



# Applications of Computer Vision in Traffic System Data Collection

by

Callan Birch-Williams

A project submitted in partial fulfillment for the  
degree of M.Sc. Software Engineering (Web Development)

in the

School of Applied Computing  
Wales Institute of Science and Art

Supervisor: Tim Bashford

January 2024

# Declaration of Authorship

I, Callan Birch-Williams declare that I am the sole author of this Project; that all references cited have been consulted; that I have conducted all work of which this is a record, and that the finished work lies within the prescribed word limits.

This work has not previously been accepted as part of any other degree submission.

Signed:     **Callan Birch-Williams**  
.....

Date:     **29/01/2024**  
.....

# Form of Consent

I Callan Birch-Williams hereby consent that my Project, submitted in candidature for the M.Sc. Software Engineering (Web Development) degree, if successful, may be made available for inter-library loan or photocopying (subject to the law of copyright), and that the title and abstract may be made available to outside organisations.

Signed: Callan Birch-Williams

Date: 29/01/2024

UNIVERSITY OF WALES TRINITY SAINT DAVID

# *Abstract*

Wales Institute of Science and Art  
School of Applied Computing

M.Sc. Software Engineering (Web Development)

by Callan Birch-Williams

Data collection plays a key role in modern traffic engineering analysis. Despite this, the majority of data continues to be collected using manual counts or intrusive installations. A large amount of research into the benefits of computer vision applied to traffic data collection exists, however it is often narrowly focused. This project uses the trivial application of object detection, object tracking and OCR to detect, track and read licence plates for various regions of interest within a video. This facilitates vehicle reidentification, allowing for holistic traffic data to be collected for a specific traffic system. The results of this project demonstrate a high degree of accuracy for licence plate detection with a custom model achieving 99%, and proposes future research for OCR. An application that future researchers can use for data collection has been produced, in addition an example dataset for a traffic system has been produced.

# *Acknowledgements*

Foremost appreciation is extended towards Dr. Tim Bashford for undertaking the supervision of this project. His experience and the guidance he provided offered valuable direction and is gratefully acknowledged. A note of gratitude is owed to family and friends for not only assisting with practical application and labelling training data, but providing a constant source of motivation.

# Contents

|                                                    |             |
|----------------------------------------------------|-------------|
| <b>Declaration of Authorship</b>                   | <b>i</b>    |
| <b>Form of Consent</b>                             | <b>ii</b>   |
| <b>Abstract</b>                                    | <b>iii</b>  |
| <b>Acknowledgements</b>                            | <b>iv</b>   |
| <b>List of Figures</b>                             | <b>vii</b>  |
| <b>List of Tables</b>                              | <b>viii</b> |
| <b>List of Listings</b>                            | <b>ix</b>   |
| <b>1 Introduction</b>                              | <b>1</b>    |
| 1.1 Motivation . . . . .                           | 2           |
| 1.2 Relevance . . . . .                            | 2           |
| 1.3 Aims . . . . .                                 | 3           |
| <b>2 Literature Review</b>                         | <b>4</b>    |
| 2.1 Data Collection Technology . . . . .           | 4           |
| 2.1.1 Manual Counts . . . . .                      | 4           |
| 2.1.2 Intrusive Technology . . . . .               | 4           |
| 2.1.3 Non-Intrusive Technology . . . . .           | 5           |
| 2.1.4 Summary . . . . .                            | 7           |
| 2.2 Computer Vision Technology . . . . .           | 7           |
| 2.2.1 Image Processing . . . . .                   | 7           |
| 2.2.2 Machine Learning . . . . .                   | 8           |
| 2.2.3 Automatic Number Plate Recognition . . . . . | 9           |
| 2.3 Related Work . . . . .                         | 10          |
| 2.3.1 Point-to-point . . . . .                     | 10          |
| 2.3.2 Area Coverage . . . . .                      | 11          |

|          |                                      |           |
|----------|--------------------------------------|-----------|
| <b>3</b> | <b>Data Collection</b>               | <b>13</b> |
| 3.1      | Real-world Data . . . . .            | 13        |
| 3.1.1    | Location . . . . .                   | 13        |
| 3.1.1.1  | Location 1 . . . . .                 | 15        |
| 3.1.1.2  | Location 2 . . . . .                 | 15        |
| 3.1.2    | Video Capture . . . . .              | 16        |
| 3.1.3    | Training . . . . .                   | 17        |
| 3.2      | Synthetic Data . . . . .             | 19        |
| <b>4</b> | <b>Implementation</b>                | <b>22</b> |
| 4.1      | Configuration . . . . .              | 22        |
| 4.2      | Process . . . . .                    | 23        |
| 4.2.1    | Detection . . . . .                  | 25        |
| 4.2.2    | Tracking . . . . .                   | 27        |
| 4.2.3    | OCR . . . . .                        | 28        |
| 4.3      | Data Processing . . . . .            | 29        |
| 4.3.1    | Registration Normalisation . . . . . | 29        |
| 4.3.2    | Event Identification . . . . .       | 30        |
| 4.3.3    | Registration Lookup . . . . .        | 31        |
| 4.3.4    | Registration Stripping . . . . .     | 31        |
| 4.3.5    | Plate Interpolation . . . . .        | 33        |
| 4.3.6    | Visualisation . . . . .              | 34        |
| <b>5</b> | <b>Results and Recommendations</b>   | <b>38</b> |
| 5.1      | Detection . . . . .                  | 38        |
| 5.2      | OCR . . . . .                        | 40        |
| 5.3      | Real-world application . . . . .     | 41        |
| 5.4      | Performance . . . . .                | 42        |
| <b>A</b> | <b>Example JSON configuration</b>    | <b>44</b> |
| <b>B</b> | <b>Licence Plate Detection</b>       | <b>46</b> |
| <b>C</b> | <b>Registration Normalisation</b>    | <b>48</b> |
| <b>D</b> | <b>Combine events</b>                | <b>50</b> |
| <b>E</b> | <b>Plate Interpolation</b>           | <b>51</b> |
|          | <b>Bibliography</b>                  | <b>54</b> |

# List of Figures

|     |                                                  |    |
|-----|--------------------------------------------------|----|
| 3.1 | M4 J45 Ynysforgan interchange . . . . .          | 14 |
| 3.2 | Labelled satellite image of location 1 . . . . . | 15 |
| 3.3 | Labelled satellite image of location 2 . . . . . | 15 |
| 3.4 | Example detection training image . . . . .       | 18 |
| 3.5 | Example generated training images . . . . .      | 19 |
| 4.1 | Step-by-step detection process . . . . .         | 26 |
| 4.2 | Example visualisation frame . . . . .            | 35 |
| 5.1 | Location 1 route density . . . . .               | 42 |
| 5.2 | Average processing time distribution . . . . .   | 43 |

# List of Tables

|     |                                                 |    |
|-----|-------------------------------------------------|----|
| 2.1 | Data collection technology comparison . . . . . | 7  |
| 3.1 | Video capture devices summary . . . . .         | 17 |
| 4.1 | Additional information via VES . . . . .        | 31 |
| 5.1 | Custom YOLO dataset results . . . . .           | 39 |
| 5.2 | Public YOLO dataset results . . . . .           | 39 |

# List of Listings

|     |                                                        |    |
|-----|--------------------------------------------------------|----|
| 3.1 | Annotation for training image 3.4 . . . . .            | 18 |
| 3.2 | TRDG generator configuration . . . . .                 | 20 |
| 3.3 | Synthetic image generation output . . . . .            | 21 |
| 4.1 | Example JSON configuration preamble . . . . .          | 22 |
| 4.2 | Example JSON configuration detection and OCR . . . . . | 23 |
| 4.3 | Example JSON configuration ROI . . . . .               | 23 |
| 4.4 | Example event output CSV . . . . .                     | 30 |
| 4.5 | Anonymise registration hashing process . . . . .       | 33 |
| 4.6 | Frame visualisation process . . . . .                  | 37 |

# Chapter 1

## Introduction

Data collection plays a key role in modern traffic engineering analysis. At the end of 2022, there were 40.7 million licensed vehicles on the road in the UK, with this statistic increasing almost year-on-year since World War II [1]. As the world becomes increasingly dependent on data-driven insights, understanding traffic patterns has become critical for optimising not only road traffic systems but public transportation as a whole. Despite this, the majority of traffic data collection methods currently used either still rely on labor-intensive manual counts (Department for Transport statistics indicate that 96.25% of their counts are conducted manually [2]), or expensive enterprise solutions which are not practical in smaller applications or outright unavailable to smaller entities. Furthermore, the type of data collected is often limited to unrelated, single capture locations.

This project presents the modern application of computer vision as a suitable solution that bridges the gap between traditional manual data capture and modern technology, while providing access to unique data that is impossible to capture with existing proposed computer vision methods. This has been achieved by creating an application that can amalgamate data collected from multiple views, leveraging computer vision technology, to produce a holistic and comprehensive overview of vehicles journeys through a traffic system.

## 1.1 Motivation

Prior to the commencement of this project, there was a proposal to analyse detailed road traffic system data and algorithmically optimise traffic light timings for existing and theoretical (Both temporary or permanent) traffic systems. However, during the initial feasibility study into the availability of the necessary data in the public space, it became evident that this information is not readily available to independent researchers. Additionally, there was no guarantee that the relevant bodies possessed the relevant data or would show interest in cooperation.

This lead to the exploration of feasible data collection methodologies, from which the conclusion was drawn that the relevant data required to support the initial project could not be obtained. As a result of this, an interest arose in the development of a publicly available data collection methodology that would allow independent researchers to capture holistic traffic system data for free use.

## 1.2 Relevance

As traffic systems become increasingly complex, the ability for human enumerators to collect comprehensive data is becoming increasingly expensive and unrealistic. A wide variety of emerging technologies present various solutions applicable to different areas. When considering the benefits and drawbacks of different data collection technologies, it becomes apparent that computer vision provides a compelling solution to a wide range of applications for various reasons. It is non-intrusive and affordable for developing countries or research groups. Furthermore, continuous advancements in both computational power and artificial intelligence only stand to strengthen the argument for computer visions application in traffic data collection and analysis.

