**Automated Surveillance using Background Subtraction and Shape Analysis**

Final Report for CS39440 Major Project

*Author*: Alexander Shaun O’Donnell ([alo14@aber.ac.uk](mailto:alo14@aber.ac.uk))

*Supervisor*: Bernie Tiddeman (bpt@aber.ac.uk)

16th April 2016

Version 1.0 (Draft)

This report is submitted as partial fulfilment of a BSc degree in  
Computer Science (Inc Integrated Industrial and Professional Training) (G401)

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Wales, UK

**Declaration of originality**

I confirm that:

* This submission is my own work, except where clearly indicated.
* I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
* I have read the regulations on Unacceptable Academic Practice from the University’s Academic Quality and Records Office (AQRO) and the relevant sections of the current Student Handbook of the Department of Computer Science.
* In submitting this work I understand and agree to abide by the University’s regulations governing these issues.

Name Alexander Shaun O’Donnell



Date 16/04/2017

**Consent to share this work**

By including my name below, I hereby agree to this dissertation being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name Alexander Shaun O’Donnell



Date 16/04/2017

**Acknowledgements**

I am grateful to…

I’d like to thank…

**Abstract**

Computer vision and image interpretation are evolving topics in the world of technology. Many institutions have invested in this field of research for security solutions, as technologies such as facial recognition, object placement and pedestrian detection can provide potential safety nets to the challenges they face. The rise of automation technologies is reducing human error, which is something a high-risk field such as security could do without. This project looks at pedestrian detection/tracking, the idea that a program could take up the role of monitoring live CCTV footage and keep a record of pedestrians and other movements. We can achieve this by using what is known as a classifier, an algorithm that can lead software to make categorical decisions. By providing examples of how pedestrians may appear in a video, the classifier can distinguish between what is and what isn’t a person. Assuming the cameras used in the videos are static, moving objects can be separated from a non-moving environment using background subtraction. Combining these tools, we can extract large moving objects from video footage and use the classifier to see whether the object is a person or not. Rather than having a police officer go through hours of CCTV footage, this program can view the video instead, recording any passenger or movement that occurred with a timestamp of when it happened.

**Contents**

1. Background, Analysis & Process 6

1.1. Background 6

1.1.1. Security and Automation 6

1.1.2. Pedestrian Detection 7

1.2. Analysis 7

1.3. Process 7

2. Design 8

2.1. Overall Architecture 8

2.2. Detailed Design 8

2.2.1. Even More Detail 8

2.3. User Interface Design 8

2.4. Other Relevant Sections 8

3. Implementation 9

4. Testing 10

4.1. Overall Approach to Testing 10

4.2. Automated Testing 10

4.2.1. Unit Tests 10

4.2.2. User Interface Testing 10

4.2.3. Stress Testing 10

4.2.4. Other Types of Testing 10

4.3. Integration Testing 10

4.4. User Testing 10

5. Critical Evaluation 11

6. Appendices 12

A. Third-Party Code and Libraries 12

B. Ethics Submission 13

C. Code Samples 14

Annotated Bibliography 15

# Background, Analysis & Process

This section should discuss your preparation for the project, including background reading, your analysis of the problem and the process or method you have followed to help structure your work. It is likely that you will reuse part of your outline project specification, but at this point in the project you should have more to talk about.

**Notes:**

* **All of the sections and text in this example are for illustration purposes. The main Chapters are a good starting point, but the content and actual sections that you include are likely to be different.**
* **Look at the document on the Structure of the Final Report for additional guidance.**

## Background

### Security and Automation

As an interdisciplinary field, computer vision has provided many solutions to security. From facial recognition for matching perpetrators to crowd analysis for tackling gang culture and hooliganism, it synergises well with high risk environments. Originating in academic environments, computer vision made its way into the world of security as early as 2001. Police in Tampa, Florida used facial recognition software during the Super Bowl XXXV in January 2001 to detect known criminals and terrorists attempting to attend the event. The software, known as “FaceIt” was developed by Identix and managed to find 19 people with pending arrest warrants [1].

