsc.tab=0.

[21] Miscellaneous Image Transformations - Thresholding. URL: http://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous{_}transformations.html?highlight=threshold{#}threshold.

[22] Morphological Operations. URL: http://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion{_}dilatation/erosion{_}dilatation.html.

[23] Image Filtering - OpenCV. URL: http://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html{#}voidmedianBlur(InputArraysrc, OutputArraydst, intksize).

[24] Structural analysis and shape descriptors - OpenCV. URL: http://docs.opencv.org/2.4/modules/imgproc/doc/structural{_}analysis{_}and{_}shape{_}descriptors.html?highlight=findcontours{#}findcontours.

[25] Geometric Image Transformations - OpenCV. URL: http://docs.opencv.org/2.4/modules/imgproc/doc/geometric{_}transformations.html{#}getperspectivetransform.

[26] Hough Circle - OpenCV. URL: http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/hough{_}circle/hough{_}circle.html.

[27] ISSF-Olympic Games Rules and Regulations. URL: http://www.issf-sports.org/theissf/championships/olympic{_}games.ashx.