There is an abundance of research that presents computer vision as a suitable technology for vehicle data collection, however it is often narrowly focused. That is, proposed methodologies often consider a single point of interest as opposed to traffic systems as a whole. Conversely, other proposed applications often approach traffic data collection at a city or nationwide scale providing holistic, powerful estimations. However, this often comes with the cost of lacking focused and accurate domain specific information.

There are existing enterprise solutions that take advantage of computer vision technology to facilitate sophisticated data collection in a specific system as opposed to single point capture. In one example, the view of a four-way road junction can be partitioned and journey data through it can be collected with a reported 98% accuracy [3]. Another system uses the view of a multi-lane road system to isolate traffic events by lane and classification type (E.g. Truck southbound on lane one) [4]. These systems are a step in the right direction for the target domain and provide compelling evidence that the basis for this project is justified, while demonstrating that there is clear room for development in terms of consolidating data between multiple views, widening the number of traffic systems that can be analysed.

## 1.3 Aims

The primary aims of this project can be described as follows:

- A1 **Feasibility study:** This project aims to further present computer vision as a viable option in the area of traffic data capture. Particularly, to offer a solution for independent researchers interested in both conducting data collection or developing on the ideas presented.
- A2 **Application:** It is intended that future research and data capture can be conducted through the configuration, and use of, the application produced during this project.
- A3 **Dataset:** The production of a proof of concept dataset with data collected from a real-world environment using the application produced during this project.

# Chapter 2

## Literature Review

### 2.1 Data Collection Technology

#### 2.1.1 Manual Counts

Manual counts can be considered the traditional method of traffic data collection, where trained enumerators are present at a specific site to collect information. Despite technological advancements, manual data collection still accounted for 8,000 Department for Transport traffic statistics in 2022 [5]. Furthermore, human observation provides access to data that cannot be easily obtained through automatic traffic counters (E.g. vehicle occupancy, classification and pedestrians). The main shortcomings of manual counts are related to the human factor, namely; training and labour costs, data inaccuracy and inability to capture data outside a focused area.

#### 2.1.2 Intrusive Technology

In the context of traffic systems, intrusive technology can be considered any data collection methodology that requires direct contact with, or modification of, the road surface under observation.

- **Pneumatic road tubes:** Pneumatic road tubes are installed (Often temporarily) across an observation point. When a vehicle's tires pass over the

tube, air pressure closes a switch which produces an electrical signal. Data made available by this technology includes counts, spacing and classification by number of axles. Some studies suggest that the accuracy of pneumatic road tubes can be masked by positive and negative errors cancelling each other out. It is argued that the absolute error of a typical 15-minute count is approximately 10% [6]. These deviations are often associated with multi-axle vehicles, which also contribute to wear.

- **Inductive loops:** Inductive loop detectors are insulated, electrically conducted loops embedded in the road surface. A vehicle's presence within an inductive loop decreases the inductance of the loop. This is interpreted by a sensor and a pulse is sent to a controller signifying the presence of a vehicle [7].
- **Passive magnetic:** Magnetic sensors are passive devices that indicate the presence of a metallic object by detecting perturbations in the magnetic field [8], making them suitable for detecting both moving and stationary ferrous metal vehicles.

These related technologies are often installed together to facilitate vehicle classification through collection of length and wheelbase information [2][9]. While the technology itself is typically inexpensive, disruptive installation and maintenance (By nature of requiring partial or complete road closure) results in significant overhead. While existing systems are likely to remain in place for a long time, it is argued that the reliability and accuracy of these technologies are being made increasingly redundant by modern non-intrusive methods [10].

### 2.1.3 Non-Intrusive Technology

Non-intrusive technology for traffic data collection refers to any technology that does not necessitate direct modification of the road surface, or interact directly with vehicles under observation. This is particularly relevant in areas where road closure for cutting and boring is undesirable, such as bridges and tunnels or areas with high throughput. Additionally, non-intrusive measures provide an avenue for researchers to collect traffic data without relying on manual counts, as traditional intrusive methods are bound by permission to modify the area of interest. They

are typically much easier to remove from a location once sufficient data collection has been conducted.

- **Infrared:** Infrared sensors are categorised as either active or passive. Active sensors both emit and detect infrared light whereas passive sensors only detect infrared radiation. In the context of vehicle data collection, active sensor systems emit infrared light from a transmitter to a receiver perpendicular to the flow of traffic. Research indicates that active infrared sensors achieve count, speed and classification accuracy of over 90% [11], however distinction between multi-axle vehicles and two-axle vehicles in varying traffic conditions requires correction factors.
- **Microwave radar:** Microwave radar vehicle detection and classification systems extract speed and shape information from vehicles using the doppler signature they generate [12]. Further research applies Fast Fourier Transforms (FFTs) to enhance traffic detection [13].
- **Ultrasonic:** Ultrasonic vehicle detection sensors are commonly mounted above the road surface at a fixed distance. Ultrasonic waves are used to detect a vehicles presence based on the time between emission and reception of the reflected wave. Some research shows that presence expressed as a percentage over a period of time results in ambiguity between vehicle count and vehicle speed [14], which is nonetheless a metric of congestion [15].
- **Passive acoustic:** Passive acoustic sensors are placed adjacent to the flow of traffic and are used to capture the sounds emitted by vehicles. Audio data is analysed using signal processing algorithms, producing traffic data such as count, volume and speed detection. Specific sound signatures can be associated with various vehicle classifications based on audible characteristics like engine tone and tyre noise, however some research indicates that classification accuracy is inadequate in comparison to the accuracy of other data it produces [16]. Other areas of research consider the extraction of large scale, real-time traffic data using an array of sensors [17] and incident detection and response based on abnormal sound signatures [18].
- **Computer vision:** Computer vision technology makes use of cameras, typically mounted overhead or with a good view of the traffic system under observation (Though exploratory research has been conducted into aerial

vehicle data collection). Recorded or real-time imagery is then processed to extrapolate any type of vehicle data required, subject to the accuracy of the techniques employed.

### 2.1.4 Summary

Figure 2.1 summarises the characteristics of each data collection technology discussed in this section. It should be noted that combinations of these technologies, or arrangement of multiple devices, can provide data not specified here. Additionally, other research may suggest that certain applications of these technologies may allow differing data capture.

|               |                      | Count | Speed | Classification | Occupancy | Presence |
|---------------|----------------------|-------|-------|----------------|-----------|----------|
| Intrusive     | Pneumatic road tubes | ✓     | ✓     | ✓              | ✗         | ✗        |
|               | Inductive loops      | ✓     | ✗     | ✓              | ✓         | ✓        |
|               | Passive magnetic     | ✓     | ✗     | ✗              | ✓         | ✓        |
| Non-Intrusive | Infrared             | ✓     | ✓     | ✓              | ✓         | ✓        |
|               | Microwave radar      | ✓     | ✓     | ✓              | ✓         | ✓        |
|               | Ultrasonic           | ✓     | ✓     | ✗              | ✗         | ✓        |
|               | Passive acoustic     | ✓     | ✓     | ✓              | ✓         | ✓        |
|               | Computer vision      | ✓     | ✓     | ✓              | ✓         | ✓        |

TABLE 2.1: Data collection technology comparison

## 2.2 Computer Vision Technology

### 2.2.1 Image Processing

There are various traditional image processing methodologies that are highly applicable to the process of vehicle counting. These commonly fall within one of two categories; motion-based and appearance-based techniques. Motion identification techniques rely on movement to distinguish vehicles from the surrounding environment [19]. On the other hand, appearance-based techniques use vehicle characteristics (Shape, colour, etc.) to isolate vehicles from their backgrounds.

Temporal frame differencing computes the pixel-difference between two consecutive frames to isolate some moving foreground. To suppress noise from the output an empirical threshold is used. Two of the major drawbacks of this method are foreground aperture and ghosting, caused by parameters such as vehicle speed and

camera frame rate [20]. Improved results can be achieved by considering three frames for computation [21][22], where two different outputs are taken; frames one and two, and frames two and three. These outputs are then combined through logical conjunction.

In locations where the background content is unlikely to change, background subtraction [23] can be used to isolate the foreground by performing a subtraction between the current frame and a background model. In practical traffic applications (Where pedestrians, lighting, environmental changes etc. produce noise) these models are often the result of background reconstruction algorithms, which use various instances of the actual background to create a reliable model [24][25]. While publications in the study area focus on moving vehicles, an argument could be made for the application of background subtraction in stationary environments such as vehicle parking spaces.

Optical flow is an image processing technique used to capture image motion. A series of frames are analysed and velocity for points within the frames are calculated. This facilitates the estimation of where points could be in the next sequence of frames [26].

Unlike motion-based techniques, appearance-based techniques are capable of detecting stationary vehicles. Scale Invariant Feature Transformation (SIFT) [27][28], Speeded-Up Robust Features (SURF) [29][30], and Histogram of Oriented Gradients (HOG) [31] are image processing techniques commonly referenced in relation to vehicle detection.

### **2.2.2 Machine Learning**

Modern publications in the interest of vehicle detection typically focus on the emerging applications of machine-learning or deep-learning techniques. Convolutional Neural Networks (CNNs) provide a scalable approach to image classification [32], though require a large amount of labelled data to be trained effectively. Region-based CNNs (R-CNNs) build upon the classification properties of CNNs by implementing a selective search algorithm to propose regions of interest for a CNN to classify, facilitating object detection. Conventional R-CNNs are considered computationally expensive in the area of real-time vehicle detection and as a result faster R-CNN (A technique where a shared convolutional layer is used

to process all regions together) is often used [33][34]. Kernelized Correlation Filter (KCF), an object tracking framework, is also employed in high-speed vehicle tracking applications [35] [36]. You Only Look Once (YOLO) is one of the most popular object detection algorithms and is used extensively throughout research related to real-time vehicle detection [37] [38] [39].

