BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

EVALUATION OF TARGET SHOOTING BY COMPUTER VISION

HODNOCENÍ STŘELBY DO TERČE POMOCÍ POČÍTAČOVÉHO VIDĚNÍ

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR **JANA GREGOROVÁ**
AUTOR PRÁCE

SUPERVISOR **prof. Ing. ADAM HEROUT, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2021

Brno University of Technology
Faculty of Information Technology

Department of Computer Graphics and Multimedia (DCGM) Academic year 2020/2021

Bachelor's Thesis Specification

Student: **Gregorová Jana**
Programme: Information Technology
Title: **Evaluation of Target Shooting by Computer Vision**
Category: Image Processing

Assignment:

1. Describe the problematic of sports shooting, focus on aspects relevant for automatic evaluation of shots in the target.
2. Get familiar with the problematic of acquisition and processing of image/video.
3. Collect a suitable dataset of video records of shooting training, cover important situations and scenarios.
4. Experiment with algorithms of image processing that would allow for detection of new shots into the targets and to evaluate the score.
5. Design a system for automatic evaluation of sports shooting by a camera connected to a computer (or smartphone).
6. Implement the proposed system and improve it iteratively based on usage and collecting more data.
7. Evaluate the achieved results and propose possible future work; create a poster and a short video for presenting the project.

Recommended literature:

- Gary Bradski, Adrian Kaehler: Learning OpenCV; Computer Vision with the OpenCV Library, O'Reilly Media, 2008
- Richard Szeliski: Computer Vision: Algorithms and Applications, Springer, 2011
- Steve Krug: Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability, ISBN: 978-0321965516
- Steve Krug: Rocket Surgery Made Easy: The Do-It-Yourself Guide to Finding and Fixing Usability, ISBN: 978-0321657299

Requirements for the first semester:

- items 1.-4., considerable advancements on items 5 and 6.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor: **Herout Adam, prof. Ing., Ph.D.**
Head of Department: Černocký Jan, doc. Dr. Ing.
Beginning of work: November 1, 2020
Submission deadline: May 12, 2021
Approval date: May 11, 2021

Abstract

This thesis involves using computer vision in sports shooting for the purpose of innovating bullseye shooting with firearms specifically. The main goal was to calculate shooting score and display it to the shooter in the course of shooting in order to make shooting more effective, safe, comfortable and appealing to new audience. The result of this work is a complex solution consisting of camera connected to a computer and a user-friendly application, ready for distribution. Score evaluation is achieved by processing camera output using said application. The target is automatically detected in the video, then the image is stabilized using cross-correlation of two 2D arrays and new target hits are detected using background subtraction.

Abstrakt

Cílem této práce je aplikace, která bude v reálném čase automaticky vyhodnocovat skóre střelby na terč ze střelné palné zbraně pomocí kamery připojené k počítači. Střelec tak bude mít možnost sledovat výsledky střelby přímo ze střeleckého stanoviště i během střelby. Výsledná aplikace zpracovává vizuální data z kamery, která sleduje terč. Terč a plochy na tomto terči s různým skóre jsou detekovány automaticky. Obraz z kamery je stabilizován pomocí křížové korelace a nové zásahy jsou detekovány v masce popředí. Výsledkem této práce je aplikace, která zefektivňuje trénink sportovní střelby.

Keywords

Automatic Score Evaluation for Bullseye Shooting, Computer Vision in Sports Shooting, Background Subtraction, Template Matching, Image Stabilization, Cross-correlation of Two 2D Arrays, User Interface

Klíčová slova

automatické vyhodnocení sportovní střelby, počítačové vidění, detekce popředí, stabilizace obrazu, křížová korelace, Fourierova transformace, uživatelská rozhraní

Reference

GREGOROVÁ, Jana. *Evaluation of Target Shooting by Computer Vision.* Brno, 2021. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor prof. Ing. Adam Herout, Ph.D.

Rozšířený abstrakt

Cílem této práce je aplikace, která pomocí kamery připojené k počítači, na němž běží tato apllikace, sleduje papírový terč a během střelby ze střelné palné zbraně na tento terč počítá skóre střelby. Střelec pak během střelby má možnost na obrazovce vidět obraz terče, na němž jsou vyznačeny detekované zásahy a jejich skóre. Po skončení střelby je střelci zobrazeno detailnější vyhodnocení daného kola. Střelec má možnost prohlížet, jak střílel v čase a kontrolovat, kolik bodů bylo přiřazeno jednotlivým zásahům. Tento systém slouží k zefektivnění tréninku sportovní střelby.

Kamera k počítači může být připojená buď přes USB, nebo RTSP. Aplikace sama vyhledá kamery připojené přes USB při startu aplikace. Pro připojení kamery přes RTSP je uživatel vyzván k zadání RTSP adresy. Kamera může být umístěná za palebnou čarou. V tomto případě je potřeba opatřit kameru objektivem, který zajistí dostatečnou kvalitu obrazu terče. Další možnost umístění kamery je těsně před terčem. V tomto případě musí být kamera umístěná níže pod terčem, aby nedošlo k její destrukci během střelby.V tomto případě je terč snímán pod úhlem, což má za příčinu deformaci kružnic na terči v elipsy. Program toto řeší pomocí homografie.

Aplikace na videu automaticky vyhledá terč a ořízne video tak, aby dále zpracovávala co nejmenší část pozadí. Obraz terče je pak během střelby zarovnáván pomocí křížové korelace. Zásahy jsou detekovány v masce popředí. Každý nově detekovaný objekt v masce popředí je otestován, zda má charakteristiky zásahu v terči; je zkoumána velikost objektu, poměr výšky a šířky objektu a poměr obsahu objektu k jeho opsanému obdélníku.

Vyhledávání bodovaných ploch na terči proběhne pouze jednou, a to při prvním nalezení terče na videu. Obraz terče je rozmazán tak, aby zanikly menší detaily terče (například číslice na terči) a dále převeden na binární obraz. V tomto je nalezen černý kruh uprostřed terče. Tento je aproximován pomocí elipsy. Poté program pomocí detekce hran vyhledá nejmenší elipsu terče. Pomocí čtyř bodů z každé z těchto elips je nalezena homografie. Program dále odhadne polohu kružnic terče pomocí dřívr nalezeného černého kruhu. Pomocí homografie pak tyto kružnice transformuje tak, aby odpovídaly perspektivě terče.

Skóre střelby je vypočítáno podle polohy všech pixelů detekovaného objektu, který odpovídá zásahu v terči. Pro každou bodovanou plochu terče, počínaje plochou s nejvyšším počtem bodů, je vytvořena maska této plochy. Pomocí bitového & je tato maska porovnána s maskou zásahu v terči. Pokud mají tyto dvě masky neprázdný průnik, pak je zásahu přiřazeno skóre a detekce pro tento snímek končí. Tímto je zajištěno, že pokud zásah leží na hraně dvou různě bodovaných ploch, pak je mu přiřazeno skóre té plochy, která je hodnocena vyšším počtem bodů.

Součástí této aplikace je uživatelské rozhraní, díky kterému uživatel může omezit maximální počet zásahů v terči pro dané kolo, nastavit časový limit střelby nebo vzdálenost terče.

Toto řešení je určeno k zabudování na střelnici, nebo pro jednotlivce, kteří si s sebou na střelnici přinesou vlastní kameru a počítač.

Toto řešení je určené pro střelce všech dovednostních úrovní. Krom zkušených střelců i uživatelé bez jakýchkoliv zkušeností se střelbou nemají problémy s ovládáním aplikace; systém je tedy vhodný i pro začínající střelce.

Tento systém bych chtěla rozšířit o mobilní aplikaci tak, aby si jednotlivci nemuseli brát na střelnici počítač. Dále bych chtěla uživatelům umožnit ukládání výsledků předchozích kol tak, aby se mohli podívat, jak střílel v čase během dne/týdne/měsíce/roku. Na konci kola by uživatel také měl mít možnost export výsledků do formátu, který bude zjištěn prostřednictvím uživatelského dotazníku.

Systém by také mohl sledovat chování a určité tendence střelce a doporučovat metody pro zlepšení a tréninkové techniky.

V budoucnu plánuji udělat studii, kde budu v čase porovnávat výsledky tréninku střelců samouků, střelců s profesionálním trenérem a střelců, kteří při tréninku používají systém popsaný v této práci. Výsledky pak dokážou, zda a jak moc střelcům může pomoci použití této aplikace při tréninku.

Evaluation of Target Shooting by Computer Vision

Declaration

I hereby declare that this Bachelor's thesis was prepared as an original work by the author under the supervision of prof. Ing. Adam Herout, Ph.D. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Jana Gregorová
May 12, 2021

Acknowledgements

I would like to thank prof. Ing. Adam Herout, Ph.D. for his help and support.

Contents

Chapter 1

Introduction

This work aims to innovate bullseye shooting with computer vision. The focus is on shooting training over use in competitions, however, the solution was designed to be helpful in training for competitions as well. The solution is intended particularly for firearm shooting, but the same approach for target detection and target hit detection will function with air guns as well.

I decided to start this project because I wanted to use a system for automatic score evaluation myself and I thought it might also be useful to other shooters. The reason such system is needed is that in the course of shooting, the shooter is generally not able to see where exactly their shots hit the target (unless using auxiliary equipment such as a telescopic sight), therefore the shooter has little to no idea of their current score. The shooter could use the knowledge of target hit location and its score to re-adjust their position during a shooting round and immediately see the impact on resulting dexterity.

I found general user requirements based on my own experience as a shooter and then validated or amended these requirements based on results of a survey I conducted.

The result of this work is a composite solution which uses a camera pointed at a static target and a computer application to evaluate shooter score and display detected target hits in the course of a shooting round. The camera can be placed either near the shooter and connected to the computer through USB or near the target and connected over WiFi. Target hits are detected and their score evaluated in real time based on said camera output. The program first detects and locates the target in the video using template matching. Individual scoring areas are detected by locating two of the target circles, fitting ellipses on these circles, finding homography using said ellipse's points and warping estimated target circles using the previously found homography transformation matrix. Estimated target circles are found using the area of the black circle in the middle of the target. The target is stabilized in the video using cross-correlation. New target hit detection is carried out using background subtraction on the stabilized video. Each target hit score is evaluated based on the full area of the detected target hit. The user is able to interact with the program through an intuitive user interface. The solution is designed to be fit for individual shooters to use in any place as well as for shooting ranges as a built-in system. The solution significantly improves time-efficiency and safety of shooters' training.

I took part in the Excel@FIT students' conference with this work. The project was awarded by the expert panel.

Chapter 2

Computer Vision in Shooting Sports

2.1 Overview of Shooting Sports

Shooting sports is a very broad area consisting of many different scoring rule systems and a wide variety of disciplines. This diploma thesis focuses on casual training of bullseye shooting [1] in preparation for not only slow precision fire but also rapid fire competitions. In both of these disciplines, the shooter stands still at their firing point and shoots at a paper target which is firmly fastened to one spot. These disciplines differ in time limit. In slow precision fire, the shooter has generally one minute per record shot, which allows for better dexterity. In rapid fire, on the other hand, the time is limited to two seconds per record shot (e.g. 5 shots in 10 seconds), making the practice considerably more challenging.

A shooting target is a flat surface, usually paper or tag board and contains one or more bullseyes (see Fig. 2.3). A bullseye is an aiming point printed on a target. Hits in a bullseye are assigned the highest score, which is typically ten points. A target usually contains other areas with other, lower than bullseyes' score amounts. A different amount of points is assigned to the shooter depending on the shot hole location in the target. If a shot hole area either comes in contact with the outside of a scoring area or it crosses two areas, it is given the higher value. It is possible that the shooter hits the same area multiple times. According to NRA Precision Pistol Rules [1], only visible hits are scored (see Fig. 2.1). However, in the case of grouping three or more shots together so close that it is possible for another shot to go through the enlarged hole without leaving a mark, the shooter will be scored hits for the non visible hits as if they passed through this area. In such cases where this area crosses multiple scoring areas, these shots are assigned score of the highest traversed scoring area. All hits outside scored areas are scored as misses with zero points. The shooter's goal is to score as many points as possible.

During casual training, the distance between the target and the shooter can vary depending on shooter's skill, firearm used or preference. However, it is generally not shorter than 10 m. That way, the shooter is unable to see where exactly their hits land unless using a telescope (see Fig. 2.2). In slow precision shooting, it is important especially for beginners (due to their generally irregular movement patterns and great variability in errors [2]) to see the place in the target they hit right after firing; this way, they could instantly see the impact their body position and breathwork has on their shooting dexterity and could react to it right in the next shot by adjusting body position, breathwork and concentration levels.

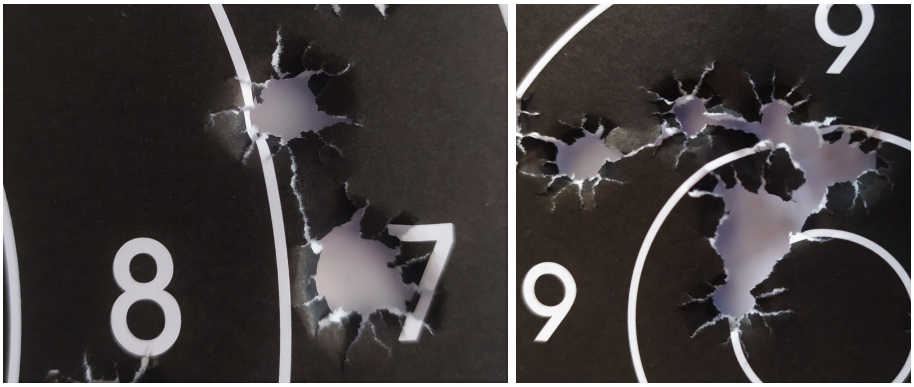

Figure 2.1: **left:** A close (doubtful) hit situation. The upper left shot would be scored as a cut 8. **right:** Grouped shots. It is impossible to tell how many shots hit the bullseye, provided that no other information is given.

Figure 2.2: In the course of shooting, the shooter is hardly aware of their actual score and towards which directions they tend to pull the gun to. Provided that the shooter had instant feedback, they could react to it immediately compared to after finishing a shooting round. This potentially makes the training procedure much more effective.