Computer Vision is an abstract topic, the idea of getting a system to understand what is being depicted in an image is particularly useful for security. Many institutions have researched this field as it can provide solutions to some of the challenges faced by the industry. For example, in 2015 the Ministry of Defence (MoD) cooperated with QinetiQ to develop middleware software architecture that would be used alongside sophisticated sensing equipment to perform detection and behaviour classification in a specified area [2].

A typical surveillance setup would involve a security guard or member of law enforcement observing several cameras. These cameras are usually static, where a single camera will maintain the same viewing angle for its duration. This results in institutions investing in many cameras to cover all angles of a premises, and can be straining on the employee tasked to observe them all. This is amplified in isolated or rural environments, where cameras may only observe a small number of pedestrians every hour leading to fatigue and thus an increase in human error. Previous studies have coined the term ‘boredom factor’, where an observer’s levels of concentration degrade over time from viewing the same monotonous feeds for long hour shifts [3].

Part of the job for police officers is to review pre-recorded CCTV footage recorded for potential evidence, with footage lengths ranging from minutes to days. In 2016 CCTV operators from the Suffolk Constabulary and the Metropolitan Police observed more than 1100 hours CCTV footage in the search for a missing person [4]. The hours spent on this particular case involved detection rather than recognition, as they do not include identifying the individuals found. The ability to automate this process would save time, resources and manpower that could be better spent in agencies such as these. It is vital that our law enforcement adapts and becomes more efficient at retrieving and storing information for the sake of the victims and our privacy. To automate the process of collecting timestamps of people appearing in videos, we can look no further than the field of pedestrian detection.

### Pedestrian Detection

Pedestrian detection has become a fundamental task in video surveillance, the ability to detect and track a human provides great support for reducing trespassers in restricted areas. Typical pedestrian detection applications rely on a classifying algorithm, and some form of machine learning with training/testing phases.

The training phase provides the program with examples of how the object it’s looking for may appear (a person in our case). These examples come in the form of a dataset, a collection of images representing the object to detect. Some datasets include what is known as negative examples, images which either represent something different or don’t include the object altogether. This opens up the classifier to different categories so it can overcome common appearances that interfere with its decision making. For example, including images of cars or animals would help the program differentiate pedestrians from other objects relevant to the environment.

There are a wide range of methods for detecting and classifying objects, many of which are based on the Viola-Jones object detection framework.

Pedestrian detection has seen progress in recent decades, the increase in publicly available datasets and frameworks has made computer vision more accessible.

What was your background preparation for the project? What similar systems did you assess? What was your motivation and interest in this project? Viola jones

## Analysis

Taking into account the problem and what you learned from the background work, what was your analysis of the problem? How did your analysis help to decompose the problem into the main tasks that you would undertake? Were there alternative approaches? Why did you choose one approach compared to the alternatives?

There should be a clear statement of the objectives of the work, which you will evaluate at the end of the work.

In most cases, the agreed objectives or requirements will be the result of a compromise between what would ideally have been produced and what was felt to be possible in the time available. A discussion of the process of arriving at the final list is usually appropriate.

As mentioned in the lectures, think about possible security issues for the project topic. Whilst these might not be relevant for all projects, do consider if there are relevant for your project. Where there are relevant security issues, discuss how they will this affect the work that you are doing. Carry forward this discussion into relevant areas for design, implementation and testing. ASSUMPTIONS, compare problems of surveillance to pedestrian detection solutions, BIG EFFORT TO REDUCE FALSE POSITIVES, false negatives not so bad but cause loss of trust. Racial profiling, False positives a breach of rights?.

## Process

You need to describe briefly the life cycle model or research method that you used. You do not need to write about all of the different process models that you are aware of. Focus on the process model that you have used. It is possible that you needed to adapt an existing process model to suit your project; clearly identify what you used and how you adapted it for your needs.

# Design

You should concentrate on the more important aspects of the design. It is essential that an overview is presented before going into detail. As well as describing the design adopted it must also explain what other designs were considered and why they were rejected.