### 2.2.3 Automatic Number Plate Recognition

Automatic Number Plate Recognition (ANPR) is an image processing technology used to isolate vehicles within an image and extract the content of their number plates into a machine-readable format (I.e. plain text). ANPR is comprised of two main image processing steps, plate localisation and Optical Character Recognition (OCR). ANPR deployed in predictable applications (That is, where lighting and number plate variance is expected to be minor) can use traditional image processing techniques. However, more advanced ANPR systems often employ sophisticated machine learning algorithms to localise the number plate [40]. Once the plate has been isolated, an OCR engine is responsible for extracting the characters. Image thresholding is used to separate visual artefacts from the isolated image. Individual characters are then evaluated using either their geometric properties (Feature detection) or a recognition model, based on a dataset of characters (Pattern recognition), to find the probabilistic closest match [41]. Typically, OCR requires an intermediate step to identify the style of writing used. However, in the UK there are enforced rules for number plates specifying letter spacing, size and style. Additionally, front and rear plates use fixed, separate colours to both facilitate thresholding and identifying the orientation of the vehicle.

ANPR has seen extensive use in modern applications. UK law enforcement highlights the use of ANPR technology for vehicle surveillance in the interest of detecting and deterring criminal activity, reporting an average of 60 million ANPR read events recorded daily [42]. Additional uses include automation of traffic systems such as tollgates and car parks.

When considering the applications of ANPR within traffic data collection, there are some immediate benefits that can be identified. As number plates are unique to each vehicle, identifying a vehicle at a different location becomes a matter of cross-referencing the number plate with known information. Furthermore, vehicle classification becomes a possibility through the use of systems such as the Vehicle

Enquiry Service (VES), a web service that provides vehicle details of a specified vehicle by their registration [43].

## 2.3 Related Work

The purpose of this section is to identify works closely related to the aims of this project. That is, proposed or existing technology that considers a traffic system at large as opposed to single-point data capture and analysis.

### 2.3.1 Point-to-point

There are various proposed applications that aim to detect a vehicle at different, but related points. The main benefit of these systems is access to additional data such as journey time. This is notably different to time spent derived from average vehicle speed on a road, as it can be applied to specific paths through complex traffic systems.

As discussed previously, intrusive technology is capable of achieving vehicle classification. Various publications develop this idea further by proposing that inductive loop detectors can facilitate vehicle re-identification [44]. In one study, vehicle signatures are matched from point A to point B using a maximum-a-posteriori classifier and found a 67% re-identification accuracy. Once calculated travel time was introduced as a feasibility metric, this figure reached 97.4% [45].

Automated Vehicle Identification (AVI) systems facilitate the immediate, automatic detection of specific vehicles. The underlying technology relies on vehicles being equipped with a transponder device that identifies itself when a signal is received from a reader device in a fixed location. Within the scope of traffic data collection (Specifically the use of probe vehicles to create a sample size of travel times), the applications of AVI are debated. While some publications proposed that the sample mean approaches the population mean as sample size increases, others draw attention to the fact that the sample mean will be biased in cases where a disproportionate number of probes report their results during a specific period of time or traffic condition. A comprehensive overview of these considerations are outlined by Hellinga and Fu [46]. The main drawback of AVI systems are

that they require the vehicle to be equipped with a device ahead of time, making it see little modern use in traffic data collection specifically. However, it has strong practical applications in industrial systems such as toll collection, rail and freight identification, gate automation etc. [47][48].

As discussed previously, AVI systems are often impractical for traffic data collection as it would require every vehicle under observation to be cooperative. Various researchers have proposed approaches that exploit the modern expectation that the majority of vehicles will contain at least one device with either Bluetooth or Wi-Fi compatibility [49]. The premise of the technology is based around pinging a device for its MAC address [50], by which it is uniquely identifiable at each observation location. One potential downside of such a system would be in the assumption that each vehicle contains one device that can be pinged, whereas in practice this could be multiple or none. Vehicle occupancy statistics for England between 2002 and 2018 [51] indicates an occupancy range of 1.55 to 1.59 per vehicle. Furthermore, head units in modern vehicles commonly use Bluetooth and Wi-Fi technology (Future points of interest could be; researching the number of compatible devices per vehicle, or if passenger devices could be differentiated from head units). In summary, the ability to uniquely identify vehicles is useful for determining time spent in a traffic system, whereas the accuracy of counts can be seen as unreliable due to the likelihood of retrieving MAC addresses, in addition to the uncertain number of devices.

### 2.3.2 Area Coverage

Global Positioning System (GPS) technology has become increasingly relevant in the world of traffic data collection. Traffic data is collected in real-time through the use of a GPS equipped device, typically a phone, that reports vehicle position, speed and direction to some central application [52]. Early research into GPS technology provided ample evidence to support its position in modern traffic data collection [53][54][55] and has since been adopted by major companies. For example, Google Traffic transitioned from traffic sensors and cameras on primary roads to a crowdsourcing GPS-based approach for the analysis of traffic flow [56][57]. Furthermore, GPS technology provides the ability to capture large scale traffic data for free use [58], something traditionally available to only government transportation agencies and private companies. At a surface level, GPS-based solutions

come with similar drawbacks to aforementioned methodologies. However, both the issue that driver cooperation is required, and that relations between vehicles and devices are indeterminate, have mitigating factors. As GPS-based systems are often distributed as consumer applications, there is an incentive for drivers to share information. A notable trend with modern GPS-based traffic technology is that it provides powerful estimations for an extremely broad area of interest, but not precise analytical data applicable to the optimisation of specific traffic systems.

Unmanned Aerial Vehicles (UAVs), or drones, offer a unique solution to traffic data collection. UAV technology facilitates area coverage and analysis at a rate incomparable to road-side methodologies. Little to no domain specific configuration is necessary, and they can be deployed at various locations for a period of time, producing a holistic overview of a traffic system. Research in this area is emerging [28][59][60], however many publications prefer to focus on the capabilities of UAVs for surveillance. A notable limitation of current UAV technology is flight time, with commercial camera drones ranging from around thirty minutes to a few hours [61], as well as limitations on areas of flight [62]. Suitability is also reduced in traffic systems where there are flyovers, tunnels or other interferes with aerial coverage. Another consideration to make is that UAVs require additional technology to capture traffic data, namely radar or computer vision, further consolidating the idea that computer vision technology is relevant to development in this area.

# Chapter 3

## Data Collection

### 3.1 Real-world Data

#### 3.1.1 Location

During the planning phase of this project, major consideration was given to the selection of real-world locations to deploy and test the designed application. The initial objective was to establish an observational network encompassing every entrance, exit and node of a significant traffic system. The types of locations suitable for such an application would include complex traffic systems like multiphase roundabouts or intersections with unique inflow and outflow points. These environments offer both high volume and sufficient complexity to not only demonstrate the feasibility of the application, but also to create a dataset that provides unique and practical traffic system data for analysis. Furthermore, it would demonstrate the advantages of computer vision given the impracticalities of traditional data capture methods in such a dense environment.

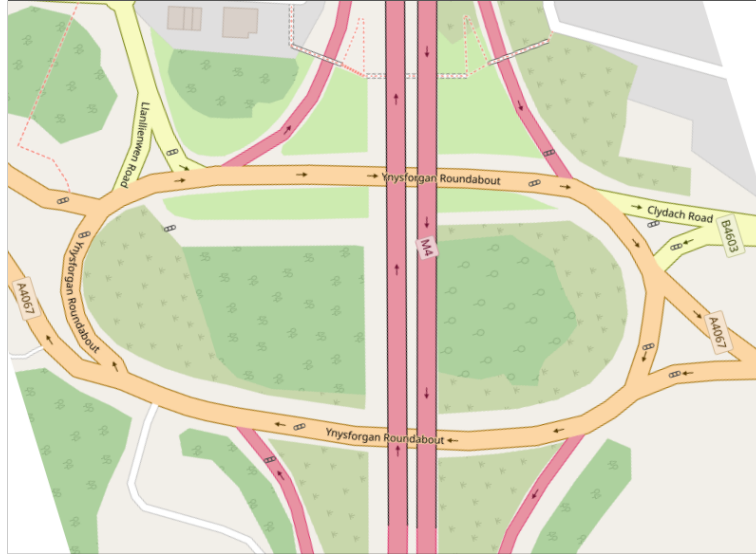


FIGURE 3.1: M4 J45 Ynysforgan interchange

Consider the traffic system depicted in Figure 3.1. There are multiple inflow and outflow points, as well as traffic lights staggered around the roundabout. Additionally, there is a bus lane that starts at one of the inflow points and then bypasses the roundabout, governed by its own traffic lights. Holistic traffic data for this system could provide a strong foundation for traffic pattern analysis, whether that be optimising the timings of traffic lights or evaluating the implications of future modifications or temporary disruptions. From a related technology standpoint, the effect of installing intrusive technology in such an environment would cause significant cost and interference.

While there are existing camera installations in locations similar to, and including Figure 3.1, the relevant bodies did not express interest in contributing data. As a result, the practical constraints of this project necessitated an independently fulfilled, more focused environment. Despite these limitations, locations were strategically chosen to maximise the variation of data collected. The two locations selected for this project allowed for the deployment of multiple cameras that provide a holistic view of an environment that can not be captured by a singular device. It is proposed that the smaller scale of this project does not diminish the significance or applicability of its findings.

### 3.1.1.1 Location 1



FIGURE 3.2: Labelled satellite image of location 1

The first location selected for this project was a three-way intersection, depicted in Figure 3.2, chosen for the additional complexity of one of the inflow points not being governed by the traffic light system but rather forking directly into one of the outflow points, controlled by broken white lines giving priority to other traffic approaching the same outflow point.