In the course of one shooting round, varying number of shots in varying time limit can be fired, but exact limits have to be determined without exception before the shooting round starts. The score is evaluated only after the shooting round is over. The person who calculates the score (during competitions, a non-competitor is selected for this position according to the hosting organization's rules; during training, it is usually a coach or the shooter himself who calculates score) has to take the necessary safety measures (all shooting in the area has to be stopped) as they walk into the shooting field to take a closer look at the target card they're evaluating. The score for each shot is evaluated manually (a scoring

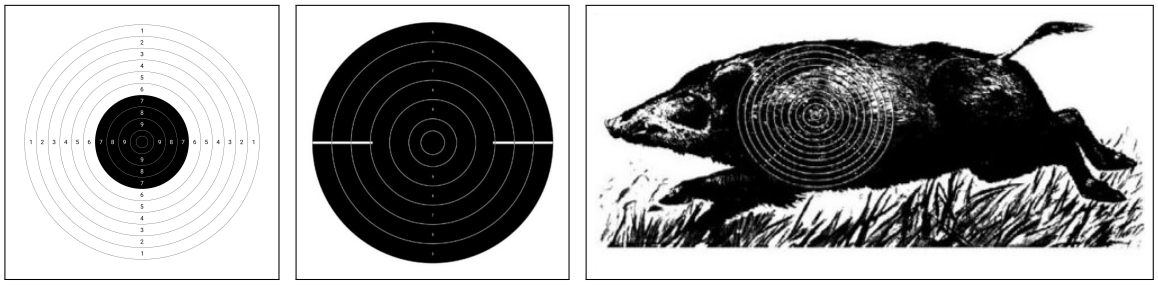

Figure 2.3: Different paper target types (images originated from ISSF General Technical Rules [4]). Left to right: 25 m Precision and 50 m Pistol Target, 25 m Rapid Fire Pistol Target, 50 m Running Target for Paper Targets.

gauge can be used for doubtful hits [3]) and marked on the target. During competitions, these results are recorded in designated score cards as well.

When the round is over, in competitions, the fired targets are exchanged for new ones and the statistical office retains them until the expiration of the time allowed for challenges and protests. When training, the shooter is free to keep or discard the target, or to cover their target hits with cover up patches in order to reuse the same target.

2.2 Existing Solutions

There are several solutions dedicated to shooting score evaluation or shooting training aid available on the market. I analyzed these and grouped them depending on the technology they use to operate:

1. Mobile Phone Solutions,
2. Computer Applications,
3. Solutions With the Use of Designated Scanners,
4. Electronic Scoring Targets With Acoustic Location of New Target Hits,
5. Electronic Scoring Targets With Light Triangulation,
6. Electronic Scoring Targets Using Piezoelectric Sensors to Locate New Target Hits,
7. Open Source Solutions,
8. Other Solutions.

Mobile Phone Applications

Mobile phone solutions are the most affordable and portable way to calculate and store shooting results. Existing solutions use computer vision to evaluate shooting score based on a photograph of a target after a shooting round is finished. It is necessary for the shooter to take the photographs of the shooting target up close after the shooting round is finished. The shooter is not able to see their score in the course of shooting.

TargetScan ISSF Pistol and Rifle [5, 6] is an application for Android and iOS operating systems (see Fig. 2.4). It evaluates score using shooting target photographs.

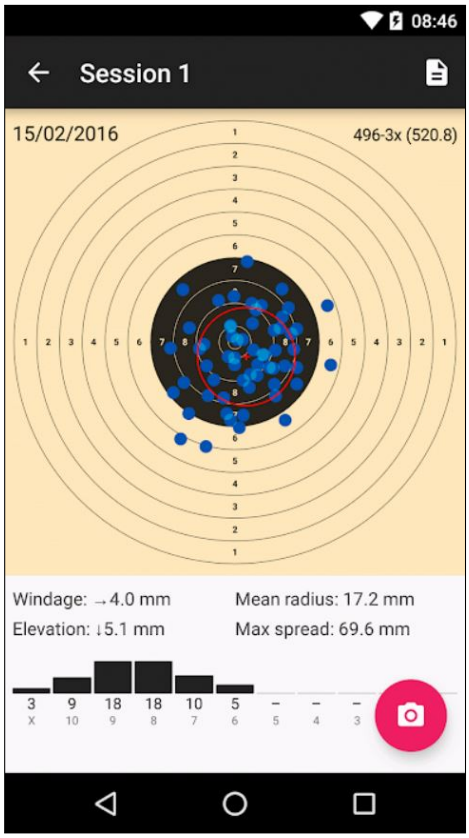

Figure 2.4: An example of an existing solution which is available for mobile phones: TargetScan ISSF Pistol & rifle. Image originated from the application's store page [5].

Many different shooting targets are supported, the user is able to pick the one they're using themselves. The user also picks ammunition they use. The user can adjust the result, add undetected hits, remove false positives and move detected hits to a more accurate position. It is not possible to see the score while shooting. The application also saves results of previous shooting sessions, calculates Windage, Elevation, Mean Radius and Extreme Spread and shows heat map. Is able to calculate stats for each round separately and group all rounds to a session and calculate stats for the whole session. Allows to export session into .pdf and to e-mail it. Costs $19.99 (549.99 CZK), $24.99 on App Store. This application offers optional in-app purchases to cover maintenance costs. The application is rated 4.4 out of 5 on google play with 705 user reviews, 4.8 out of 5 on app store with 25 user reviews. A study [7] concluded that this application is valid and reliable enough for measuring amateur and semi-professional shooting performance. However, it has not been certified by ISSF up to this point.

Target Scanner for Competition Shooters [8]. This application works on a similar principle as the previous solution and offers similar functionality, but has worse user ratings (average of 2.6 out of 5 with 13 user reviews). The cost is $8.49 (239.99 CZK).

Computer Applications

There are currently no applications which process camera streams in real time. This section covers standalone applications only. Supporting computer applications for electronic scoring targets and target scanners are dealt with in the following sections.

E-sniper[1] detects target hits based on before and after images from a distance, which have to be imported into a computer. Framing points must be printed on target. The user is able to review the target after it is automatically evaluated and add undetected hits. Results are stored in gallery. Final score is not calculated, the program only locates target hits. The solution supports air guns only. It is currently not possible to acquire this application.

Target Scanners

Target scanners operate by scanning used shooting targets and evaluating score using a computer or other processing device. These solutions serve as a post-match analysis-shooting score is evaluated only after a shooting round is finished. Therefore, the shooter is not able to see their target up close in the course of shooting. Due to the generally large size of shooting targets for firearms (60×60 cm as opposed to 15×15 cm which is typical for air guns), these solutions are designed to work exclusively with air guns.

Orion scoring system [9] consists of a computer application, scanner and designated targets[10]. The price starts at \$290 + \$70 for an annual license fee for individuals. Orion for clubs ranges \$1110 to \$1680 with annual license at \$120; offers customizable bundles. The solution supports virtual matches. Online results for registered Orion users (with no registration fees). Scores are posted on the internet automatically.

DISAG RM-IV [11] (see Fig. 2.5) is a designated evaluation device. It can operate without a computer but has multiple SW options to work with. The user is able pick between various competition modes. The scanner costs 2 869 EUR; software components range from 88 to 296 EUR. DISAG RM-IV passed ISSF Certification Tests for Electronic Scoring Targets phase 1 (test for accuracy, specification and build standard test only) [12] and can therefore be used to evaluate ISSF shooting competitions.

Electronic Scoring Targets With Acoustic Triangulation

Electronic scoring targets with acoustic location of new target hits are distinctive targets with 3 or 4 built-in acoustic scanners. The location of each target hit is calculated by processing the data of these scanners in a computer. While shooting, the shooter is able to see their resulting score as well as the location of their target hits during shooting on a PC or a specific shooting monitor (see Fig. 2.6). Some of these electronic shooting targets (e.g. Megalink 4K187 [14], SIUS S310 [15]) have been certified by ISSF [12] for use in ISSF shooting competitions. All of these solutions work with air guns and are not suitable for firearms due to the large size or firearm shooting targets and due to firearm destructiveness. The price is approximately between \$760 and \$6, 000.

Electronic Scoring Targets With Light Triangulation

This type of electronic scoring target uses either lasers or a high-speed infrared camera to calculate shooter's score. The results are displayed on a monitor in the course of a shooting

[1] Demonstration video can be found at https://www.youtube.com/watch?v=NCOP9q88HG8.

Figure 2.5: DISAG RM-IV designated target scanner. Image originated from user manual [13].

Figure 2.6: PC-based system for personal training with the 4K187 electronic scoring target (on the left of the image) by Megalink. Image originated from the official Megalink website [16].

round. These solutions are designed for air guns only. The cost of most of these systems is not provided publicly.

The **DISAG OpticsScore** [17] system (see Fig. 2.7) uses a "measuring frame" with a high-speed infrared camera to evaluate score based on hit coordinates. The data is then processed in a so-called "Gate". The results are sent to a "Shooting information center" ("SIZ"), which shows the output for a shooter on a monitor (the SIZ can also be installed on a PC). The system offers visualization SW for match spectators and is intended to use to organize competitions as well as for home use. Score statistics can be exported to printable formats. The system is certified by ISSF [12] for use in shooting competitions.

The **Kongsberg OpticsScore Electronic Target System** [18] is intended to be used with air rifles up to .22 caliber. Possible target distance ranges from 10 m up to 2000 m. The transmission between target line and firing line is implemented by cables. Kongsberg also offers at home options of the system. A 3-lane 10m OpticScore electronic target system [19] (cheapest option) starts at $16 600. The Kongsberg company also offers electronic shooting targets with acoustic triangulation.

Figure 2.7: The components of Disag OpticsScore Electronic Shooting target with light triangulation. Left to right: Measuring frame[i], Gate[ii] (on the left of the picture), SIZ[iii], TouchScore[iv] (an interactive monitor).

[i] Source: https://www.disag.de/produkte/opticscore/messrahmen
[ii] Source: https://www.disag.de/produkte/opticscore/gate
[iii] Source: https://www.disag.de/produkte/opticscore/schutzeninformationszentrum
[iv] Source: https://www.disag.de/produkte/opticscore/touchscore

SIUS LS25/50 [20] uses lasers to detect target hits. The system is ISSF approved for use in shooting competitions [12]. The LS25/50 supports hunting targets as well. SIUS also offers shot sensors – these sensors measure shots fired as well as target hits. Most systems are able to detect target hits only.

Electronic Scoring Targets Using Piezoelectric Sensors to locate New Target Hits

Electronic scoring targets using piezoelectric triangulation by Sport Quantum use two plates and four sensors to find position of the impact [21]. The impact plate has to be changed after some time because it gets slightly damaged with each impact. The image on the target can be changed interactively using an Android application. In this application, the shooter is able to see their score, target hits, and also the location of their average hit. The solution for firearms is in development and not available as of today. According to Sport Quantum's website, it is going to function with firearms using any caliber and rifles up to 1 000 joules and for distances from 25 m to 50 m [22]. The solution for air guns is available for roughly $2 600 plus taxes [23].

Open Source Solutions

Open source solutions are attempts to create affordable versions of Electronic Scoring Target. These vary in technology used.

The **FreETarget** [24] uses acoustic triangulation to locate target hits. Electronic Target [25] uses impact shockwave triangulation to locate target hits and then displays these hits in a user application. There is also a solution[2] using a web camera and processing the image using emguCV library. A graphical representation of the results is then displayed in a user application. The shooter is then able to change the target from a distance by making the target device move the paper. All of these open source solutions are for air guns only.

Other Solutions

I have researched some other solutions which do not calculate shooter's score directly, but offer some of the functionality that I would like my solution offered as well. **PitShoot-**

[2] BidaSius – https://github.com/BlasterBB/BidaSius

ing [26] is a solution which shows target during shooting using a camera attached above target stand and provides a solution for walking into the shooting field – targets are attached to a construction which moves the target to the exact distance the user sets it to move to using a provided tablet. If the shooter wants to see the target up close in order to evaluate their score, or to change the target, the system can also be moved on the firing line. In the course of a shooting round, the shooter is able to see the target on a shooting monitor which is atached right next to the shooter. However, the system does not calculate shooting score and the camera image is often delayed or malfunctioning. The solution is built-in in a shooting range. Other related solutions are shooting match managers (e.g. **ABVisie Match Manager** [27] and **Rifle Target: Rifle Shooting Database** [28]). Shooting match managers process data input by a user to help organize matches and to register shooting results. There are also many other solutions for mobile phone and computers which help the individual shooter to keep their results and (manually) compare these over time.

Conclusions

The systems mentioned above can either use data or cable transmission, depending on the use (cables are generally used in permanent installations). Most of these systems currently only work with air guns. At the moment, there are no solutions which offer automatic score calculation in the course of shooting for firearms. I ran a survey (for exact questions and answers see Section 4.3) in which 18 subjects have participated. None of said subjects currently use a system for automatic shooting score evaluation, either because it is unavailable or they have not heard about any such system at all.

Chapter 3

Proposed Algorithms for Target Hit Detection

Even before designing a whole solution, I wanted to make the core functionality of the application, which is score calculation. I chose not to use machine learning in the solution due to wide variety of light conditions at shooting ranges and the target possibly being viewed from varying angles; the dataset would have to be immense in size. I also wanted the application to use as little process time as possible, so that it would not be complicated to make the application available for mobile phones as well, should the demand arise.

I proposed some algorithms which I then tested off-line on previously attained videos of targets being shot at (see Section 5.1) and evaluated their results. All of the following algorithm were implemented solely for the 25 m Precision and 50 m Pistol Target (see leftmost image in Fig. 2.3).

3.1 Proposed Algorithms for Target Hit Detection

Background Subtraction

Background subtraction is a method which subtracts an image of a background from an actual image with some foreground to find the foreground. This can be used to detect target hits. In theory, the target is static and can be used as the background. New target hits are going to appear in the foreground. That way, new shapes could be detected from the foreground mask and then compared to their location in the original target to find the score.

I made tests using both Gaussian mixture model [29] (implemented as MOG2 Background Subtractor in OpenCV [30]) and non-parametric model [31]. The non-parametric model showed better results for target hit identification (see Fig. 3.1 and last two lines of Table 3.1), therefore it is used in the final program (implemented as K-nearest neighbours background subtraction algorithm in OpenCV). To find optimal parameters bearing the most accurate results, I tested the KNN Background Subtractor on the full dataset with varying parameters. Detecting the shadows is undesired.

However, background subtraction in and of itself proved insufficient. At this point, the program still detects noise in the background as the bullets are stopped in the backstop of the range.