The design should describe what you expected to do, and might also explain areas that you had to revise after some investigation.

Typically, for an object-oriented design, the discussion will focus on the choice of objects and classes and the allocation of methods to classes. The use made of reusable components should be described and their source referenced. Particularly important decisions concerning data structures usually affect the architecture of a system and so should be described here.

How much material you include on detailed design and implementation will depend very much on the nature of the project. It should not be padded out. Think about the significant aspects of your system. For example, describe the design of the user interface if it is a critical aspect of your system, or provide detail about methods and data structures that are not trivial. Do not spend time on long lists of trivial items and repetitive descriptions. If in doubt about what is appropriate, speak to your supervisor.

You should also identify any support tools that you used. You should discuss your choice of implementation tools - programming language, compilers, database management system, program development environment, etc.

Some example sub-sections may be as follows, but the specific sections are for you to define.

## Overall Architecture

## Detailed Design

### Even More Detail

## User Interface Design

## Other Relevant Sections

# Implementation

The implementation should look at any issues you encountered as you tried to implement your design. During the work, you might have found that elements of your design were unnecessary or overly complex; perhaps third party libraries were available that simplified some of the functions that you intended to implement. If things were easier in some areas, then how did you adapt your project to take account of your findings?

It is more likely that things were more complex than you first thought. In particular, were there any problems or difficulties that you found during implementation that you had to address? Did such problems simply delay you or were they more significant?

You can conclude this section by reviewing the end of the implementation stage against the planned requirements.

# Testing

Detailed descriptions of every test case are definitely not what is required here. What is important is to show that you adopted a sensible strategy that was, in principle, capable of testing the system adequately even if you did not have the time to test the system fully.

Provide information in the body of your report and the appendix to explain the testing that has been performed. How does this testing address the requirements and design for the project?

How comprehensive is the testing within the constraints of the project? Are you testing the normal working behaviour? Are you testing the exceptional behaviour, e.g. error conditions? Are you testing security issues if they are relevant for your project?

Have you tested your system on “real users”? For example, if your system is supposed to solve a problem for a business, then it would be appropriate to present your approach to involve the users in the testing process and to record the results that you obtained. Depending on the level of detail, it is likely that you would put any detailed results in an appendix.

The following sections indicate some areas you might include. Other sections may be more appropriate to your project.

## Overall Approach to Testing

## Automated Testing

### Unit Tests

### User Interface Testing

### Stress Testing

### Other Types of Testing

## Integration Testing

## User Testing

# Critical Evaluation

Examiners expect to find in your dissertation a section addressing such questions as:

* Were the requirements correctly identified?
* Were the design decisions correct?
* Could a more suitable set of tools have been chosen?
* How well did the software meet the needs of those who were expecting to use it?
* How well were any other project aims achieved?
* If you were starting again, what would you do differently?

Other questions can be addressed as appropriate for a project.

Such material is regarded as an important part of the dissertation; it should demonstrate that you are capable not only of carrying out a piece of work but also of thinking critically about how you did it and how you might have done it better. This is seen as an important part of an honours degree.

There will be good things and room for improvement with any project. As you write this section, identify and discuss the parts of the work that went well and also consider ways in which the work could be improved.

In the latter stages of the module, we will discuss the evaluation. That will probably be around week 9, although that differs each year.

# Appendices

The appendices are for additional content that is useful to support the discussion in the report. It is material that is not necessarily needed in the body of the report, but its inclusion in the appendices makes it easy to access.

For example, if you have developed a Design Specification document as part of a plan-driven approach for the project, then it would be appropriate to include that document as an appendix. In the body of your report you would highlight the most interesting aspects of the design, referring your reader to the full specification for further detail.

If you have taken an agile approach to developing the project, then you may be less likely to have developed a full requirements specification. Perhaps you use stories to keep track of the functionality and the ’future conversations’. It might not be relevant to include all of those in the body of your report. Instead, you might include those in an appendix.