### 3.1.1.2 Location 2

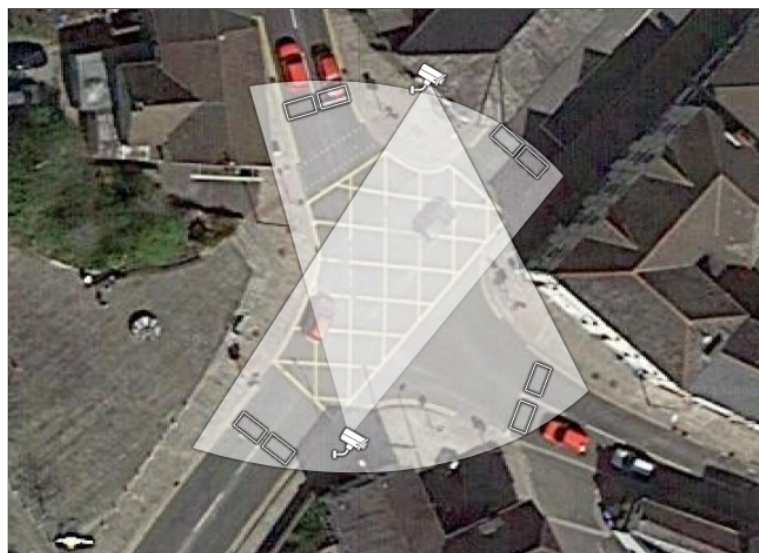


FIGURE 3.3: Labelled satellite image of location 2

The second location selected for this project was a four-way traffic light controlled intersection, shown in Figure 3.3. It was chosen not for the complexity of the traffic system itself, but rather to test the capabilities and limitations of the hardware and computer vision techniques employed. The increased distance between the cameras and the inflow and outflow points, coupled with the wide angle both cameras are required to cover, results in lower quality region of interest images due to the inability to make use of optical zoom.

### 3.1.2 Video Capture

In order to capture suitable video footage of road traffic at the chosen locations, considerations to the primary goal and constraints of the project's budget were made. In the absence of access to specialised equipment, the high-resolution capabilities of contemporary smartphones were leveraged. This approach not only insured the economic viability of this project (With the use of volunteered devices resulting in effectively no cost), but also provides compelling evidence for the use of accessible technology in the conduction of traffic system data collection. Furthermore, self-sufficient devices such as smartphones offer a level of flexibility and ease of setup that is not readily available with traditional fixed traffic recording equipment.

It should be noted however that while the use of smartphones presents a practical and cost-effective approach for this specific problem, it is important to acknowledge the inherent differences in capability when compared with specialised equipment. The smartphones used in this project, despite their high-resolution and appropriate frame rate, do not offer the same degree of environmental robustness as purpose built ANPR cameras. The absence of features such as infrared imaging results in less resilience to changes in environmental factors such as low light, reflections, and adverse weather conditions.

As a result of this, the scope and applicability of this project are suitably aligned with the operational constraints of the hardware used. The methodologies employed were designed and tested against stable video footage, with consistent overcast daytime conditions, from strategic positions intended to reduce foreground occlusion (Pedestrians, vehicles or other entities passing in front of the region of

interest). Consequently, the potential impact of the previously defined environmental changes was not within the purview of this project. This should be an important consideration when interpreting the presented findings.

To summarise, the hardware used to record video footage at both previously defined locations depicted in Figure 3.1.

|            |          | Device                | Resolution (px) | Framerate (Hz) | Bitrate (kbps) |
|------------|----------|-----------------------|-----------------|----------------|----------------|
| Location 1 | Camera 1 | Samsung Galaxy S20 FE | 3840x2160       | 59.99          | 41926          |
|            | Camera 2 | iPhone 14 Pro Max     | 3840x2160       | 60             | 53201          |
| Location 2 | Camera 1 | Samsung Galaxy S20 FE | 3840x2160       | 59.99          | 41926          |
|            | Camera 2 | iPhone 14 Pro Max     | 3840x2160       | 60             | 53201          |
|            | Camera 3 | Samsung Galaxy S23+   | 3840x2160       | 60.02          | 143970         |

TABLE 3.1: Video capture devices summary

### 3.1.3 Training

Real-world training data for this project was prepared using still images extracted from unprocessed video footage, captured by the cameras in situ at the two locations discussed previously.

Prior to manual annotation, these images were cropped to encompass a single vehicle and consequently a single licence plate. The dimensions of each image were intentionally varied and not strictly tied to the dimensions of the vehicle within it. The intended effect of this decision was to both simulate a wide range of potential dimensions for regions of interest, and to avoid overfitting the training model by only showing part of a vehicle.



FIGURE 3.4: Example detection training image

The cropped images were then labelled following the standard YOLO labelling structure [63]. A text file was created for each image, with each text file containing the annotations for its corresponding image file. For each object within an image, a new line is created in the text file. The licence plate is represented firstly by the object class, which is used to differentiate different types of objects. The position and size of the bounding box that encapsulates the licence plate is then defined by its position in the image, followed by the width and height. It should be noted that the position and size of the bounding box is defined in normalised  $xywh$  format (From 0 to 1).

```
0 0.244720 0.458671 0.111983 0.037718
```

LISTING 3.1: Annotation for training image 3.4

Consider the example label file in Listing 3.1 for the training image depicted in Figure 3.4. As there is only one object of interest within this problem space (The licence plate), the object class will always be represented by 0. Both the position  $(x,y)$  and the size in width and height  $(w,h)$  can then be derived by normalising the pixel values. For instance:

$$y_{\text{norm}} = \frac{Y}{\text{Image height}} = \frac{611}{1332} \approx 0.458671$$

In addition to labelling, each image file was named to represent the registration depicted on the licence plate. The purpose of this was to provide an efficient method for end-to-end testing of both the detection and character recognition components of the system.

## 3.2 Synthetic Data

Training an OCR model necessitates a substantial quantity of training images to achieve acceptable accuracy and robustness. Synthetic data generation offers a potential alternative to manual image annotation, especially for a domain-specific application like reading UK licence plate registrations.

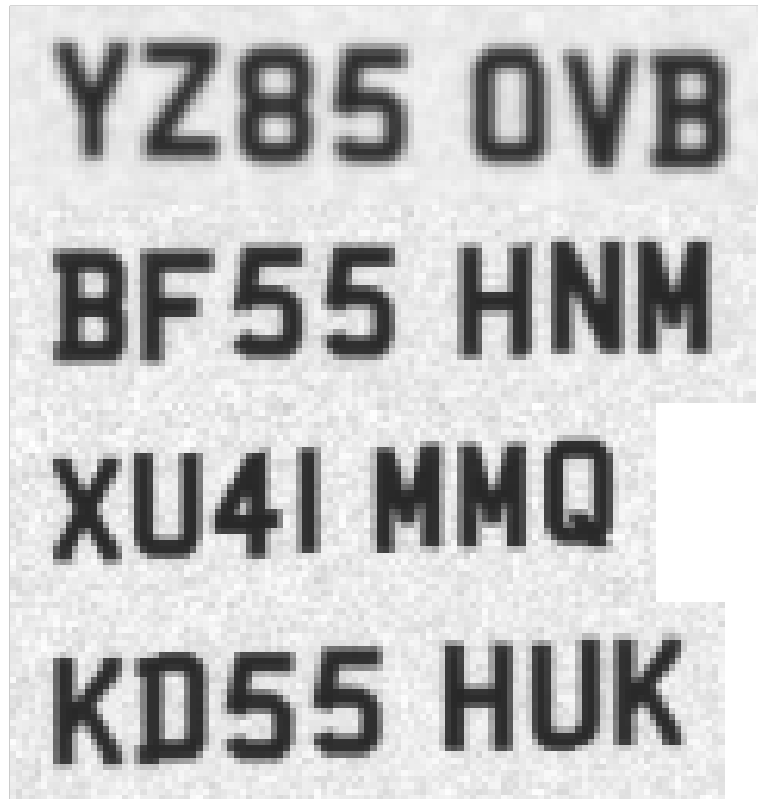


FIGURE 3.5: Example generated training images

For this project, synthetic images of UK licence plates are generated using Text Recognition Data Generator (TRDG), an open source synthetic data generator for text recognition [64]. The registrations within the generated images are randomly

generated to adhere to the current format of vehicle registration numbers in the UK. The Driver Vehicle and Licensing Agency (DVLA) introduced the current format in September 2001 and is two letters, two numbers, a space and three further letters [65]. All characters are required to be monospaced, have a stroke thickness of 14mm and be separated by 11mm (With the space before the last three letters being 33mm).

The typeface *CharlesWright*, and specifically *CharlesWrightBold* has an upper-case which conforms to the style of lettering used. As a result, when configuring TRDG this is the specific font used.

```
REGISTRATION_REGEX = r'^[A-Z]{2}[0-9]{2} [A-Z]{3}$'
FONT_PATH = 'resources/fonts/CharlesWrightBold.ttf'
OUTPUT_PATH = 'ocr/training/output'
COUNT = 25000

#Generate random valid UK current style registrations.
registrations = [rstr.xeger(REGISTRATION_REGEX) for _ in
    ↪ range(COUNT)]

#Configure TRDG
generator = GeneratorFromStrings(
    registrations,
    count=COUNT,
    fonts=[FONT_PATH],
    random_skew=True,
    random_blur=True,
    skewing_angle=2,
    blur=1
)
```

LISTING 3.2: TRDG generator configuration

Registrations are generated based on a regular expression which defines a correctly formatted current style registration string. Random skew and blur is introduced to each generated image, with the intent of increasing the robustness of the OCR

model. Images are then generated and written to an indexed image file. Additionally, *labels.txt* is created within which each row is a pair containing the path to the image, and the registration within it.

```
for image, label in generator:
    label = label.replace(' ', '')
    image.save(f'{OUTPUT_PATH}/{index}.png')
    labels_file.write(f'{index}.png\t{label}\n')
    index += 1
```

LISTING 3.3: Synthetic image generation output

The use of synthetic image generation in this project could have been developed further to generate synthetic training images for licence plate detection, by applying a random perspective transformation on the resulting image and superimposing it over a random background. The transformation would simulate the various angles at which a vehicle may be captured by a camera. Additionally, the random background could potentially aid with overfitting. Taking the aforementioned real-world training images as an example, the model may have depended on the consistent static background in each image. These theoretically generated images could also provide an avenue for end-to-end application testing, facilitating both plate detection and character recognition.