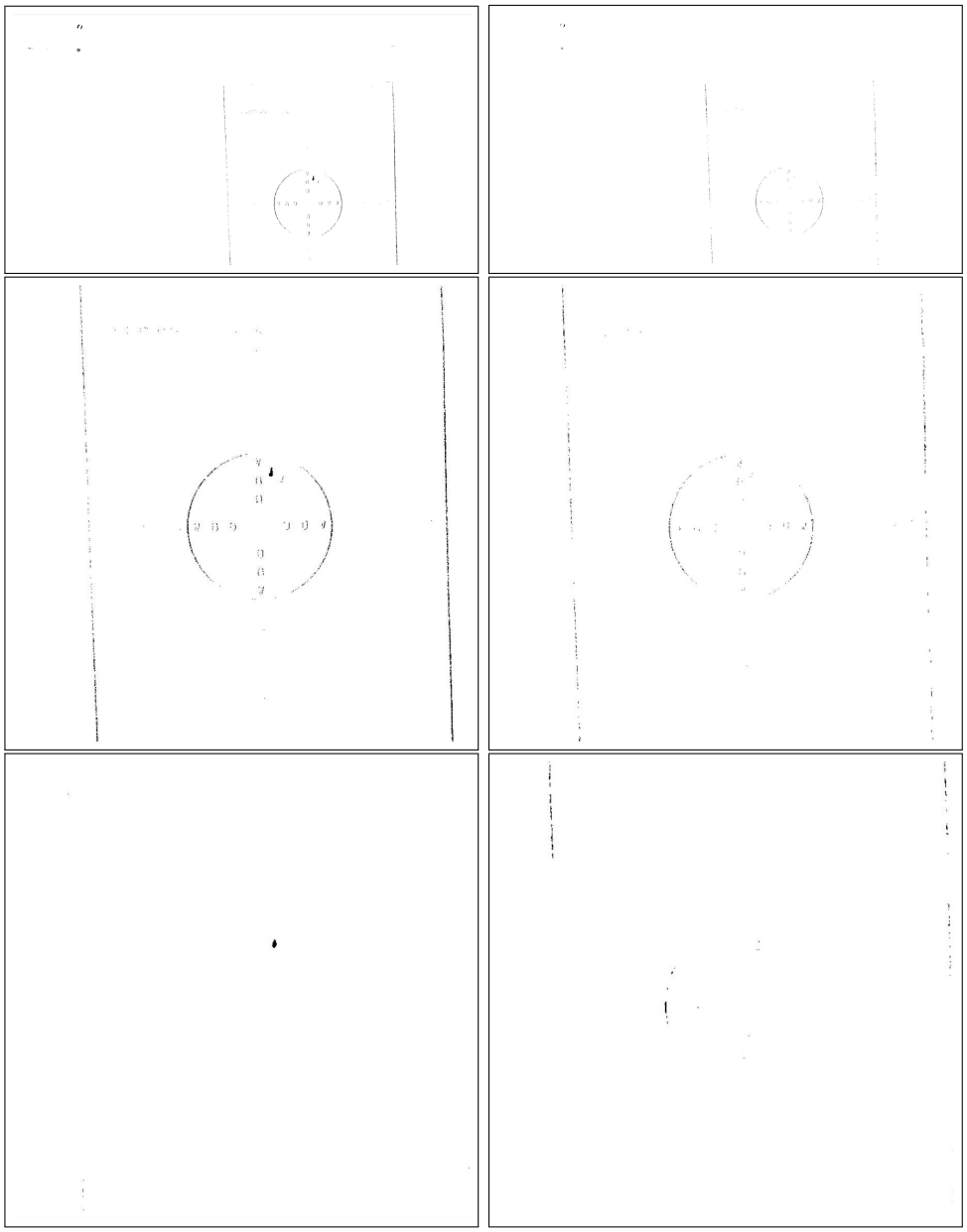

Figure 3.1: Foreground detection(left to right, top to bottom; each image is a side-to-side comparison of KNN(K-Nearest Neighbours Background Subtraction Algorithm) and MOG2(Gaussian Mixture-based Background/Foreground Segmentation Algorithm)): without any image pre-processing, after cropping the image, after implementing image stabilization using cross-correlation (see Section 3.1).

Feature-Based Image Alignment

Image Alignment [32] is a method which takes a reference image (template) and another image, which is searched for the template, as an input. The algorithm finds features in the reference image and their corresponding points in the other image. It then aligns the image by warping and cropping the image, so that it matches the reference image.

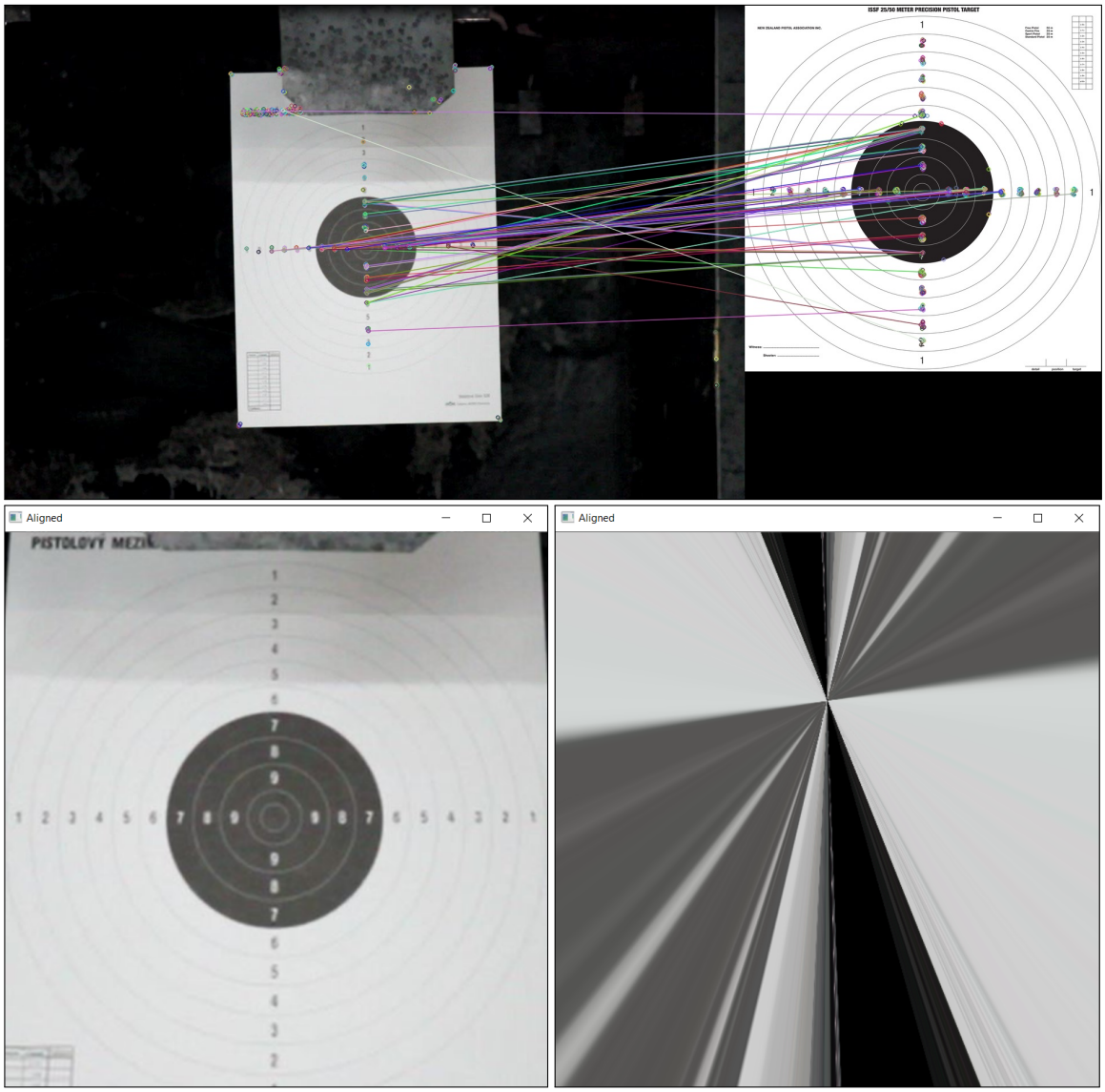

Figure 3.2: Feature based image alignment. **Top:** image alignment – feature matches. **Bottom, left to right:** image alignment in a single picture, failed image alignment in a video.

If the program located the target in each frame using a reference image of a target and aligned it, the image would be cropped so that only the target would be visible. Moreover, the target would be straightened in the image and not moving at all. This works for a static image, used in a video it takes a very long time and the result gets distorted (see Fig. 3.2). Therefore, I decided not to implement this method in the resulting application.

Image Alignment using Cross-correlation

Another approach to image alignment is by computing the cross-correlation coefficient between the first processed frame of the video and the currently processed frame. This method finds the shift between said frames (see Fig. 3.3) and aligns these frames. OpenCV does not offer a ready-to-use cross correlating function, so I implemented my own one. The

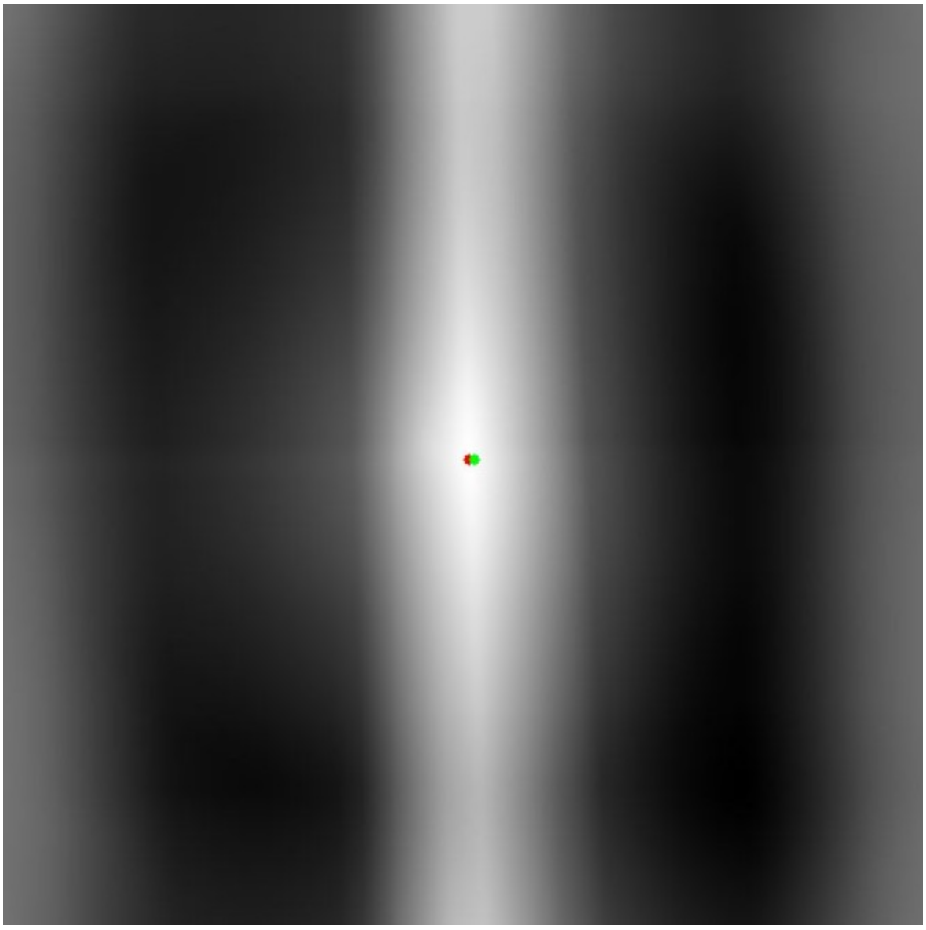

Figure 3.3: Result of applying cross-correlation on two 2D arrays (images) to find the exact position the second image shifted to. Here the positions of the red and the green dot indicate the direction and the distance in which the target moved compared to the first frame processed.

function flips one of the two images horizontally and vertically, converts both images into grayscale, blurs them using Gaussian Blur[1], subtracts mean values from each of said images and calculates a DFT[2]-based convolution of these images. The result is then normalized and searched for its brightest spot. The difference between this spot and the brightest spot found when cross-correlating the reference frame equals to the distance in pixels which the target traversed in the frame. Using the exact knowledge of the shift, the current frame is then aligned with the first frame. Foreground detection is applied on the aligned frame.

The method does not take into account a change of perspective; still, it makes the hit detection sufficiently exact and the processing time is significantly reduced compared to the previous method. The best results are obtained when the target is firmly attached both at its top and bottom. Image stabilization by shifting an image helps if the whole target shakes, but in the case when the target is only attached at its top, the target detection could possibly be less exact. The target should also be prevented from rotating.

[1] Gaussian blur in the OpenCV library
[2] Discrete Fourier Transform of two 2D arrays in OpenCV

In order to make the foreground subtraction more robust and prone to sudden movement (which does happen in spite of the image being aligned), every

$$\mathcal{D} = \frac{\mathcal{H}}{2} + 50, \tag{3.1}$$

where $\mathcal{H}$ is the background subtractor's history (and equals to 800 in this case) of frames; after image alignment, the frame will be purposefully moved by one pixel in both x and y coordinates. The immediately following frame will be moved in a similar way, but by minus one pixel. This could theoretically cause hits not to be detected if they land on numbers or edges of score areas, since the background subtractor is used to movement in these areas and does not detect it any more unless the movement is great. Either way, that was never an issue on my dataset and the background subtractor always detected hits in these areas without any issues.

Template Matching

Another method I have used in order to improve target hit detection accuracy is Template Matching[3]. Every object detected as pertaining to the foreground is compared to a reference image of a target hit which had been resized to match the size of the detected object. The template is only searched in the detected object bounding rectangle, because when each frame was fully searched through for the reference image, it took too much processing time to search through the whole picture and it wouldn't be possible to use the application in real time that way. The results have also been more satisfactory for when the template matching was used on detected objects only. In any way, this approach did not bear acceptable results. The target hits in the black circle were hardly detected, since they do not look very prominent in the original image (see Fig. 3.4). I obtained better results for hits on a white background, but the numbers inscribed on the target look too similar to target hits and can move with the target, so they got detected as false positives very often. I also attempted to add images of the numbers as reference images. If the area with detected movement was similar enough to a template containing a number, the detected area would not be marked as a target hit. This did not function very well either, the application took even more processing time and the numbers on the target still got detected as target hits and even more hits got undetected. In the end, I decided to not use any template matching for target hit detection at all.