There is a balance to be struck between what is relevant to include in the body of your report and whether additional supporting evidence is appropriate in the appendices. Speak to your supervisor or the module coordinator if you have questions about this.

* 1. Third-Party Code and Libraries

If you have made use of any third party code or software libraries, i.e. any code that you have not designed and written yourself, then you must include this appendix.

As has been said in lectures, it is acceptable and likely that you will make use of third-party code and software libraries. If third party code or libraries are used, your work will build on that to produce notable new work. The key requirement is that we understand what is your original work and what work is based on that of other people.

Therefore, you need to clearly state what you have used and where the original material can be found. Also, if you have made any changes to the original versions, you must explain what you have changed.

As an example, you might include a definition such as:

**Apache POI library** – The project has been used to read and write Microsoft Excel files (XLS) as part of the interaction with the client’s existing system for processing data. Version 3.10-FINAL was used. The library is open source and it is available from the Apache Software Foundation [5]. The library is released using the Apache License [6]. This library was used without modification.

* 1. Ethics Submission

This appendix includes a copy of the ethics submission for the project. After you have completed your Ethics submission, you will receive a PDF with a summary of the comments. That document should be embedded in this report, either as images, an embedded PDF or as copied text. The content should also include the Ethics Application Number that you receive.

* 1. Code Samples

This is an example appendix. Include as many appendices as you need. The appendices do not count towards the overall word count for the report.

For some projects, it might be relevant to include some code extracts in an appendix. You are not expected to put all of your code here - the correct place for all of your code is in the technical submission that is made in addition to the Final Report. However, if there are some notable aspects of the code that you discuss, including that in an appendix might be useful to make it easier for your readers to access.

As a general guide, if you are discussing short extracts of code then you are advised to include such code in the body of the report. If there is a longer extract that is relevant, then you might include it as shown in the following section.

Only include code in the appendix if that code is discussed and referred to in the body of the report.

Random Number Generator

The Bayes Durham Shuffle ensures that the pseudo random numbers used in the simulation are further shuffled, ensuring minimal correlation between subsequent random outputs.

// Some example code here…

# Annotated Bibliography

This final section should list all relevant resources that you have consulted in researching your project. Each reference should also include a brief annotation.

1. John Perdikaris, “*Physical Security and Environmental Protection*”, January 2014, Chapter 5.3 Surveillance and Counter Surveillance, page 147.
2. UK Government, “*Plug and Play Autonomous Sensors*”, 2015. Available: <https://www.gov.uk/government/news/plug-and-play-autonomous-sensors>. [Accessed 06 February 2017].

A press release discussing the development of new concepts in modular autonomous sensing. Provides details on some of the technologies used.

1. Gavin J.D Smith, “*Behind the Screens: Examining Constructions of Deviance and Informal Practices among CCTV Control Room Operators in the UK*”, pages 388-389, 2004. Available: <http://surveillance-and-society.org/articles2(2)/screens.pdf>. [Accessed 18 April 2017].
2. BBC, "*Missing Corrie Mckeague: Suffolk Police ‘search lacks resources’”,* November 2016. Available: <http://www.bbc.co.uk/news/uk-england-suffolk-38052508>. [Accessed 18 April 2017]  
     
   This paper…
3. W.H. Press et al. *Numerical recipes in C*. Cambridge University Press Cambridge, 1992.  
     
   This is my annotation. I can add in comments that are in **bold** and *italics*and then further content.
4. Various. Fail blog. <http://www.failblog.org/>, August 2011. Accessed August 2011.  
     
   This is my annotation. I should add in a description here.
5. Apache Software Foundation (2014) “*Apache POI - the Java API for Microsoft Documents*” (Online) Available at: <http://poi.apache.org> Accessed: 14th March 2014.
6. Apache Software Foundation (2004) “Apache License, Version 2.0” (Online) Available at: <http://www.apache.org/licenses/LICENSE-2.0> Accessed: 14th March 2014.
7. CREDIT CAVIAR/DAIMLER OPENCV