Another potential future use of these images, given some modification to their generation, could be to forgo an existing OCR model in favour of training an independent object detection model where each character on the generated licence plate is automatically annotated. Hand labelling real world images character by character would not only take a significant amount of time, but also require a large dataset to see enough character variation to train an accurate model. Synthetic data generation could therefore offer a powerful solution to this approach.

# Chapter 4

## Implementation

### 4.1 Configuration

The application produced with this project employs a JSON configuration system for reusability. A complete example JSON file for one of the videos produced in this project can be found in [Appendix A](#). The paths to the video to be processed and the desired output file are defined. The configuration then allows for specifying the start frame and its corresponding Unix timestamp. The purpose of this is to facilitate synchronisation between multiple cameras. This is a critical component of the application as a common, shared timeline between cameras is required in order to construct events from the amalgamated data.

```
"output": "scripts/generated/videos/raw/L1C1-custom-basic.csv",  
"video": "resources/videos/L1C1.mp4",  
"frame": 0,  
"unix": 1704369642,
```

LISTING 4.1: Example JSON configuration preamble

The detection parameters within the JSON configuration file outlines the detector to be used. An additional *params* key exists to allow for additional information to be passed to the specified detector. This is primarily used to specify the path to the YOLO model to be used. The OCR approach to be used is specified in a similar manner.

```
"detection": {
  "type": "yolo",
  "params": {
    "model": "scripts/detection/training/custom/runs
             detect/train/weights/best.pt"
  }
},
"ocr": {
  "type": "basic"
},
```

LISTING 4.2: Example JSON configuration detection and OCR

As discussed previously, each camera is cropped into regions of interest. These regions of interest are identified in the JSON configuration file, in  $x1y1x2y2$  bounding box format, and given a unique identifier.

```
"regions": [
  {
    "id": "outbound1camera1",
    "roi": {
      "x1": 550,
      "y1": 1080,
      "x2": 1050,
      "y2": 1450
    }
  },
]
```

LISTING 4.3: Example JSON configuration ROI

## 4.2 Process

The main application process is responsible for orchestrating the primary operations on a target video, namely plate detection, plate tracking and registration

recognition. A video configuration file, demonstrated previously, is read and the relevant detector and reader (OCR method) are created. The plate tracker for each region of interest is also instantiated at this point. Columns are written to the output CSV and are defined as follows:

***timestamp*** Unix timestamp at which the detection occurred.

***frame\_num*** Video frame in which the detection occurred.

***region\_id*** Region in which the detection occurred.

***plate\_score*** Confidence score for the plate detection.

***track\_id*** Unique identifier for an individually tracked plate.

***plate\_x1, plate\_y1, plate\_x2, plate\_y2*** Detected plate bounding box.

***registration*** Detected plate registration after processing.

***unprocessed\_registration*** Detected plate registration by OCR.

***registration\_score*** Confidence score for registration OCR.

***detection\_time*** Computational time required to detect the plate.

***track\_time*** Computational time required to calculate plate tracking.

***preprocess\_time*** Computational time required preprocess plate for OCR.

***ocr\_time*** Computational time required for plate OCR.

***postprocess\_time*** Computational time required to correct registration.

The input video is then processed frame by frame. Each frame is cropped into separate regions of interest which are processed sequentially. These regions are defined in the video configuration file by their unique identifier and relative position within the video frame. The first operation performed on a region is plate detection. The specified detector is invoked to try and detect a licence plate within the region. The resulting bounding box is then used to update the object tracking algorithm. In instances where no plate is identified within a region (Indicated by the absence of a bounding box), then that region is excluded from further processing for the given frame. Conversely, when a licence plate bounding box is

established, the image enclosed by the bounding box is segmented from the frame and enhanced to prepare it for OCR. The OCR process then attempts to identify text within the preprocessed image. Successful text extraction is followed by a post-processing phase to filter invalid registrations and attempt to rectify invalid characters. Finally, the real-world timestamp corresponding to the licence plate detection is derived from the following:

$$\text{timestamp} = \text{start\_timestamp} + \left( \left\lfloor \frac{\text{frame\_num} - \text{start\_frame}}{\text{framerate}} \right\rfloor \times 1000 \right)$$

The detected licence plate and registration is then written to the output CSV with the parameters outlined previously. This process is repeated for every region in every frame.

### 4.2.1 Detection

Two different approaches to plate detection were created for this project in the interest of comparison. Firstly, a detector was created that makes use of traditional image processing techniques to try and extract a plate from a region of interest. The image is converted to greyscale and passed through a bilateral filter, intended to remove noise from the image while preserving edges. This is important as the next operation performed is Canny edge detection, a multi-stage algorithm that extracts edges from an image. Contours within the resulting image (Found using the standard OpenCV *findContours* function [66]) are sorted by largest to smallest and excess contours are truncated to save processing time. Once a quadrilateral contour is found, it is assumed that this is the licence plate within the image. To assess if a contour is a quadrilateral, the points are reduced using the Douglas–Peucker algorithm, for which the approximation accuracy is dependent on the contour perimeter length. The complete process used for plate detection using image processing techniques is shown in Appendix B. Figure 4.1 depicts the image in each step of the detection process.



FIGURE 4.1: Step-by-step detection process

The second primary approach for plate detection uses the YOLO object detection algorithm. Two separate models were trained for comparison. The first of which is a large public dataset [67] comprised of 21174 different training images. The second uses independently sourced, manually labelled training images captured from the real-world test locations. Of which, 300 were used for training and a further 200 for validation and testing. The purpose of this is to facilitate the analysis of large generic datasets against small, domain specific datasets.

### 4.2.2 Tracking

It is reasonable to assume that the output of the OCR processing step will often produce misread registrations. One of the side effects of this is that there is no way to know if a new registration within a region is truly a new registration, or an existing registration that has been misread. There are two potential solutions to this problem, namely confidence frames and object tracking.

The confidence frames strategy dictates that a registration must be consistently recognised a predetermined number of times, within a certain timeframe to be counted as a registration. While this is a simple approach, it is likely that some licence plates that cannot be read by OCR will be missed. From a data capture perspective, it is more favourable to detect a licence plate and fail to read it, rather than not detect a licence plate at all. Furthermore, it is difficult to establish a universally applicable criterion for the number of confidence frames or the timeframe in which detection occurs, as these parameters are susceptible to change depending on the environment.

Object tracking addresses some shortcomings of the confidence frames approach, by providing the possibility to take the most likely registration for a tracked licence plate. Through project application testing it becomes apparent that the OCR of the registration is the primary constraint in data accuracy. It is therefore sensible to use a tracked licence plate as the source of truth rather than the data provided through OCR. Furthermore, in cases where the registration is not successfully read, the tracked licence plate is still recorded. As a result, the accuracy of vehicle count is dependent only on the accuracy of plate detection, rather than plate detection and registration recognition.

The object tracking algorithm used in this project is SORT, a simple implementation of a visual multiple object tracking framework based on basic data association and state estimation [68]. One important detail to note is that SORT does not handle occlusion or re-entering objects. While this may appear to be a major downside, vehicles are not expected to re-enter a region of interest (Or rather vehicles that eventually do should be considered unique) as they function as triggers in specific areas where vehicles move in a single direction. Foreground occlusion on the other hand does pose as a limiting factor, however the issue can be mitigated through planned camera placement. The simplicity of SORT also offers a performant tracking algorithm should this project be deployed in a real-time environment where computation time is important. A SORT object is instantiated for each region specified for the video being processed, and is updated with every new plate detection made for each region. The unique identifier given to each licence plate is added to the output CSV, resulting in the ability to group frame data by unique licence plate, an essential component for further data processing and analysis.

### 4.2.3 OCR

Once a licence plate has successfully been detected, OCR is responsible for extracting the registration. There are various Python OCR frameworks in this research space, with comparisons often drawn between Tesseract and EasyOCR. One comparative analysis of both in the context of ANPR found EasyOCR outperformed Tesseract with an accuracy of 95% compared to 90% [69]. This project makes use of EasyOCR for registration recognition [70].

In order to prepare the licence plate image for registration recognition, image processing is performed. A Gaussian blur is first employed to reduce noise. An adaptive thresholding operation is then performed to reduce the image to black or white. Following this morphological operations are applied, namely dilation to increase the stroke thickness of the letters after thresholding. An opening operation was also tested (By performing erosion before dialting) to try and remove blob noise from the licence plate image. It should be noted that the EasyOCR recognition pipeline includes a preprocessing step. It may be the case that better results are yielded by allowing the framework to perform the necessary operations on the original licence plate image.

Once EasyOCR extracts a registration from the licence plate, post-processing steps are conducted on the registration to attempt to produce a valid UK registration. As discussed previously, the current registration format is two letters, two numbers, a space and three further letters. The space (And any other whitespace in the registration) can be stripped. If the registration is not the expected seven characters, it can be assumed that it is not a valid registration. Conversely, if the registration is seven characters, it is passed through a lookup table that maps certain letters to numbers and visa versa, depending on the characters position in the registration. Finally, the registration is checked against a regular expression that enforces the current registration format. If this validation passes, the registration is considered a valid registration and written to the output CSV.