Restricting Hit Area Size

Each target hit occupies a certain area of relatively similar size. I estimated that a hit could take up roughly between $\frac{1}{3000}$ and $\frac{1}{14000}$ of the total frame area after it is cropped to fit the target. Discarding all objects that won't fit into this gap successfully prevents small noise and target borders from being detected as target hits (see Fig. 3.5).

Filtering out objects which change quickly over time

The program detects a relatively great mount of noise. Most of the noise is created by slight movement of the target and in the background. This causes the program to detect transient objects in the foreground mask. Target hits, on the other hand, stay relatively

[3] Image Processing: Object Detection

Figure 3.4: Result of template matching. Each detected object in the foreground is first compared to reference images of target hits (in this case the two images on the right were used). If the object is similar enough, it is marked as a hit.

similar over time. On that account, I implemented a functionality which assures that the detected object is not marked as a target hit, unless a similar object appeared in a previous frame. The area with the detected object is cropped out of the foreground mask using a bounding rectangle and compared with the same area in the foreground mask of the previous frame pixel by pixel. If the difference is greater than a threshold of 10 which I have found experimentally, then the object is not marked as a target hit. This approach prevents objects that either appear in the video for a very short time or move very fast (such as flying bullet cases) from being detected as target hits. The disadvantage of this approach is that the hit is detected with a one frame delay. If the exact time of the hit needs to be recorded and shown to the user in the statistics, it could be done by recording the time of the past frame with each hit. However, the shooter will always see the hit being detected with one frame delay during shooting. In the end, testing proved that the advantages outweigh the disadvantages and much less false positives are being detected.

Cropping the Background Out of the Video

Since the background is not important, it can be cropped out of the video for further false positive reduction and processing time reduction. The program uses template matching[4] to locate the target in the frame. The `TM_CCOEFF_NORMED`[5] operation is used, because there is a relatively great difference between the template and the source image (e.g. different illumination). A reference image of a target is used as the template. In order to make template matching scale invariant, the program loops over the function multiple times and progressively makes the template image smaller. Since the template image is bigger than all the actual targets in the dataset, there is no need to execute previously mentioned loop with the template image scaled bigger than its original. To identify a successful match, I

[4] Image Processing: Object Detection
[5] Normed Correlation Coefficient template matching operation

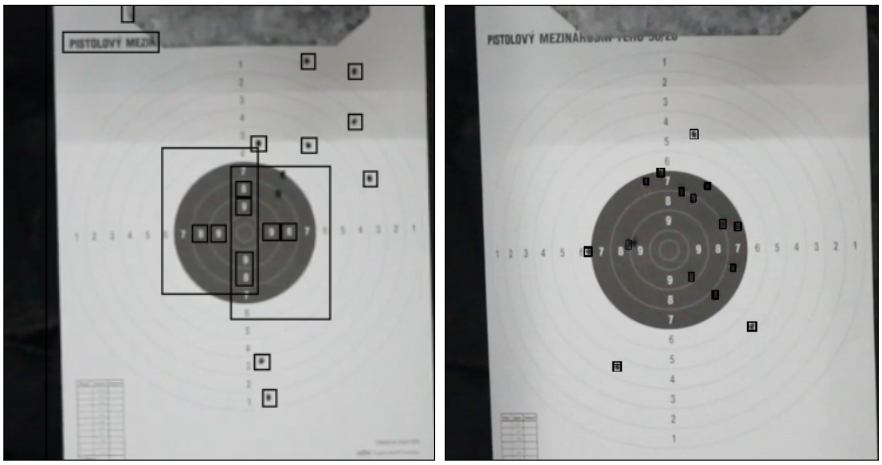

Figure 3.5: Left to right: Target hit detection without hit size restraints, target hit detection with hit size restraints and image stabilization.

experimentally found a threshold value. Finding the closest match possible is not useful in this case, as it is possible that the target is not in the camera frame at all. Once the match value is greater than said threshold value, the program saves the rectangle with the supposed target location in the source image and then uses this rectangle to crop all frames the program later works with. That way, the target template matching loop is executed only once.

Intersection Over Union

Due to the program processing each frame's foreground mask in the same way, the program detects each target hit several times (see Fig. 3.5). I implemented the Intersection over Union [33] evaluation metric to prevent the program from detecting the same hit more than once. If the ratio between the area of overlap of detected areas and the area of union of said areas is lesser than a threshold value I found through a series of experiments, then the detected object is evaluated as a new hit. Said threshold value is relatively small, therefore the program often detects grouped hits as only one target hit. This should not pose a problem for shooters of beginner through intermediate levels, as shooters of these levels are generally not able to intentionally hit the exact same spot multiple times. Expert shooters could be significantly disadvantaged, since most of their shots land right in the bulls eye. This could be solved at a later time through different evaluation metrics for expert shooters compared to beginners and intermediates (e.g. the hits could be inscribed in a circle and the score would be evaluated based on its total area, the distance of the center of the circle from the target center and number of shots).

Restricting Side Ratio of Detected Objects

Restricting the detected object area proved insufficient, as some of the target edges were still being detected. This was due to the object's bounding rectangle side ratio not being restricted in any way. A hit's bounding rectangle's side ratio should be close to one and not lesser than $\frac{1}{2}$.

Figure 3.6: Finished target hit detection.

Restricting the Area Which a Target Hit Can Occupy in Its Bounding Rectangle

The program sometimes detects parts of scoring circles. Their bounding rectangle is similar enough to a square and is of the right size, but from the object's contour in the foreground mask, it is apparent that it is not a target hit. The object is a thin, curved line, whereas a target hit resembles a filled-in circle in the foreground mask. On that account, I set restrictions to the amount of pixels the object has to occupy in its bounding rectangle. I experimentally evaluated this area as half of the area of the object's bounding rectangle.

With this restriction, target hit detection becomes sufficiently exact (see Table 3.1 and Figure 3.6). The program detects 100 % of all target hits displayed in the video.

3.2 Proposed Algorithms for Detection of Scoring Areas

Prior to assessing respective score to a newly detected hits, the location of areas with different score values has to be determined. As stated in Section 2.1, targets can vary. This has to be executed only once for the whole shooting round at the time the first frame is processed, as the target stays in one place and the score areas do not move significantly. The program will locate the edge of the area scored with seven points (edge of the black area in the middle of the target) and estimates scoring areas as circles. After that, an ellipse is fitted on the located black area in the middle of the circle. Then, using contour hierarchy and flood fill, the smallest circle on the target is located and an ellipse is fit on the area. The program finds homography[6] using points of said ellipses and warps estimated scoring areas using the transformation matrix.

[6] `OpenCV:` Camera Calibration and 3D Reconstruction: `findHomography()`

Table 3.1: Comparison of Methods Used for Target Hit Detection. Each method is a combination of the listed method and all other methods above the method. Table data were retrieved by testing listed methods on the same sample video. Actual target hit count is the human-read amount of target hits in the input video. It deviates for some methods because in some cases the hits were cropped out of the image by the program itself.

Method Used	Actual Target Hit Count	Detected Hit Count	False Positives [%]	Undetected Target Hits [%]
Background Subtractor KNN (max. 1 hit per frame)	18	1,879	99.0	0
Restricted Hit Area	18	1,611	98.9	0
Template Matching (not used further on)	18	23	78.3	66.7
Detected Object Must be Present in the Same Area of the Previous Frame	18	1,261	98.6	0
Locating the Target in the Frame and Cropping its Background Out	15	1,099	98.6	0
Image Stabilization	15	868	98.3	0
Intersection over Union	15	37	59.5	0
Detected Object Side Ratio Restrictions (x:y > 0.5)	17	18	5.6	0
Amount of Detected Pixels in Object Area > $\frac{1}{2}$ of Total Object Area	17	17	0	0
Using Gaussian Mixture Model in Place of the Parametric Model for Background Subtraction (Parameters Set to Yield Best Results)	15	30	53.3	3.3

Locating the Edge of the Area Scored As Seven Points

The frame is first cropped by $\frac{1}{6}$ on each side to assure that the background does not interfere with the threshold and the black circle stays visible in the binary image (see Fig. 3.7). Subsequently, the frame is blurred using Gaussian Blur and converted to binary image by thresholding[7]. The kernel for the Gaussian blur has to be quite large (I experimentally found that the value should be between 35 and 43), so that the details in the target are lost. It is important that only the prominent black circle in the middle is visible after applying threshold. The threshold is found using Otsu's method [34]. The program then uses contour approximation[8] to locate the prominent circle in the image.

[7] Implemented as `cv::threshold()` in OpenCV
[8] Implemented as the `cv::ApproxPollyDP()` function in the OpenCV library

Figure 3.7: **Left:** Result of applying Otsu's threshold [34] on an image which was cropped after the target had been automatically detected. The program crops the image so that the whole target is visible. In this case it means leaving part of the background in the image. When determining the threshold, the whole image is taken into account and the background interferes with the threshold value making the black circle in the middle of the target disappear in the binary image. For score area location, only the black circle in the middle of the target is necessary. By cropping the image further and making sure that the background won't be taken into account when determining the threshold, the case in the picture can be prevented. **Middle:** Result of applying manually found threshold on the same image. **Right:** Result of applying Otsu's threshold on the same original image after cropping out $\frac{1}{6}$ on each side.

Estimating Scoring Areas As Circles

The located circle's radius is then divided by four to find the radius of the bull's eye. Each subsequent circle is then found by adding the bullseye's radius to the previous circle's radius. These areas are sufficiently exact when the target is viewed up straight by the camera (see Fig. 3.8). The more angled the camera view is, the less exact the scoring area detection is.

Approximating Scoring Areas As Ellipses

Should the camera be placed near the target such as in Fig. 5.1 bottom, detected score areas are not exact enough for a satisfactory score evaluation (see Fig. 3.10 middle – this would have been the final detected score areas as described in the previous paragraph). Instead of a circle, an ellipse is fitted on the located contour using OpenCV function `fitEllipse()`[9]. Another ellipse is found by applying Canny's edge detection[10], contour hierarchy and flood fill on the image of a target (see Fig. 3.9). Four points of each ellipse (spaced by 90 degrees) are used to find homography matrix[11]. This matrix is subsequently used to warp the score circles which are drawn as described in the previous paragraph. The result improves the score area location only slightly for when the camera is viewing the target up straight, but the improvement is significant when the target is viewed at an angle.

[9] `OpenCV:` Structural Analysis and Shape Descriptors: `fitEllipse()`.
[10] `OpenCV:` Feature Detection: `Canny()`
[11] `OpenCV:` Basic concepts of the homography explained with code

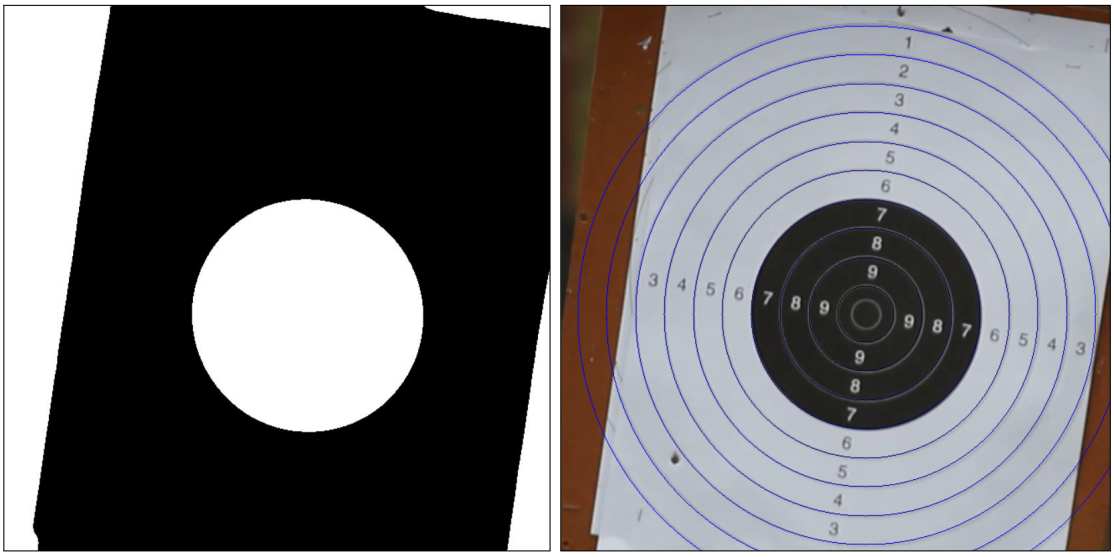

Figure 3.8: Locating score areas: binary image of the target which is used to find different score areas (left), estimated score area borders as circles (right).

3.3 Proposed Algorithms for Score Calculation

The score is evaluated based on the exact area of the hole in the target created by the firing bullet. The whole area of the hit is visible in the foreground mask after impact. That way, the hit score can be evaluated by applying bitwise & on the foreground mask of the hit and a scoring area.

Right after the particular hit is assessed as a new hit, its area is cropped out of the foreground mask and inserted into an empty mask image to isolate the hit from the foreground mask of the whole target which is likely to contain multiple other objects. The new mask is then compared by applying bitwise & with each scoring area's mask separately starting from the areas with the highest score. If any non-zero pixels are found in the result, then the respective score of the scoring area is assessed to the hit and the hit evaluation is finished.

The automatic score calculation results are the same as when calculated by a human for scoring areas with six to ten points. As the scoring areas are further from the ellipse fitted between scoring areas of six and seven points, the automatic scoring might be less exact, as the application is unable to approximate deformations of the paper target in space (see Fig. 3.11). This can be prevented by attaching the target from the bottom as well as its top.