## 4.3 Data Processing

### 4.3.1 Registration Normalisation

When processing each video frame, it can be anticipated that for a single plate (That is, one that is identified as the same plate between frames by the SORT tracking algorithm), there will be discrepancies in the registration identified during the OCR step. This necessitates some process to try and assign a single registration to a single plate between frames. There were two proposed solutions, taking the registration with the highest recorded OCR confidence score, and taking the mode of each character, with the former being selected for this project. The justification for this being there is a non-zero possibility that there are multiple misread frames where the legibility of the registration is poor (Although this will be somewhat mitigated by the registration processing discussed earlier), which would result in a badly represented mode for some characters. Conversely, it may be reasonable to assume that there is at least one frame where the registration is read correctly by the OCR model. As a result, the decision was made to select the registration with the highest OCR confidence score.

Appendix C depicts the process used to achieve this. Two passes are made through the input CSV file. The first pass records, for each uniquely tracked plate (Given by the composite key of the *regionid* and the *trackid*), the highest confidence score and the associated registration. The second pass then every registration for each unique plate to the determined most likely registration.

### 4.3.2 Event Identification

The CSV file produced by the main video processing techniques and subsequent normalisation gives licence plate positions and information for every single frame. In order to produce meaningful interpretations of this data, it is necessary to transform the continuous stream of frame data to discrete, timestamped events. Specifically, for each region of interest, the two relevant events are both the arrival and departure of each distinct licence plate.

This is achieved by sorting the CSV file by *trackid* and then *framenum*, so that all frame data for each unique plate is grouped in sequential order. For each row in the sorted CSV file, if the *trackid* has changed since the previous row, it can be assumed that this is a new licence plate that is now being tracked. As a result, an *entered* event can be associated with this row. Similarly, if the *trackid* is not the same in the next row, we can assume that this is the last recorded frame for this specific licence plate, and an *exited* event can be associated with this row. A new column *event* is created at the start of the CSV file to hold this information. It should be noted that during this process frame-specific information such as computation time is truncated, leaving only the relevant columns in the output CSV. Finally, the CSV is sorted again by *timestamp*, such that events follow their real-world order to facilitate further data analysis.

```
event,timestamp,track_id,region_id,registration
entered,1704369642.0,1.0,inbound2camera1,51ff...
exited,1704369642.0,1.0,inbound2camera1,51ff...
entered,1704371642.0,5.0,inbound2camera1,57ad...
exited,1704372642.0,5.0,inbound2camera1,57ad...
```

LISTING 4.4: Example event output CSV

Listing 4.4 depicts an example output from event identification, with vehicle registrations stripped, a process which will be discussed further. Once each video has been processed and the resulting data has been converted to discrete events, the files can be amalgamated into a single event CSV representing the entire traffic system under observation. This process (Shown in Appendix D) simply concatenates the data in each file, then sorts the resulting rows by timestamp, as discussed previously. The data at this point is now ready for researcher interpretation, however there are further supplemental data operations that can be carried out.

### 4.3.3 Registration Lookup

In order to enrich the data collected with the methodologies outlined previously, additional vehicle information is retrieved from the DVLA database by querying the Vehicle Enquiry Service (VES). The information identified as relevant to this research is outlined in Table 4.1, as well as the associated attribute names in both the VES response and the resulting additional column in the output CSV file.

| Description         | VES Response Field       | CSV Output Column       |
|---------------------|--------------------------|-------------------------|
| Year of Manufacture | <i>yearOfManufacture</i> | <i>manufacture_year</i> |
| Make                | <i>make</i>              | <i>make</i>             |
| Colour              | <i>colour</i>            | <i>colour</i>           |
| Fuel Type           | <i>fuelType</i>          | <i>fuel_type</i>        |
| Engine Capacity     | <i>engineCapacity</i>    | <i>engine_capacity</i>  |
| CO2 Emissions       | <i>co2Emissions</i>      | <i>co2_emissions</i>    |
| Wheel Plan          | <i>wheelplan</i>         | <i>wheelplan</i>        |

TABLE 4.1: Additional information via VES

When it comes to querying VES efficiently, there are two major advantages derived from the previously discussed OCR and normalisation components of this project. As licence plate registrations at this stage have been resolved and validated to adhere to the current UK vehicle registration format, queries to VES are more likely to yield successful results. Additionally, erroneous registrations are removed, leaving an empty registration. This allows the lookup process to skip rows in the input CSV where there is no registration present. Nonetheless, there are additional considerations that can be made to reduce the number of calls made to VES. As registrations are likely to appear multiple times (Event pairs recorded at both traffic system entry and exit, as well as re-entry), the result for each specific registration can be cached, so that only a single query is required for each unique registration. It also stands to reason that performing vehicle registration queries after data has been amalgamated to a single file reduces the number of total calls for a location as a result of caching. An example

### 4.3.4 Registration Stripping

From both an ethical and legal standpoint, removing personal information from published data may be in a researchers interest. For example, the datasets produced in this project have had licence plate registrations replaced by a hashed

value. This removes personal information while still facilitating the ability to recognise traffic patterns. It is possible to reidentify the same vehicle, without knowing any identifiable information about that vehicle.

When selecting the approach for stripping a vehicle's registration from a dataset, there are a few considerations that should be made. As within the context of this project there is no logical resolution of a hashing collision, the chosen hashing function should have a very low chance of collision. Cryptographic hash functions such as SHA-256 and MD5 are designed to be collision resistant and provide a compelling solution. Additionally, adding a salt to each registration before hashing can provide an extra layer of security. This is particularly significant when considering both the number of existing UK registrations at the end of 2023 being 41.2 million [71], and the use of structured registration formats, leaves a reasonably brute-forceable input range. This raises the further question of should the registration stripping operation should be deterministic between runs (Should the salt value used be maintained). While being able to conduct further research without invalidating a previous dataset is a powerful premise, consideration to the aforementioned factors should be given. This decision is ultimately best evaluated on a per-research basis.

For the simplicity of this project, a cryptographically strong random salt is generated and combined with each registration in the input CSV before being hashed using SHA-256. The salt is not maintained for further use, effectively permanently anonymising the original registration data. The hexadecimal representation of the hashed registration replaces the registration in the output CSV. This process is depicted in Listing 4.5.

```
def anonymise(input, output):  
  
    #Generate a random, unsaved salt. Future researchers may want  
    ↪ to reuse the same salt for a project.  
    salt = secrets.token_bytes(16)  
  
    with (open(input, mode='r', newline='') as file_in,  
          open(output, mode='w', newline='') as file_out):  
        reader = csv.DictReader(file_in)  
        writer = csv.DictWriter(file_out, fieldnames =  
                                ↪ reader.fieldnames)  
  
        writer.writeheader()  
        for row in reader:  
  
            #Hash the salt and registration.  
            hash = hashlib.sha256()  
            hash.update(salt + row['registration'].encode())  
  
            #Update the registration to the hexadecimal  
            ↪ representation of its hashed value.  
            row['registration'] = hash.hexdigest()  
            writer.writerow(row)
```

LISTING 4.5: Anonymise registration hashing process

### 4.3.5 Plate Interpolation

When considering data visualisation and other applications where information continuity takes precedence over unaltered data, there exists a need to generate and insert suitable substitutions in the absence of genuine data. In the context of this project, this occurs in the form of missing frames for a unique licence plate. It is reasonable to anticipate that during the processing of a video, the plate detection method employed may not identify the licence plate within the region of interest for every frame in which it is visible. As a result, there may be one or

many missing frames of information for that licence plate. The primary interest for missing frame generation in this project, is for visualisation. While previous frame data for a licence plate can be used for details such as registration (Under the assumption that the data has been normalised), the bounding box for the licence plate will have changed between frames as a result of vehicle movement. It is therefore required to interpolate the position of the licence plate based on known information.

To achieve this, the input CSV is first organised into a dictionary, with the indices being distinct licence plates (Given by the composite key of *regionid* and *trackid*) and the values being lists of all frames associated with that plate. By doing this, it is possible to sequentially step through the frames for a specific licence plate. If a frame is missing then it should be interpolated.

This is achieved by determining the previous and next frames, which may be multiple frames away from the current frame. Every frame between the previous frame and the next frame is interpolated. This is achieved by linearly spacing each coordinate over the number of missing frames. It may be more suitable in different applications to take more than the immediate previous and next frames in order to interpolate the coordinates against a overarching trend, to account for vehicle acceleration. However, the accuracy for the use cases within this project are sufficient due to the minimal amount of time a moving vehicle remains within a region of interest, diminishing the effects of vehicle acceleration.

The resulting CSV file is then re-sorted by *framenum* once interpolation is complete. A new column is appended to the data which specifies which frames are interpolated. The implementation of the plate interpolation used in this project is listed in Appendix [E](#).

### 4.3.6 Visualisation

Data visualisation is a powerful tool not only for the presentation of project functionality, but also facilitating iterative development of an application. It provides a medium through which obscure trends or edge-cases can be identified via manual observation. In the context of this project, visualisation describes the annotation of input videos with processing information such as bounding boxes and extracted data. This allows for fine-tuning of regions of interest through observation of issues

such as foreground obfuscation or bad placement resulting in plates missing the bounding box. These are important details that cannot be interpreted through data alone. Additionally, issues with specific components of the processing life-cycle are easier to identify through visualisation. For example, if the detection model is falsely identifying regions as a licence plate, this may incorrectly reflect in data as the OCR model performing poorly. Figure 4.2 depicts a cropped frame from a generated visualisation.