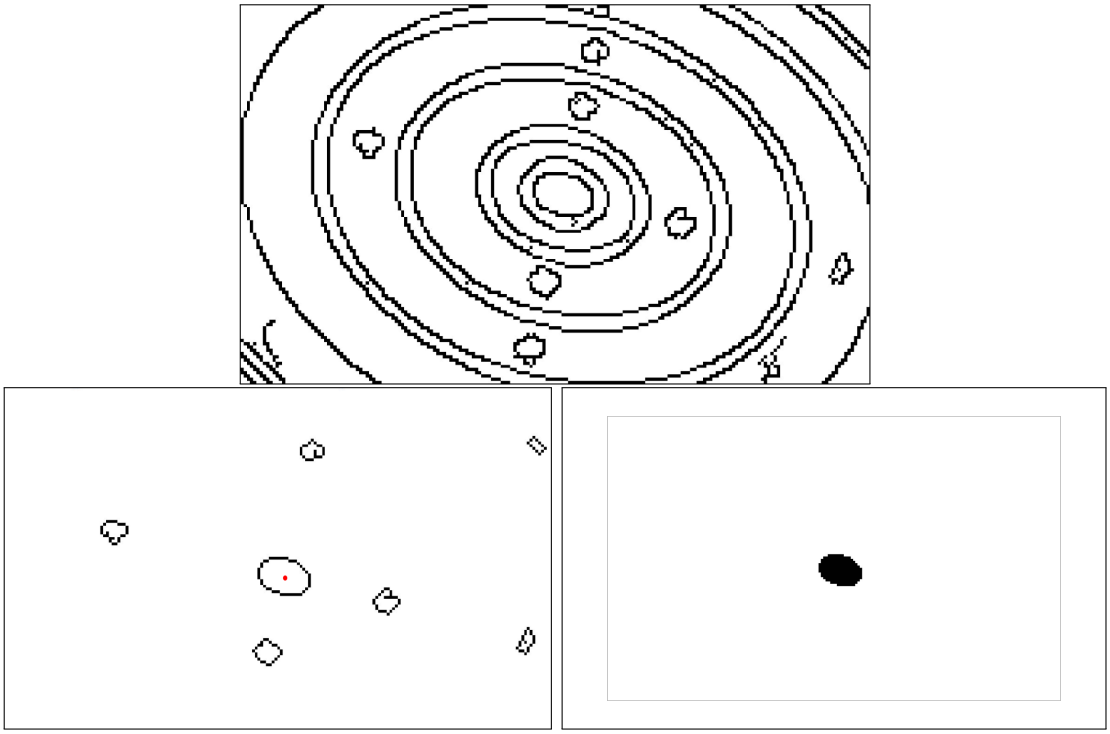

Figure 3.9: Estimating score areas as ellipses. **Top:** applying Canny's edge detection[v] on the middle of the target. **Bottom left:** Detecting the contours in the previous image and drawing only the contours which fit in size restrictions and have no children in the hierarchy in a separate matrix. The red dot in the center symbolizes the position of the center of the previously fitted ellipse. **Bottom Right:** The contours are filled using `cv::floodFill()`[vi]. Bitwise & is applied on each contour and the center of the previously found ellipse. If the intersection is not empty, then the contour is of the smallest target circle and an elipse is fitted on to it.

[v] `OpenCV:` Canny Edge Detector

[vi] `OpenCV:` Miscellaneous Image Transformations: `floodFill()`

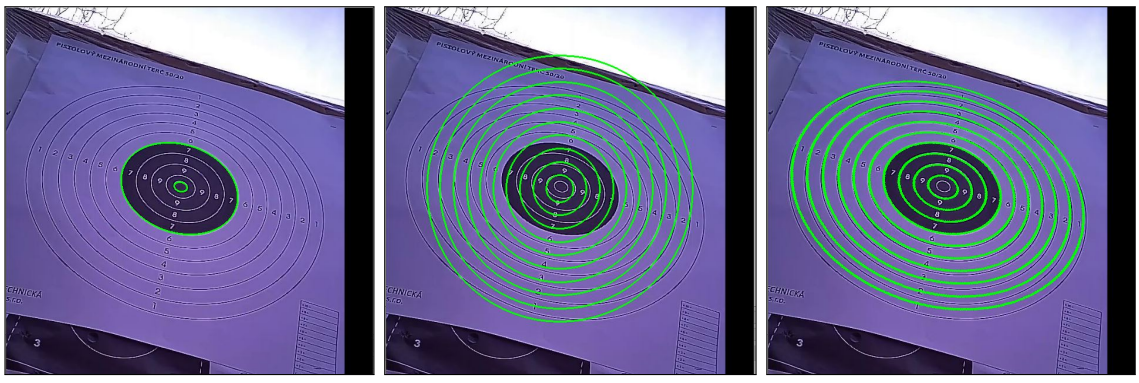

Figure 3.10: A more exact score area location. **Left:** one ellipse is fitted on the smallest score area border, another one is fitted in the borders of the black scoring area. **Middle:** Estimated scoring circles. **Right:** warped circles using homography.

Figure 3.11: Deformations of target in space. The score calculation might be less exact further from the target center, if the target is not attached both at its bottom and its top.

Chapter 4

Solution Requirements

Although bullseye shooting disciplines have worked without computer aid satisfactorily for many years, some areas could be made more effective, comfortable or appealing with little use of modern technology. In order to design and implement a valid solution, specific areas to be innovated have to be precisely determined first. So, what are these areas that the shooting training loses its efficiency in the most and could be changed with least effort? The answers will probably vary person to person depending on the shooting skill, training approach and knowledge of technology.

4.1 User Group Definition

This application will be used by people with interest in bullseye shooting with firearms. These people's age, professional specialization and experience with technology can vary. It is important to take that into account especially when designing a user interface. The application should be easy to use for any person which is able to shoot with firearms.

4.2 User Requirements

I decided to approach user requirements definition by defining the requirements myself first (as a shooter and a future user of the system), and then validating these requirements and changing them as necessary by running a survey.

As an intermediate shooter, seeing where my hits land during slow precision fire is very important. Usually I would have to identify my wrong movement tendencies after a shooting round is over. However, if I had an immediate feedback after a shot, I could determine certain movement tendencies and adjust my position right after seeing where the shot went through the target.

I use pistols of 9 mm caliber most frequently, and from my experience and research results, the 9×19 mm Luger Parabellum is the most popular handgun and submachine gun cartridge [35]. Therefore, the system should be compatible with guns of these cartridges in the first place. Other popular cartridges are the .223 Remington and .22 LR, and I would like the system was compatible with these cartridges as well. Different cartridges used differ in the size of the hit in the target.

When it comes to rapid fire, it is not that important to see where the hit landed as there is very little time in preparation for the next shot. In these cases, I would greatly

appreciate having a device to measure my shooting time, show it, and/or count only those shots that landed in the target during specific time frame.

As for score calculation, a simple application to do that for me could bring significant improvements in time efficiency. It would remove the necessity to wait for all shooters on the firing line to be done with their shooting rounds and the necessity to walk all the way to the target and to calculate the score. Reducing the amount of times a person has to cross the firing line would bring great improvements in safety and it would further improve time efficiency.

Furthermore, a system which could remember which hits belong to which round and highlight them in different colors would make changing the target or covering past hits between each round obsolete. Again, this saves some extra time.

Finally, I would welcome an easy way to save my shooting statistics to see how I progress over time and to analyze my past statistics in order to be able to determine the factors which affect my performance, and, if such effects are desirable or not. Currently I never record my statistics due to high amount of work which is needed to do that.

The finalized solution should, in addition, be safe, practical, comfortable to use, accessible and easy to work with for shooters of all ages and skill levels.

4.3 User Survey

In order to validate the user requirements I have specified in the previous section, I run a user survey. I asked subjects with experience with shooting from different backgrounds to participate. 18 subjects have participated in the survey.

The results were as follows:

1. **Which sports shooting disciplines are the most important for you?**

   In this question, the participants were able to list all the relevant disciplines. A participant could list as many disciplines as preferred. The results of this answers will help determine which disciplines the system described in this thesis will focus on.

Answer	Number of answers
Slow Precision Bullseye Shooting	8
IPSC[vii] Disciplines	7
Defense shooting	5
Rapid Fire	4

Table 4.1: User survey results: Most important shooting disciplines

[vii] International Practical Shooting Confederation: https://www.ipsc.org/

2. **How many different commercial shooting ranges do you visit?** The answers to this question will help determine what the physical solution and eventual shared statistics should focus on: In the case where most subjects visit one or two ranges, the system should be built-in at the range. If most people frequently visit different shooting ranges throughout the year, it would be useful if they could take the system to any range they prefer with them.

3. **What kinds of paper targets do you prefer?** The answers to this question will help to determine which target types I will focus on when creating the score evaluation. The subjects were allowed to give more answers than one.

Answer	Number of answers
I Visit Only One Shooting Range.	3
I Shoot at Two Different Shooting Ranges	3
I Frequently Visit Different Shooting Ranges (more than two)	12

Table 4.2: User survey results: The number of different shooting ranges a user will visit.

Answer	Number of answers
25 m Precision and 50 m Pistol Target	12
Human Silhouette	8
Other Types of Paper Targets	10

Table 4.3: User survey results: Most popular shooting targets.

4. **What are your preferred shooting distances?** The answers to this question will effect the physical solution design. The subjects were allowed to list any number of answers. All of the subjects' preferred shooting distances are between 10 and 25 m. 7 subjects also prefer shooting longer ranges (50 to 250 m).

5. **Which types of firearms do you use the most?** The answers to this question will help to determine which data should be collected for system development and testing. 16 answered with „9 mm pistols". 7 stated that they use rifles. Other answers varied.

6. **What is your experience level in bullseye shooting (or firearm shooting in general)?** The answers will help to determine methods used for score calculation and how much grouped hits need to be focused on.

Answer	Number of answers
Beginner	1
Intermediate	16
Expert	1

Table 4.4: User survey results: Skill levels of participating subjects.

7. **Is saving shooting statistics with the option of looking at pasts statistics important for you?**

   12 subjects answered with "Yes", 6 answered with "No".

8. **Would you consider the fact that a system for automatic score evaluation is integrated in a shooting range as a positive aspect which might make you more likely to visit the particular range?**

   10 subjects answered positively. 8 subjects stated that such system will not have any effect on their decision.

9. **What type of physical solution would you prefer to use with the score evaluating system?**

The answers to this question will help to determine the approach to the physical solution design.

Answer	Number of answers
I'd prefer taking my own camera and a computer with an installed application for score evaluation to the shooting range.	0
I'd prefer taking my own camera and a smartphone with an installed application for score evaluation to the shooting range.	2
I'd prefer cameras installed at the shooting range and taking my own smartphone with an installed application for score evaluation to the shooting range.	2
I'd prefer a fully integrated system at the shooting range.	8
I prefer calculating the score without the use of such systems, I like to do it myself.	2
I'm interested in all of the solutions mentioned above	2
Other	2

Table 4.5: User survey results: Preferred physical solutions.

10. **Would you be interested in acquiring a system for automatic score evaluation as an individual?** 9 subjects answered with „Yes“, 6 subjects stated that they would prefer the system was fully integrated at a shooting range, 3 subjects are not interested in acquiring the system at all.

4.4 Conclusions Derived From the User Survey Results

The results of the survey combined with my pre-definition of user requirements resulted in the following list of user requirements:

- the system will focus on slow precision bullseye shooting,
- the system should be built-in at a commercial shooting range,
- the system will be optimized for use with the $25\,m$ Precision and $50\,m$ Pistol Target,
- the system will be designed for shooting with firearms at distances from 10 to 25 m,
- the system will be optimized for use with firearms chambered in 9×19 mm Parabellum calibers,
- the scoring system will be optimized for intermediate shooters, therefore, grouped hits do not have to be taken into account during target hit detection and shooting score evaluation

- the system will save shooting statistics, so that they can be viewed at a later time.

These requirements do not limit the system in any way for future development. I made the list so that I know what to focus on in the first place, in order to release a working version as soon as possible, from which the most users will benefit. In the future, I would like to add more features (see Chapter 8) in order to make the system useful for even more firearm shooting enthusiasts.

Chapter 5

Overall Solution Design

5.1 Data Collection

In order to obtain the data necessary for development, I had to record videos of targets being shot at. Since the solution has to be built-in at a shooting range, I imagined that each shooting stall would be equipped with a camera focusing on the target respective for the particular stall. Therefore I used a camera (Canon EOS 600D with $85 - 200\,\text{mm}$ lens) placed right next to the shooter to take the video (see Fig. 5.1 top). The camera could also be placed above the shooter, on top of the shooting stall, but that would not make a difference large enough to be taken into account in the solution. This way, I have taken videos at three different shooting ranges. One of the shooting ranges is an indoor range with relatively poor light conditions. The other two shooting ranges are outdoor ranges. In the latter, I have taken videos in both sunny and rainy weather conditions.

I also experimented with using a WiFi camera which would be placed in the shooting field, near the target (see Fig. 5.1 bottom). I have taken videos at an outside shooting range with this setup.

5.2 Physical Solution Design

The physical solution consists of a camera, a computer and the infrastructure necessary for delivering the video output into the processing device. The possible camera placements are described in the previous section.

In the first case where the camera is placed on the firing line (next to or above the shooter), a relatively powerful camera with lens is required in order to acquire data in sufficient quality. The advantage of this is that the image is not curved in any way, providing accurate results. This solution is not very comfortable for individuals, as it requires the shooter taking additional equipment to the shooting range. However, it might be ideal for built-in solutions.

In the second case where the camera is placed in the shooting field, there is a small risk that the camera will be damaged. Also, when changing the shooting distance, the camera has to be moved with the target. Therefore, for attaching the camera, designated camera holders could be placed on the ground, or the camera could be attached to the target holder. The risk of damaging the camera could be minimized by placing the camera below the target rather than above it (most intermediate shooters tend to pull the gun upwards). In this case, a camera with relatively low resolution will suffice. The camera

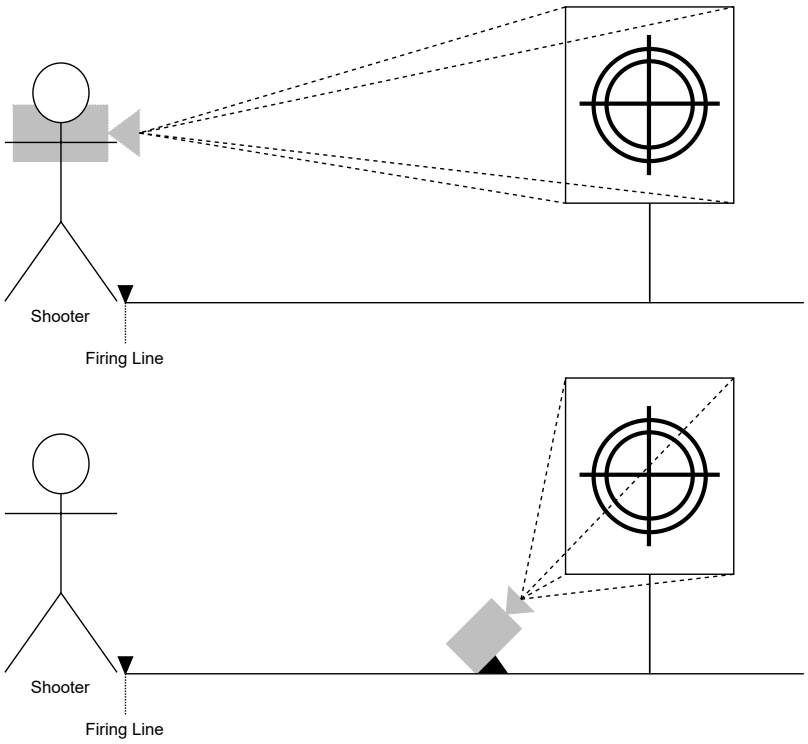

Figure 5.1: Camera placement options. **Top:** a camera with relatively powerful lens is placed on or right behind the firing line. **Bottom:** an inexpensive camera is placed near the target in the shooting field.

Figure 5.2: Sample images taken by cameras at different placements. Left: image taken by a camera from the shooter's position. Right: image taken by a camera near the target in the shooting field. Note the placement's impact on deformation of target scoring areas.

needs to be an inexpensive one, because there is a possibility (although not very probable) that the shooter will accidentally hit the camera. The data from the camera could be sent

to a processing device over WiFi. Due to the camera placement, the target is viewed at an angle which should be taken into account in processing of the camera output, as it could provide inaccurate results otherwise (see Fig. 5.2 and Fig. 3.10). The solution offers way more variability as opposed to the previous one and is suited for individual shooters to take and use wherever they would like to as well as shooting ranges as a permanently integrated system.

I have experimented with changing the camera resolution and found the minimum resolution for target hit detection as 640×480. However it is not ideal and some target hits might go undetected. The optimal camera resolution would be 1920×1080 with 30 frames per second. If the camera resolution is too large, the device might be too slow for processing the frames.

At the opposite point, the user is going to be communicating with the system through a user interface, running on a PC, a smartphone, or a tablet. For systems integrated at a shooting range, a touch screen would be ideal.

The target stand will remain unchanged at this time, but may have to be equipped with the camera holder in the future, in the case of the WiFi camera proving as the best solution.

5.3 User Interface Design

The user interface should be made as simple as possible both in appearance and controls, so that the user is not distracted from shooting. During shooting, besides their target with highlighted target hits, the user should be able to see only brief statistics (such as target hits, score count and time), just to have an overview about how they are doing. The detailed statistics can be viewed after the shooting round is over.

In order to create a quality user interface, the processes which the user is able to do have to be strictly defined first. I defined a use case for this reason.

The primary goal of this process is to receive score evaluation. The shooter starts by setting up their target and scoring system (unless it is provided by the shooting range). In the application, they can optionally select their shooting specifications (target distance, magazine capacity, time limit etc.) for a more exact score evaluation. Then they start the score calculation by clicking a start button and they can start shooting. After they are finished with the round, they stop the score calculation and view their statistics. From there, the shooter is able to repeat the whole act again.

The user interface should be as compatible as possible with touch screens and for usage without a keyboard, so that the system can be integrated in a shooting range with tablets as user input/output devices, and so that it is easy to make an application for smartphones with as little changes in the user interface as possible.

In the first version of the design, the user interface consisted of four windows. In the first window, the user would set their shot limit, target distance and the time limit. When ready, the user would continue on to the camera setup, where the user would see their camera output and detected score areas. The user would continue and start shooting from here. After some user feedback I have received, I integrated these two sections into one. This way, the user is always able to see the camera output, and does not have to change the whole window.

The final design of the user interface is divided in three sections:

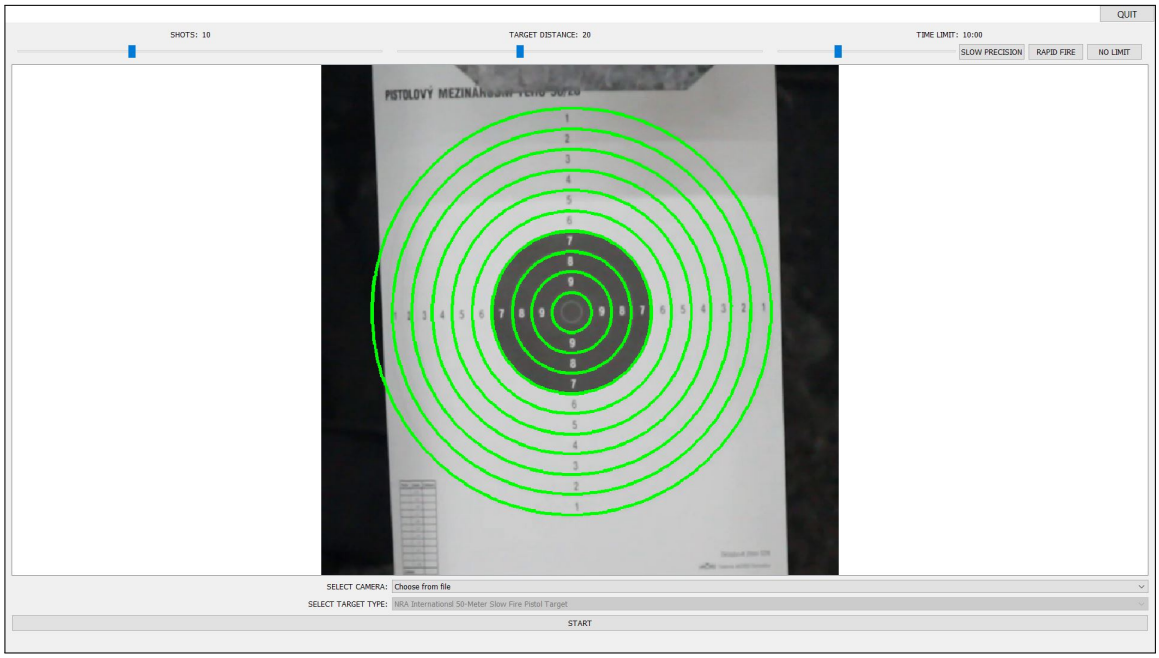

Figure 5.3: The setup window. On the top of the window, the user is able to set up after how many detected hits the score calculation stops, their target distance and time limit with designated sliders. The time limit can also be set using the buttons next to the time limit's slider: the "SLOW PRECISION" button sets the time limit to one minute per shot (e.g. provided that the user set their shot limit to 9, the time limit will be set to nine minutes); the "RAPID FIRE" sets the time limit in a similar manner to two seconds per shot and the "NO LIMIT" button removes the time limit altogether.

1. Setup – the initial setup window which is going to show camera output and detected score areas (see Fig. 5.3),
2. Shooting – camera output with highlighted hits, score count and time (see Fig. 5.4),
3. Detailed Statistics – this window is going to display an overview of more detailed statistics of the finished shooting round (see Fig. 5.5).

After another user's feedback and re-examining the way shooter training is organized, I decided to save the user settings between rounds. In one training session, the shooter will usually focus on a limited set of aspects which they would like to improve. Therefore, a shooter will usually do a couple of rounds with the same setup (target distance, shot limit and time limit), before changing the setup. Saving the settings between rounds prevents the user from having to change them before each round.

5.4 Plans for User Testing

Three sets of user tests will be carried out:

1. usability tests,
2. installation tests,

Figure 5.4: The window which is displayed during shooting. On the top of the window, the user is able to see how many shots they have left before the end of the round, their score so far, last hit's score and remaining time. Target with highlighted target hits is displayed in the middle of the screen. The shooter is able to stop the shooting round before running out of shots or time using the „CLOSE“ button.

3. end-to-end tests.

The first set will focus on interaction of the user with the interface. The users are not required to be present at a shooting range for this test. The shooting part of the test is not necessary and will be simulated with pre-recorded videos. This test will focus solely on user-friendliness of the interface. The user will be asked to use the application as if they were shooting, but instead selecting a connected camera, the user will select a pre-recorded video of a target being shot at. I will be watching the users to interact with the application myself and evaluate the results. This set will be carried out immediately after the interface is bug-free.

In the second set, the users will be asked to install the application on their device and connect a camera. The results of these tests will prove if the users will be able to install the solution themselves.

In the last set of tests, the users will be asked to use the solution at a shooting range during training. The results will determine if the solution is useful and might help to find final bugs and issues. This set of tests will be carried out in the last phase of development.

The subjects for participating in these tests will be found through the survey. Participants of the survey will be able to volunteer for testing.

5.5 Possible Distribution of the Solution

The system should be designed to be built-in at a shooting range (see requirements in Section 4.4), but I would also like to consider the possibilities for making the solution

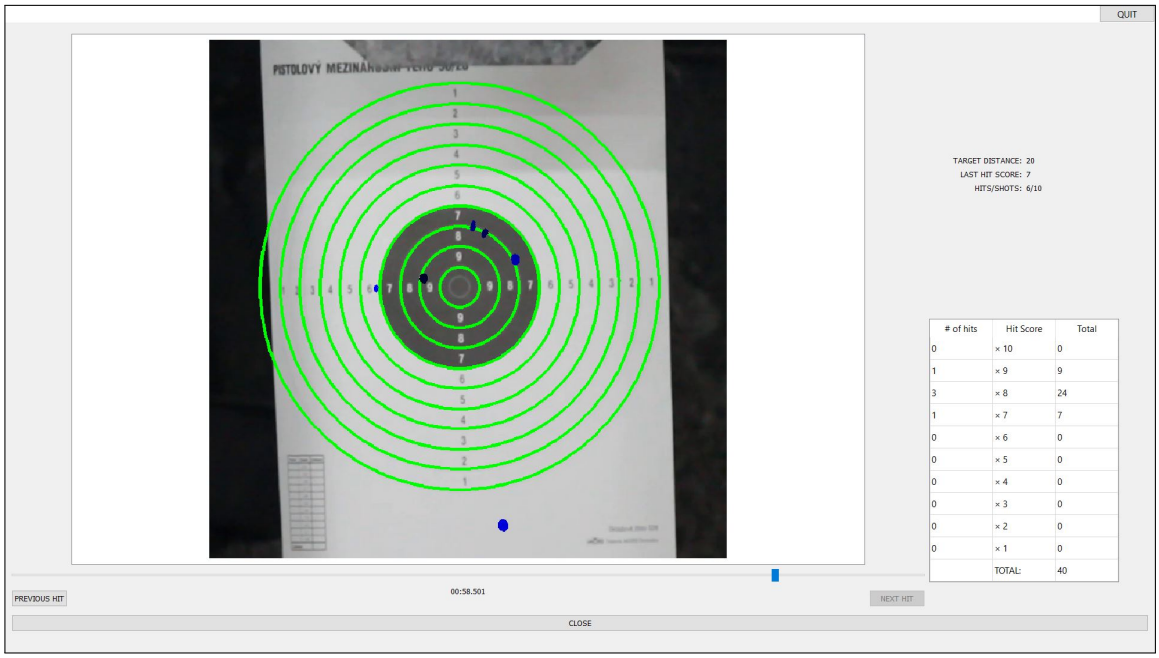

Figure 5.5: The statistics displayed after the shooting round ends. Using the slider on the bottom (history slider) of the picture of the target, the shooter is able to view the target at different times during the shooting round. Target hit illustrations appear and disappear on the illustration of the target depending on the selected time on the slider. The most recent hits are displayed in the lightest color. The user is also able to see the score evaluated for the last hit displayed using the history slider. For easier navigation, there are two buttons on the bottom of the slider. The user can use these to view the next or the previous detected target hit. On the right bottom side of the screen, there is a table with a more detailed overview of the shooting score. The user can return to the setup screen with the “CLOSE” button.

available for individuals at this phase. The application stays the same in both the case when it is built-in at a shooting range and for use by individuals. The physical solution depends entirely on either the user’s or the shooting range’s preferences.

The system could be offered either as proprietary or license. Considering that there would probably be a relatively large amount of maintenance work needed to keep the system running, a license system is probably more suitable in this case. This also potentially makes the system more available to a larger amount of users, which is a desired aspect.

Chapter 6

Implementation

In this chapter, I will briefly describe the implementation, so that the reader has an overview about how the program functions. For the detailed description of implementation, see the complete documentation available on the attached data medium.

I have decided to implement the application in two phases. In the first phase, I will implement score evaluation from a pre-recorded video. In the second phase, I will implement a user interface and integrate a camera so that the application can work in real time. I will test the result of each phase on all the data I have collected (see Section 5.1) prior to running user tests as described in Section 5.4. The whole project will be developed and tested on Windows 10 Education, Version 20H2, 64-bit operating system, x64-based processor.

6.1 Implementation of Score Evaluation From a Pre-recorded Video

I implemented the whole project in C++. In the first phase, I have used the Visual Studio 2017[1] for an integrated development environment and I installed the OpenCV library. I tested the functionality by visualizing the output using the OpenCV `cv::imshow()`[2] function.

Target Detection And Ellipse Fitting

The video is processed in the `VideoProcessor` class. The `LoopVideo()` function processes the whole video frame by frame. The same class also detects target hits and evaluates shooting score. On the first processed frame of the video, an instance of the `ScoreAreaLocator` class is created. The class is responsible for locating target areas and storing these areas. The `FindCropRect()` function locates the target in the first processed frame of the video as described in Section 3.1. The reference image of a shooting target which the camera output image is searched for is opened with the `OpenResImg()` function. The resulting cropped image is then processed with the `FindRadiusIncr()` function. The function locates two ellipses in the target and stores them as `cv::RotatedRect` in a vector of the `ScoreAreaLocator` class instance.