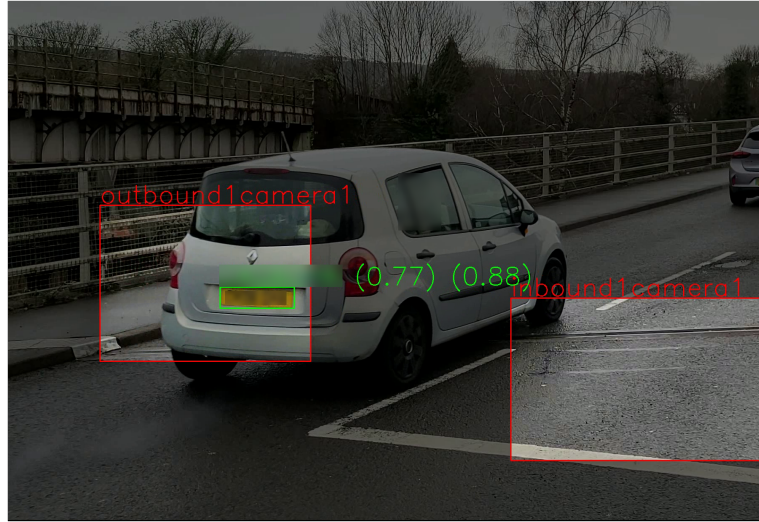


FIGURE 4.2: Example visualisation frame

In order to create an annotated output video, both the resulting CSV file selected for visualisation and the original video configuration file are required. This is due to the fact that the configuration file holds the region bounding box information, coupled with the specific region *id*. Additionally, the video configuration file holds the starting frame number, which is essential to synchronising output data with the original video. Once the required information is read from both files, the input video frames are sequentially read, modified and then written to the specified output video file.

Listing 4.6 outlines the visualisation process for a single frame. In order to emphasise the regions of interest, the original frame is reduced to half opacity. This is done to create a background layer upon which elements can be added. Firstly, each region of interest is given full opacity and a bounding box, based on the coordinates specified in the video configuration file. The regions *id* is also annotated above the bounding box. For each plate that exists for the current frame, the

bounding box is drawn and annotated with its associated registration. Additionally, the confidence score for both the plate detection and OCR are appended to the annotation (It should be noted that the OCR score is derived from the original detection, irrespective of post-processing or normalisation). There is no need to replace the dimmed region with the region defined by the plate bounding box as the plate will always fall within a region of interest.

```

#Half the alpha of the frame.
dimmed = cv2.convertScaleAbs(frame, alpha=0.5)

#For every region.
for id, (x1, y1, x2, y2) in regions.items():

    #Replace the dimmed region with the original frame region.
    dimmed[y1:y2, x1:x2] = frame[y1:y2, x1:x2]

    #Add a visual bounding box and region name.
    cv2.rectangle(dimmed, (x1, y1), (x2, y2), (0, 0, 255), 2)
    cv2.putText(dimmed, id, (x1,y1 - 5), cv2.FONT_HERSHEY_SIMPLEX,
        ↪ 2, (0,0,255), 2)

#For each plates that exists for the current frame.
if frame_num in plates:
    for plate in plates[frame_num]:
        registration, plate_score, registration_score, x1, y1, x2,
        ↪ y2 = plate

        #Add a visual bonding box and plate registration.
        cv2.rectangle(dimmed, (x1, y1), (x2, y2), (0, 255, 0), 2)
        cv2.putText(dimmed, f"{registration}
        ↪ ({float(plate_score):.2f})
        ↪ ({float(registration_score):.2f})",
            (x1,y1 - 5), cv2.FONT_HERSHEY_SIMPLEX, 2,
            ↪ (0,255,0), 2)

#Write frame.
writer.write(dimmed)
frame_num += 1

```

LISTING 4.6: Frame visualisation process

# Chapter 5

## Results and Recommendations

To summarise this projects outcomes and propose areas for future work, it can be concluded that the applications licence plate detection capabilities yield reliable performance. Conversely, the OCR techniques employed to recognise vehicle registrations demonstrate less effective results, impacting the proficiency of the application when it comes to vehicle reidentification.

### 5.1 Detection

Through testing both licence plate detection techniques, it is evident that the use of object detection systems such as YOLO yields better results. This is supported by the number of detections made by both the custom model and the one trained on a public dataset compared to the image processing approach. The image processing approach correctly detected a licence plate in 62/100 annotated test images. It should be noted that this figure was collected through manual testing, it may be of future researchers interest to develop a system through which more specific metrics are captured, as in the case of the YOLO models. The difference in detections clearly demonstrates the advantage the trained models have. The primary reason image processing produces fewer detections can be associated with its vulnerability to slight changes in environment. Despite the research being conducted in a particular area under the same conditions, discrepancies in lighting, licence plate position etc. result in the wrong area being selected.

| Class | Images | Instances | Box(P) | R    | mAP50 | mAP50-95 |
|-------|--------|-----------|--------|------|-------|----------|
| all   | 100    | 99        | 0.99   | 0.99 | 0.995 | 0.818    |

TABLE 5.1: Custom YOLO dataset results

| Class | Images | Instances | Box(P) | R    | mAP50 | mAP50-95 |
|-------|--------|-----------|--------|------|-------|----------|
| all   | 100    | 99        | 0.98   | 0.97 | 0.982 | 0.702    |

TABLE 5.2: Public YOLO dataset results

The training results for both the custom and public datasets indicate that both models were able to detect licence plates with a very high degree of accuracy. Interestingly the custom model, despite being trained on only 300 images, was able to outperform the public model, with a significantly higher mAP50-95 metric and a higher box precision (Albeit detecting one more image successfully). This may partially be due to the fact that the test data and training data was annotated by the same researcher, whereas the public dataset may have slightly different annotation styles for licence plates, namely how close the bounding box is drawn to the actual licence plate. Even if this is the case, it has been demonstrated that a low number of images is sufficient for training a model given a predicable environment. It would be interesting for further analysis to be conducted to determine how much the performance of the custom model degrades when deployed on other traffic systems.

Based on the results outlined previously, it is recommended that future researchers consider a machine learning approach to plate detection, give its superior accuracy and the fact that it is less vulnerable to smaller changes that hinder traditional image processing techniques. Further consideration should be given to the benefits and drawbacks of a custom domain specific model and a generic public model with greater coverage.

One of the shortcomings of the YOLO object detection approach is the detection of truncated plates. When a licence plate begins to exit a region of interest, it continues to be detected as a licence plate. This is partially responsible for some OCR failures as the entire registration is no longer in frame. It is proposed that future research could be conducted into plate classification in the interest of only

detecting complete licence plates. One such solution would be to post-process the detected licence plate for an expected aspect ratio, although this may not be robust for all possible licence plates (For example, square motorcycle or import vehicle licence plates).

While there are less available metrics to evaluate the performance of the SORT tracking algorithm, the data visualisation component of this project provides a medium through which it can be verified. While it performs correctly under observation, there were instances of failure, namely misinterpreting the same licence plate as a new licence plate. These occurrences are primarily associated with foreground obfuscation, examples of which are discussed with the real-world results. In order to generate more metrics for tracking performance, the authors of SORT recommend Local Metrics for Multi-Object Tracking [72] as a framework to better measure and understand tracking performance, in a manner less biased towards the performance of the object detection itself.

## 5.2 OCR

When considering the overall performance of this project, the accuracy of OCR is the area in which primarily requires further development.

One important factor to recognise is that some registrations that may have been successfully read by the OCR process were lost due to the trivialisation of UK licence plate registration formats. This project focused on the current style registrations, while prefix, suffix, dateless, etc. plates are still valid within the UK. Future researchers may consider expanding upon the methodologies used to try and validate and correct the text produced by OCR. Despite this, there are cases in which valid registrations are clearly legible under human scrutinisation, where OCR fails to correctly identify the registration. The average confidence score for the default EasyOCR model was 0.44.

To identify the effectiveness of the post-processing operations performed on the captured, two figures have been produced. The percentage of exact matches is only 14%, strongly indicating that the post-processing operations on the registration are justified. Additionally the similarity, derived from the Jaro-Winkler similarity metric, is only 29%.

This project was unsuccessful in training a custom EasyOCR model with the synthetic data generated. Future work could be undertaken to utilise the synthetic data generation techniques outlined in this project. Furthermore, experimentation with alternative OCR frameworks may be an avenue future researchers want to explore. An interesting alternative to existing OCR frameworks may be to research the feasibility of performing custom OCR, by training an object detection model such as YOLO to classify registration letters within an image. The classified letters can then be ordered by their position in the licence plate in order to extract the registration. Due to the accuracy shown previously this may yield more robust performance than generic OCR solutions which are not tailored to the problem space.

It could be argued that as the licence plate recognition yields a high degree of accuracy, the data that is successfully captured through OCR can be proportionally scaled up to produce an approximation that can be used for traffic flow insights. One potential drawback of this may be that OCR may perform better in some regions than others, resulting in skewed data. Further research into the details of this approach may be interesting to undertake.

If future researchers decide to move away from EasyOCR as the platform for registration recognition, consideration should be given to implementing skew correction as part of the preprocessing steps before OCR. EasyOCR performs skew correction during its preprocessing pipeline. Skew correction is an important step in normalising the input for character recognition. This is even more significant in the application of registration recognition as licence plates can be expected to be rotated due to the turning of vehicles in a traffic system. Furthermore, the optimal position of cameras in real-world applications is overhead to reduce foreground occlusion, contributing to rotation in another axis.

### 5.3 Real-world application

From the two locations researched during this project, a dataset has been created for the first location. Through video and data analysis it became apparent that the hardware used resulted in poor data collection for the second location from which few conclusions could be drawn. Nonetheless, it provided valuable insight into the technical limitations of the hardware used, and the methodologies employed.

For the processed location, 1691 unique vehicles were detected entering the traffic system over a 30-minute period, of which 859 have details held by the DVLA. This suggests that 50.8% of the registrations captured are correct. Figure 5.1 presents an example of the interpretations that can be made from the resulting data collection.