[1] https://visualstudio.microsoft.com/vs/older-downloads/

[2] OpenCV: High-level GUI: imshow()

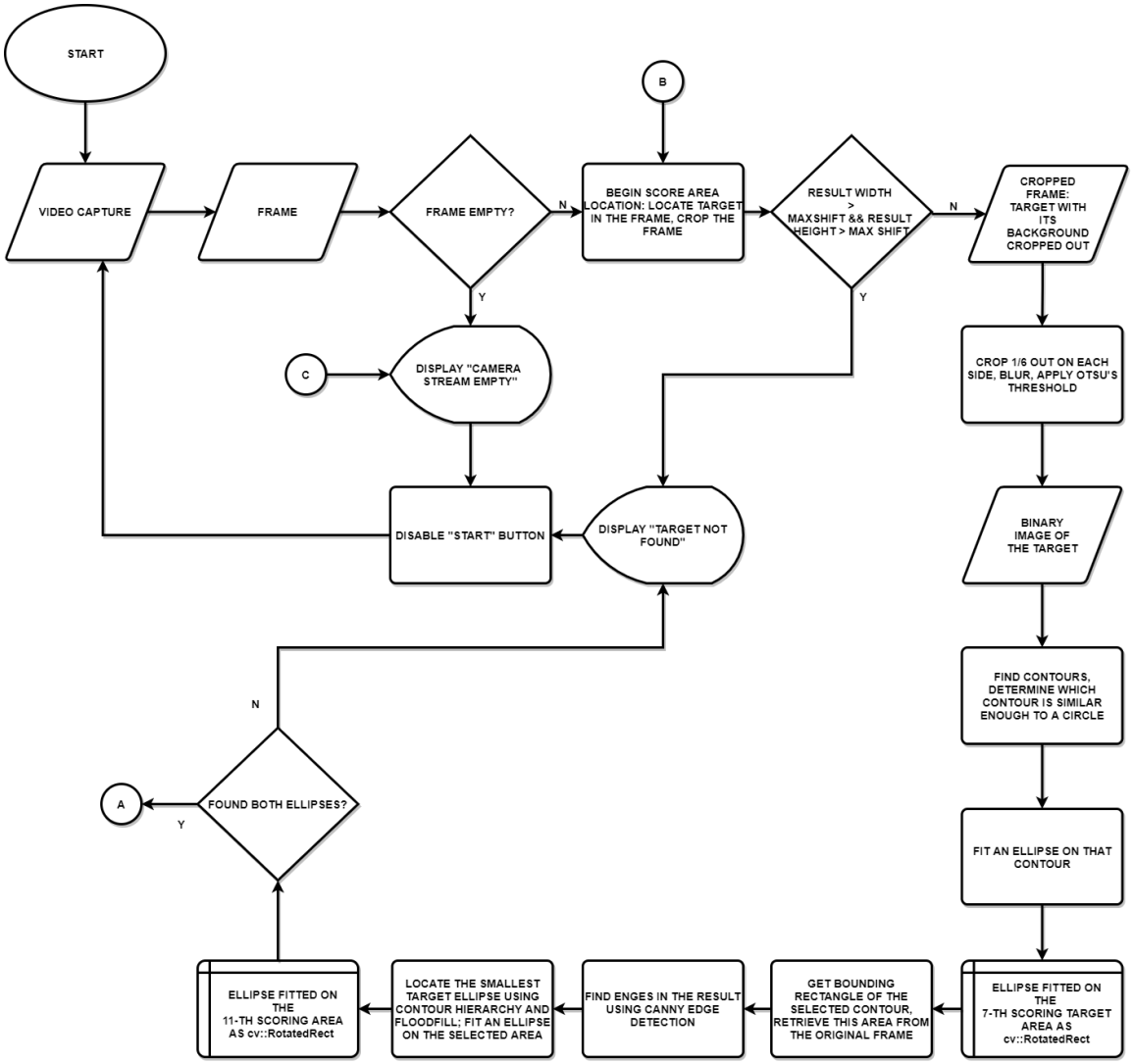

Figure 6.1: Flow chart diagram of the application: target detection, ellipse fitting.

Estimating Score Areas

The scoring areas are estimated as circles in the `GetScoreCirclesForHomography()` member function of the `ScoreAreaLocator` class. The function uses the detected black circle in the middle of the target to find the radius of each scoring circle. Following that, the `DrawScoreCircles()` draws the estimated scoring areas as concentric circles in a binary image. The center of the previously fitted innermost ellipse is used as the center of these circles.

Approximation of Estimated Scoring Areas

From each of the ellipses found in Section 6.1, four points are extracted using the function `cv::ellipse2Poly()`[3]. These eight points are later used to find homography as described

[3] `OpenCV: Drawing Functions: ellipse2Poly()`

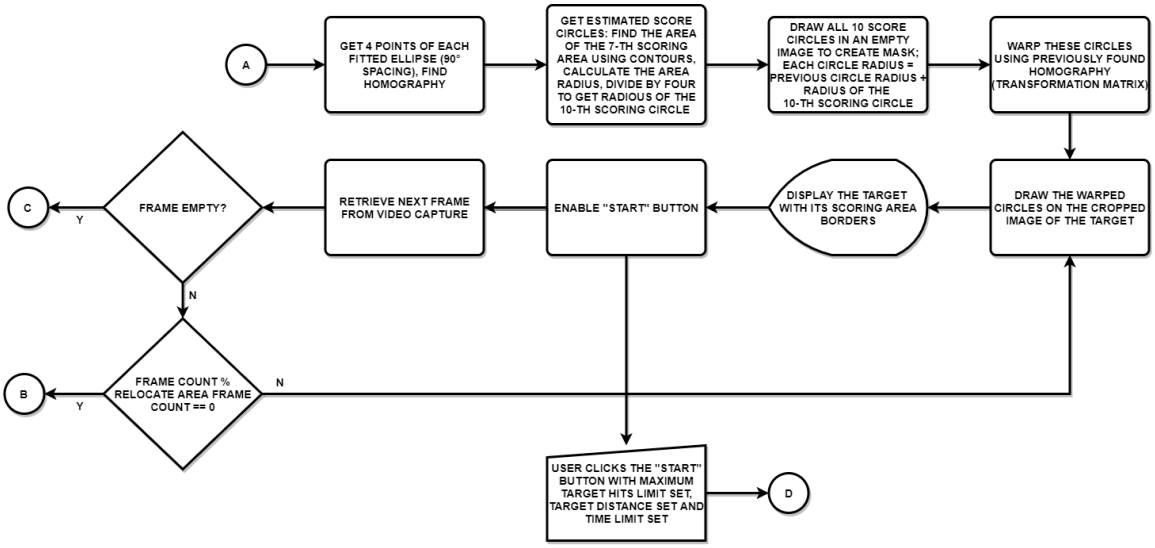

Figure 6.2: Flow chart diagram of the application: find homography, get estimated scoring areas, warp the areas using the previously found homography to get more accurate locations of scoring areas.

in Section 3.2. The scoring area estimated as circles are approximated by warping using the previously found homography.

Image Stabilization

The `VideoProcessor` class is responsible for image stabilization. In the `FFTCorrelate` member function, cross-correlation is applied on the currently processed frame and the first processed frame. The caller then finds the shift of the frame and aligns the currently processed frame with the first processed frame.

Target Hit Detection

In the aligned frame, background is subtracted. The foreground is then searched for objects corresponding to the conditions of target hits as described in Section 3.1. The objects which do not fit the given conditions are filtered by the `ContourMatchesHitSpecifications()` function. Before marked as a new target hit, intersection over union is calculated on the detected object. Provided that the program evaluates the first detection of the object, a new instance of the `TargetHit` class is instantiated. The instance is saved in the `Target` class in a vector of all target hits detected in the current shooting round.

Score Calculation

The `UpdateTargetHitsAndFinishRund()` member function of the `VideoProcessor` class is responsible for calculating score and initiating the end of the shooting round, if maximum target hits was detected. The score is calculated as described in Section 3.3.

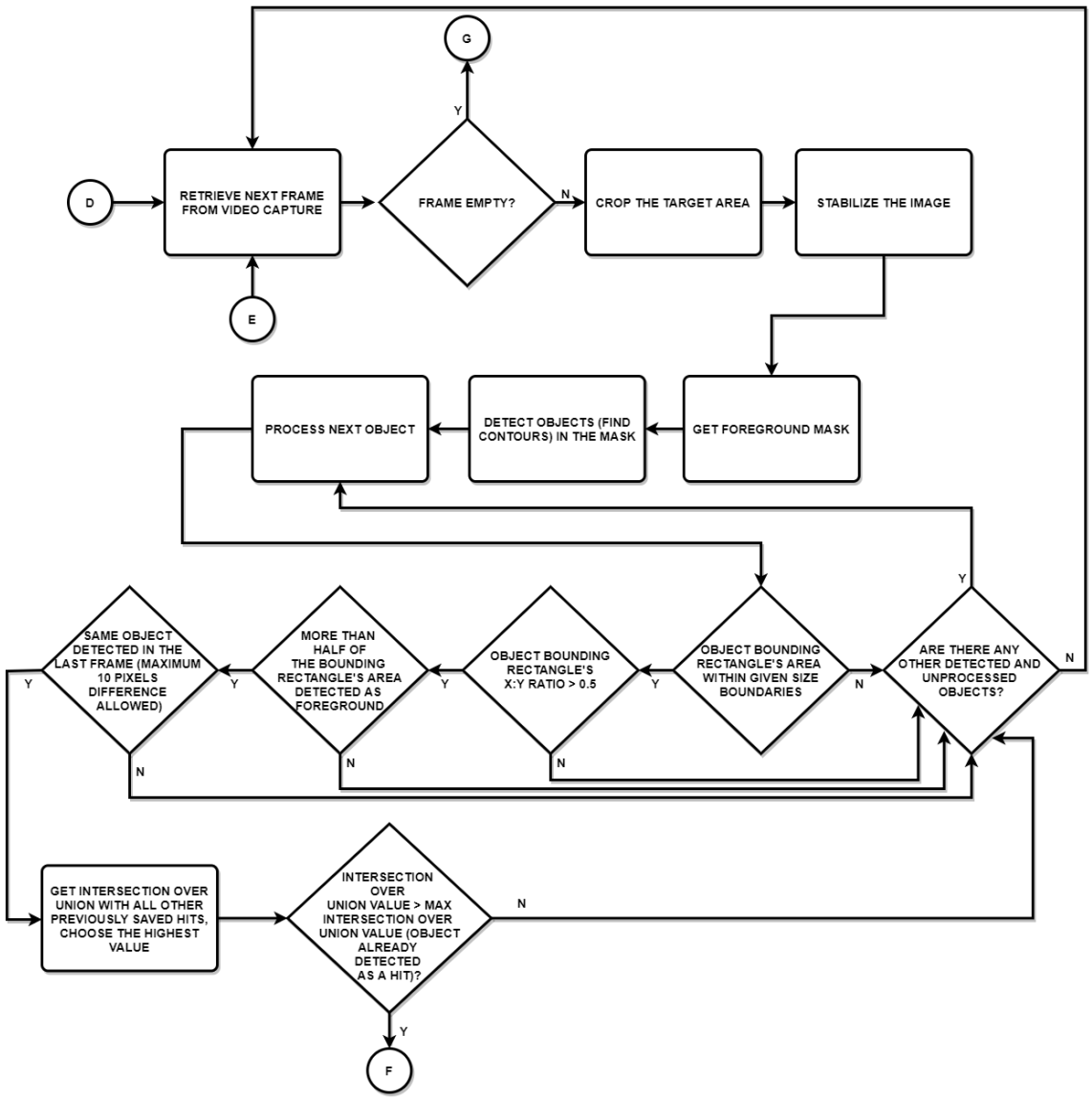

Figure 6.3: Flow chart diagram of the application: image stabilization, target hit detection.

6.2 User Interface Implementation

I have used the Qt framework[4] (version 5.12.2) for user interface implementation. In this phase, I switched integrated development envirements and used QtCreator in place of Microsoft Visual Studio, because of easier integration of the Qt framework in the IDE.

I used the MinGW compiler to compile the project. The default setup did not support OpenCV libraries, so I had to download the OpenCV sources again in version 4.2.0[5]. I used CMake to generate a MinGW makefile, with the `WITH_QT` option checked. Following that, I run the "`mingw32-make -j 8`" and "`mingw32-make -install`" commands in the folder with OpenCV build.

[4] https://www.qt.io/product/framework
[5] https://opencv.org/releases/page/2/

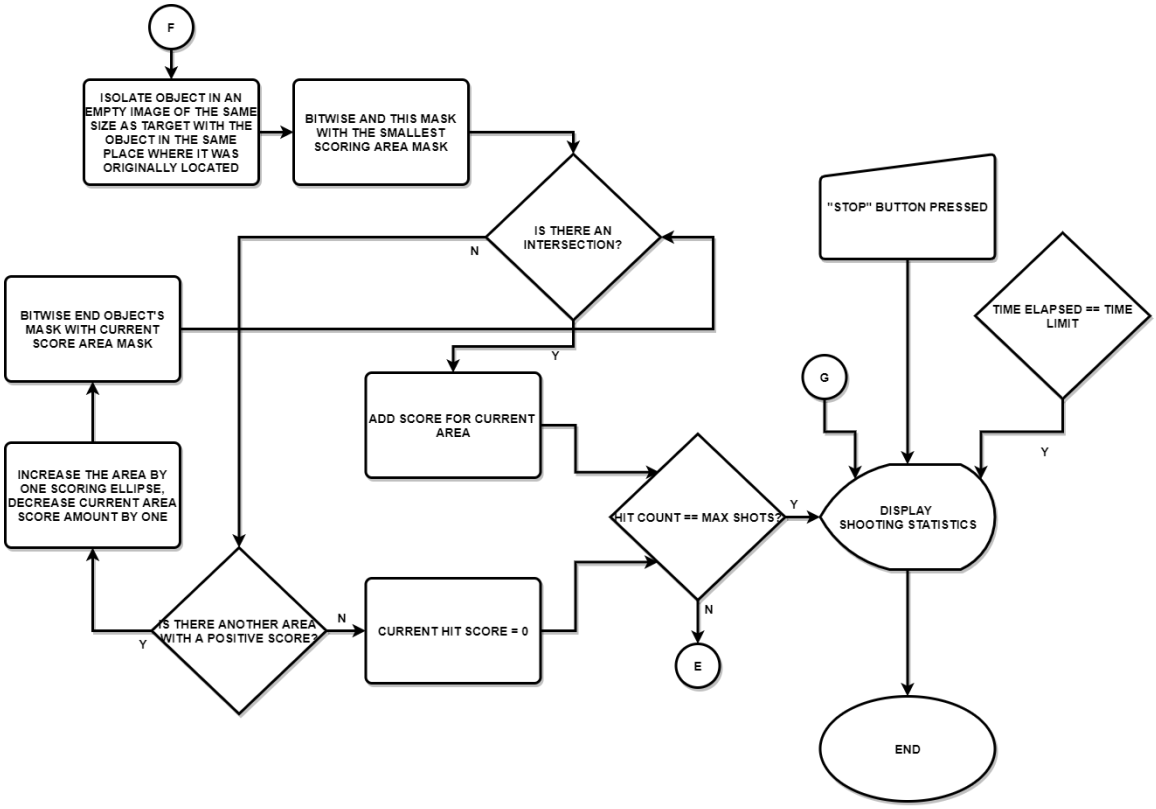

Figure 6.4: Flow chart diagram of the application: score calculation.

All of the user interface is implemented in the `MainWindow` and the `InputDialog` classes. The main window changes dynamically depending on user's actions.

Chapter 7

Testing and Evaluation

Usability Tests

This set of tests focuses on testing the user-friendliness of the interface. The subject is left to work with the application on their own and observed during the process. The subject explores the functionality of the application without any advice or user manual. For this phase of testing, I chose two different subjects, both of which have average experience with using a computer; one of which is an active shooter, the second with no shooting experience whatsoever. I decided to test the application on a subject without any previous shooting experience to prove that the application could be used with beginner and even first time shooters without any issues.

Both of these subjects were able to select a sample video and have it evaluated by the software. Both subjects found the settings and experimented with different setups. Both subjects were able to use the screen with statistics to read the final scores and to further analyze the shooting round using the history slider.

One of the subjects suggested that it might be useful to have the program open the first selected camera on startup. In the initial phases of the user interface development, this was implemented. However, I decided to change it later to the application starting with an empty screen, so that the user does not feel like their privacy is violated in any way by the application opening their camera and starting to evaluate its output data without the user's permission. I plan to leave it as-is until given reasons for a change which would outweigh the privacy aspect. I intend to make the user feel that they control the application and that they keep the control over it at all times.

Installation Tests

The installation of the application is relatively simple. The user extracts their received .zip archive on their computer and run the executable file. All subjects tested were able to perform the task without any issues.

End-to-end Tests

I was unable to perform this set of tests as I did not have resources left for this phase. I would like to conduct a study for this phase in the future to determine the usefulness of the solution in numbers. I will compare shooters' learning curves of bullseye shooting without any training help, with the application used during the training and with an experienced

professional trainer. I will also make a questionnaire for these subjects and examine their personal satisfaction levels with the shooting itself and their learning process.

Evaluation

For the detailed evaluation of used algorithms, see Chapter 3. For a summary and comparison of methods used for target hit detection, see Table 3.1.

Chapter 8

Conclusion

I created a solution which automatically calculates shooting score from a video of a target which is being shot at. The solution is suitable both for use by individuals taking their own camera and a computer to a shooting range, and to be built-in at a commercial shooting range, so that shooters will already have the solution set up and ready for use. I implemented target hit detection and score calculation from a video of a target. The input video can be retrieved in real time or a pre-recorded video.

While taking sample videos simulating what the application will process, I have experimented with different camera setups, different camera resolutions and other external conditions in order to find the minimum camera requirements and the external conditions requirements. The target can be viewed from varying angles, but the camera has to remain in one place during a shooting round. The distance between the target and a shooter can vary and the maximum distance depends on the camera and the camera lens used. A Canon 600D with $80 - 200$ mm lens is able to capture the target at 25 m with sufficient score evaluation results.

The system was optimized for use with firearms chambered in 9×19 mm Parabellum caliber, but probably most likely will still be accurate enough in use with other most common calibers and even some air guns. In the future, I would like to test some other most common calibers and implement some optimizations regarding the caliber/hit size, if necessary.

The application currently supports the International 25 m Precision and 50 m Pistol Target. The target should be attached both at its top and bottom for most accurate results.

Saving shooting statistics is not possible within the application at this time. I would like to implement this functionality in the nearest future along with a possibility to export shooting statistics into one or several formats which will be selected upon results of a user survey.

I implemented a user interface for the program. The user interface is user-friendly enough to be used by users with basic knowledge regarding computers. Users having no experience with shooting are able to understand the user interface as well, therefore it is also suitable for new shooters.

Currently, the program detects grouped hits as one target hit. I would like to address this in the future by either amending the part of the program which is responsible for target hit detection, or implementing differing shooting score evaluation metrics for experienced shooters altogether.

I plan to make a user application for mobile phones next in order to make the solution more convenient for individuals. As for functionality, I would like to implement score evaluation for other paper target options (silhouette, targets with multiple bullseyes, rapid fire targets etc.) as well as reactive targets and steel targets. Additionally, I would like to implement a unified scoring system between different bullseye shooting disciplines as an experiment. Also, the system could possibly be able to analyze shooter's movement tendencies and provide with some training advice or suggested drills.

Later I'd like to conduct a study in which I will compare beginner and intermediate shooters' skill over time. I'll divide the shooters into three groups, one of which will train without any external help; the second group will train with an experienced human trainer and the third group will train with the application I made and without any other external help. This will provide quantitative results of the usefulness of the solution.

Bibliography

[1] NATIONAL RIFLE ASSOCIATION OF AMERICA. *NRA PRECISION PISTOL RULES*. May 2018. Available at: https://competitions.nra.org/documents/pdf/compete/RuleBooks/Pistol/pistol-book.pdf.

[2] ROBAZZA, C. *ISSF coach (second level) sport science*. Nov 2020. Available at: https://www.issf-sports.org/getfile.aspx?mod=docf&pane=1&inst=497&file=1.Sport_Science_material.pdf.

[3] NATIONAL RIFLE ASSOCIATION OF AMERICA. *NRA Tournament Operations Guide: The Ultimate Guide to Conducting NRA Sanctioned Matches*. Dec 2020. Available at: https://issuu.com/compshoot/docs/tourn_ops_guide/32.

[4] INTERNATIONAL SHOOTING SPORT FEDERATION. *6. GENERAL TECHNICAL RULES*. Feb 2020. Available at: https://www.issf-sports.org/getfile.aspx?mod=docf&pane=1&inst=458&file=1.%20ISSF%20General%20Technical%20Rules.pdf.

[5] DEEP SCORING LTD. *TargetScan ISSF Pistol & Rifle (Beta)*. Feb 2021. Available at: https://play.google.com/store/apps/details?id=com.gabrowski.targetscan&hl=en&gl=US.

[6] DEEP SCORING LTD. *TargetScan – pistol & rifle*. Sep 2020. Available at: https://apps.apple.com/ca/app/targetscan-pistol-rifle/id448045769#?platform=iphone.

[7] MON, D. and TEJERO GONZÁLEZ, C. Validity and reliability of the TargetScan ISSF Pistol & Rifle application for measuring shooting performance. *Scandinavian Journal of Medicine and Science in Sports*. Nov 2019, vol. 29, p. 1707–1712. DOI: 10.1111/sms.13515.

[8] AC APPS. *Target Scanner for Competition Shooters*. Sep 2020. Available at: https://play.google.com/store/apps/details?id=target.scanner&hl=en&gl=US.

[9] SHOOTER'S TECHNOLOGY LLC. *Athena: The World's First Affordable Electronic Scoring Target*. Available at: http://www.orionscoringsystem.com/orion/AboutOrion.aspx.

[10] SHOOTER'S TECHNOLOGY LLC. *Orion Scoring System Bundles*. Available at: http://www.orionscoringsystem.com/orion/BundleList.aspx.

[11] DISAG GMBH & CO KG. *Elektronische Scheibenauswertung von DISAG*. Jul 2018. Available at: https://www.disag.de/produkte/rm-iv/.

[12] International Shooting Sport Federation. *Results of ISSF Certification Tests for Electronic Scoring Targets*. Nov 2018. Available at: https://www.issf-sports.org/getfile.aspx?mod=docf&pane=1&inst=31&iist=29&file=Results_of_ISSF_Certification_Tests_for_Electronic_Scoring_Targets_-_Version_November_2018.pdf.

[13] Knestel Elektronik GmbH and DISAG GmbH & Co KG. *OPERATING INSTRUCTIONS DISAG RM IV Ring and target measuring machine*. Sep 2020. Available at: https://www.disag.de/wp-content/uploads/Bedienungsanleitung-Englisch-14te-Ausgabe.pdf.

[14] Megalink AS. *4K187 target*. Available at: https://www.megalink.no/en/products/10m-15m-targets/4k187-target.html.

[15] SIUS AG. *S310*. SIUS AG. Available at: https://sius.com/wp-content/uploads/2018/07/B-KB-S310-en-1.pdf.

[16] Megalink AS - Electronic Scoring System. *PC-based target 10m*. Available at: https://www.megalink.no/en/products/pc-based-personal-targets/pc-based-target-10m.html.

[17] DISAG GmbH & Co KG. *Der OpticScore Messrahmen*. Feb 2020. Available at: https://www.disag.de/produkte/opticscore/messrahmen/.

[18] Godt Sagt, Kongsberg Target Systems. *FREQUENTLY ASKED QUESTIONS*. Apr 2021. Available at: https://www.kongsbergtargets.com/faq/.

[19] Godt Sagt, Kongsberg Target Systems. *3-LANE 10M OPTICSCORE ELECTRONIC TARGET SYSTEM*. Oct 2020. Available at: https://www.kongsbergtargets.com/products/3-lane-10m-opticscore-electronic-target-system/.

[20] SIUS AG. *LS25/50 LASERSCORE*. Oct 2020. Available at: https://sius.com/en/product/ls25-50-laserscore-11/.

[21] SPORT QUANTUM. *Technology*. May 2019. Available at: https://sportquantum.com/en/technology/.

[22] SPORT QUANTUM. *SQ 50 – INTERACTIVE TARGET FOR SPORT SHOOTING AT 25 AND 50 METERS*. Apr 2020. Available at: https://sportquantum.com/en/targets-shooting-sport/sq50/.

[23] Suarez, R. *Sport Quantum Air Pistol & Rifle Electronic Target Review*. Jul 2020. Available at: https://www.olympicpistol.com/sport-quantum-air-pistol-rifle-electronic-target-review/.

[24] Brown, A. *FreETarget* [https://github.com/ten-point-nine/freETarget/releases/tag/1.6.1]. 1.6.1. Jan 2021.

[25] Waterman, M. and Salazar, D. Electronic Target. In:. Dec 2011, p. 4 – 16.

[26] PITSHOOTING.EU. *PROGRESSIVE INTELLIGENT TARGET SYSTEM*. Jun 2019. Available at: https://www.pitshooting.eu/#catalog.

[27] ABVISIE. *MATCH MANAGER*. Jun 2014. Available at: https://www.abvisie.nl/downloads/matchmanager-info-en.pdf.

[28] ENFILADE. *About Rifle Target: Rifle Shooting Database*. Available at: https://enfiladestudios.com/about-rifle-target-rifle-shooting-database.html.

[29] ZIVKOVIC, Z. Improved Adaptive Gaussian Mixture Model for Background Subtraction. In:. September 2004, vol. 2, p. 28 – 31 Vol.2. DOI: 10.1109/ICPR.2004.1333992. ISBN 0-7695-2128-2.

[30] OPENCV. *OpenCV: Open Source Computer Vision*. Oct 2019. Available at: https://docs.opencv.org/4.1.2/index.html.

[31] ZIVKOVIC, Z. and VAN DER HEIJDEN, F. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*. 2006, vol. 27, no. 7, p. 773–780. DOI: https://doi.org/10.1016/j.patrec.2005.11.005. ISSN 0167-8655. Available at: https://www.sciencedirect.com/science/article/pii/S0167865505003521.

[32] MALLICK, S. *Feature Based Image Alignment using OpenCV (C++/Python)*. Mar 2018. Available at: https://learnopencv.com/feature-based-image-alignment-using-opencv-c-python/.

[33] ROSEBROCK, A. *Intersection over Union (IoU) for object detection*. Nov 2016. Available at: https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/.

[34] GREENSTED, D. A. *Otsu Thresholding*. Jun 2010. Available at: http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html.

[35] BARNES, F. C. *Cartridges Of The World*. 8th ed. Charles T. Hartigan, 1997. ISBN 0-87349-178-5.

Appendix A

Demonstration Video

The video demonstrating the application described in this thesis is available on this link: https://youtu.be/_zTaVjfcVs8.

Appendix B

Attached Data Medium Contents

Compiled Solution

To access the compiled solution, open the "Compiled" folder on the attached data medium and run the executable.

Source Codes

To compile the solution from source:

1. Download and install CMake[1].
2. Download OpenCV 4.2.0 sources[2].
3. Download and install MinGW[3].
4. Find the "bin" folder in your MinGW installation folder and add it into your "Path" system environment variable.
5. Download Qt creator[4].
6. During the Qt installation process, tick the following:
   - **Qt 5.12.2**: MinGW 7.3.0 32-bit
   - **Developer and Designer Tools**: Qt Creator 4.8.2 CDB Debugger
7. Open CMake and paste Your OpenCV 4.2.0 folder locations with the source files in the "Where is the source:" input line.
8. Select the folder for the build.
9. Select MinGW Makefiles as the generator for the project, select the "Use default native compilers" option and click "Finish".

[1] CMake can be downloaded from this page: https://cmake.org/download/

[2] OpenCV 4.2.0 can be found here: https://opencv.org/releases/page/2/

[3] MinGW can be downloaded from here: https://sourceforge.net/projects/mingw-w64/files/Toolchains%20targetting%20Win32/Personal%20Builds/mingw-builds/installer/mingw-w64-install.exe/download

[4] Download the executable from this link: https://download.qt.io/official_releases/qt/5.12/5.12.2/

10. Uncheck the "ENABLE_PRECOMPILED_HEADERS" option, check the "WITH_OPENGL" and "WITH_QT" options and select the "Qt5_DIR" folder. Click on the "Configure" button, and then on the "Generate" button.

11. Open the command prompt and go to Your folder with OpenCV 4.2.0 build.

12. Execute the "`mingw32-make -j 8`" command.

13. Execute the "`mingw32-make install`" command.

14. Open the .pro file in the "Source" folder in Qt Creator.

15. Build the project, run qmake and then run the program.

Documentation

1. **Pre-compiled** – to access the documentation, open the "index.html" file in Your browser (Google Chrome is recommended). The file is located in the "./doc/html" folder in the "Compiled" folder. You can also paste the "Compiled" folder into Your C:// folder, open the "Compiled" folder and open the "Documentation" in Your browser.

2. **Generate from sources** – install Doxywizard[5] and GraphViz[6]. Open the "Doxyfile" file from the "Source" folder on the digital medium in Doxywizard. Select the "Run" tab in Doxywizard and click "Run doxygen". You can either view the HTML ouput from there, or go back to the "Source" folder on the digital medium. Open the "doc" folder, then the "html" folder, find the "index.html" file and open it in Your browser.

Sample Shooting Videos

You can find the sample videos in the folder "sample_videos" in the "Compiled" folder. These can be selected in the application instead of a camera.

[5] You can download Doxywizard from this page: https://sourceforge.net/projects/doxygen/.

[6] The GraphViz can be downloaded from this page: https://graphviz.org/download/. Select the stable version according to the system You are using.