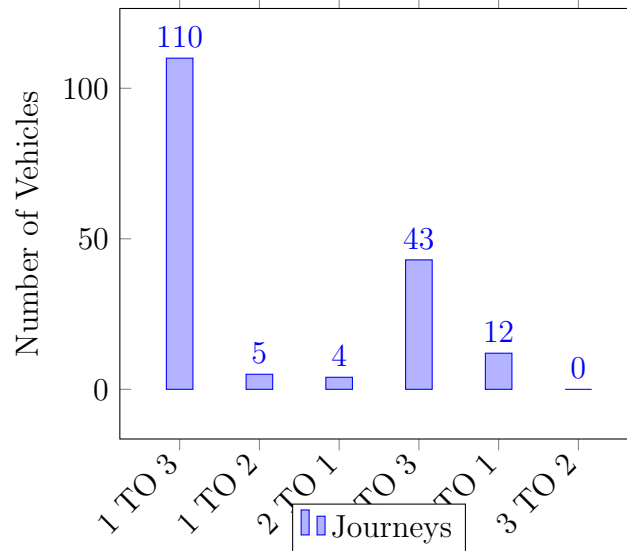


FIGURE 5.1: Location 1 route density

## 5.4 Performance

Performance metrics for each main processing step for this project were recorded for each camera in the first location. For every camera the computational time detection and OCR took were orders of magnitude greater than any other process. As such that is the data focused on. The video that took the longest to process licence plates was camera 2, which may be explained by the difference in region of interest size. The average licence plate detection time was 14.69ms. Interestingly, camera 2 also had the longest average time for OCR, at 23.01ms. It would be reasonable to expect the OCR time to be similar for every camera as licence plates are of similar pixel size. Further research into what effects performance may be valuable. The hardware used to process the videos used in this project is provided for reference as follows:

**Processor** AMD Ryzen 5 5600X 6-Core Processor 3.70 GHz

**Graphics** NVIDIA GeForce RTX 3080

**Memory** 32GB

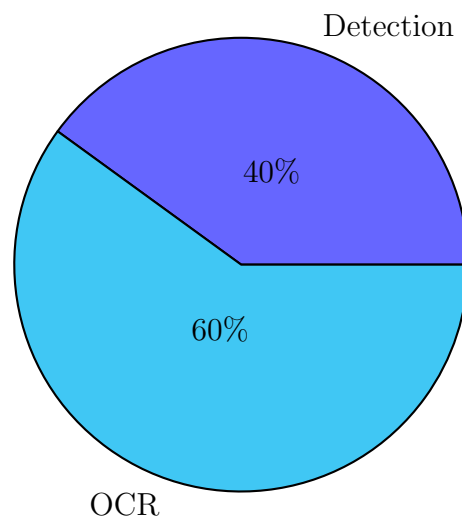


FIGURE 5.2: Average processing time distribution

# Appendix A

## Example JSON configuration

```
{
  "output": "generated/videos/raw/L1C1-custom-basic.csv",
  "video": "resources/videos/L1C1.mp4",
  "frame": 0,
  "unix": 1704369642,
  "detection": {
    "type": "yolo",
    "params": {
      "model": ".../best.pt"
    }
  },
  "ocr": {
    "type": "basic"
  },
  "regions": [
    {
      "id": "outbound1",
      "roi": {
        "x1": 550,
        "y1": 1080,
        "x2": 1050,
        "y2": 1450
      }
    },
    {
      "id": "inbound1",
      "roi": {
        "x1": 1525,
        "y1": 1300,
        "x2": 2365,
        "y2": 1685
      }
    },
    {
      "id": "inbound2",
      "roi": {
        "x1": 3030,
        "y1": 1250,
        "x2": 3650,
        "y2": 1650
      }
    }
  ]
}
```

## Appendix B

### Licence Plate Detection

```
def detect(self, image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    #Find edges within the image.
    bfilter = cv2.bilateralFilter(gray, 11, 17, 17)
    edged = cv2.Canny(bfilter, 30, 200)

    #Find contours within the image, sorted by size.
    keypoints = cv2.findContours(edged.copy(), cv2.RETR_TREE,
        ↪ cv2.CHAIN_APPROX_SIMPLE)
    contours = imutils.grab_contours(keypoints)
    contours = sorted(contours, key=cv2.contourArea,
        ↪ reverse=True)[:10]

    roi = None
    for contour in contours:
        #Determine if the contour is a quadrilateral, using perimeter
        ↪ size as an accuracy metric.
        perimeter = cv2.arcLength(contour, True)
        approx = cv2.approxPolyDP(contour, 0.015 * perimeter, True)
        if len(approx) == 4:
            roi = approx
            break

    #If no suitable contour is found return none.
    if roi is None:
        return None, -1, None

    #Take the bounding box of the contour from the original image.
    mask = np.zeros(gray.shape, np.uint8)
    cv2.drawContours(mask, [roi], 0, 255, -1)

    (x, y) = np.where(mask==255)
    (x1, y1) = (np.min(x), np.min(y))
    (x2, y2) = (np.max(x), np.max(y))
    return gray[x1:x2 + 1, y1:y2 + 1], -1, [x1, y1, x2, y2]
```

# Appendix C

## Registration Normalisation

```

def normalise(input, output):
    #Read csv
    data = []
    with open(input, mode='r', newline='') as file:
        reader = csv.DictReader(file)
        for row in reader:
            data.append(row)

    best_registrations = {}
    for row in data:
        #A uniquely tracked plate is defined by the composite key
        → of the region and the tracking id.
        #This is to avoid a potential collision if a different
        → region uses the same tracking id for a plate.
        plate = (row['region_id'], row['track_id'])

        #Fetch the registration score, if the registration is
        → blank, the score should be graded below zero.
        #Blank registration will therefore only be used if there
        → are no alternatives.
        score = float(row['registration_score']) if
        → row['registration'] else -1

        # Update the best registration for this plate if the
        → current one has a higher score.
        if plate not in best_registrations or score >
        → best_registrations[plate]['score']:
            best_registrations[plate] = {
                'registration': row['registration'],
                'score': score
            }

    #Step through data and update each unique plate to its best
    → scoring registration.
    for row in data:
        plate = (row['region_id'], row['track_id'])
        if plate in best_registrations and
        → best_registrations[plate]['registration']:
            row['registration'] =
            → best_registrations[plate]['registration']

    #Write csv
    with open(output, mode='w', newline='') as file:
        writer = csv.DictWriter(file, fieldnames=data[0].keys())
        writer.writeheader()
        writer.writerows(data)

```

# Appendix D

## Combine events

```
def combine(inputs, output):
    combined_rows = []
    header = None

    for input in inputs:
        with open(input, 'r', newline='') as file:
            reader = csv.DictReader(file)

            #Fetch the header from the first file read.
            if header is None:
                header = reader.fieldnames

            #Read rows.
            for row in reader:
                combined_rows.append(row)

    #Sort rows by timestamp then track id.
    combined_rows.sort(key=lambda x: (float(x['timestamp']),
    ↪ float(x['track_id'])))

    #Write csv.
    with open(output, 'w', newline='') as file:
        writer = csv.DictWriter(file, fieldnames=header)
        writer.writeheader()
        for row in combined_rows:
            writer.writerow(row)
```

# Appendix E

## Plate Interpolation

```
def interpolate(input, output):

    #Read csv
    original_rows = []
    with open(input, mode='r', newline='') as file:
        reader = csv.DictReader(file)
        for row in reader:
            original_rows.append(row)

    #Make a dictionary entry for each unique plate (Given by
    ↪ composite key of region_id and track_id).
    plates = defaultdict(list)
    for row in original_rows:
        key = (row['region_id'], row['track_id'])
        plates[key].append(row)

    #For every unique plate.
    interpolated_rows = []
    for plate_rows in plates.values():

        #If there is more than one frame in which this plate
        ↪ exists.
        if len(plate_rows) > 1:

            #Mark existing frames as not interpolated.
            for row in plate_rows:
                row['interpolated'] = False

            #Extract existing frames.
            first_frame = int(plate_rows[0]['frame_num'])
            last_frame = int(plate_rows[-1]['frame_num'])
            existing_frames = set(int(row['frame_num']) for row in
            ↪ plate_rows)

            #Step frame by frame between the first and last frame
            ↪ for this plate.
            for frame_num in range(first_frame, last_frame + 1):

                #If this frame is missing then it should be
                ↪ interpolated.
                if frame_num not in existing_frames:

                    #Find the previous and next frames (E.g.
                    ↪ previous filters for only frames before
                    ↪ the current, then takes the highest).
                    prev_frame = max(filter(lambda x: x <
                    ↪ frame_num, existing_frames))
                    next_frame = min(filter(lambda x: x >
                    ↪ frame_num, existing_frames))
```

```

#Retrieves the row for the previous and next frames.
prev_row = next(row for row in plate_rows if int(row['frame_num'])
    ↪ == prev_frame)
next_row = next(row for row in plate_rows if int(row['frame_num'])
    ↪ == next_frame)

#Linearly interpolates every missing frame between the previous
    ↪ and next.
interp_x1 = np.linspace(float(prev_row['plate_x1']),
    ↪ float(next_row['plate_x1']), next_frame - prev_frame + 1)
interp_y1 = np.linspace(float(prev_row['plate_y1']),
    ↪ float(next_row['plate_y1']), next_frame - prev_frame + 1)
interp_x2 = np.linspace(float(prev_row['plate_x2']),
    ↪ float(next_row['plate_x2']), next_frame - prev_frame + 1)
interp_y2 = np.linspace(float(prev_row['plate_y2']),
    ↪ float(next_row['plate_y2']), next_frame - prev_frame + 1)

#Create a new row, taking the interpolated coordinates for the
    ↪ current frame from the distribution.
new_row = prev_row.copy()
new_row['frame_num'] = frame_num
new_row['plate_x1'] = interp_x1[frame_num - prev_frame]
new_row['plate_y1'] = interp_y1[frame_num - prev_frame]
new_row['plate_x2'] = interp_x2[frame_num - prev_frame]
new_row['plate_y2'] = interp_y2[frame_num - prev_frame]
new_row['interpolated'] = True
interpolated_rows.append(new_row)
else:
plate_rows[0]['interpolated'] = False

#Add the plate rows to the output rows.
interpolated_rows.extend(plate_rows)

#Sort the interpolated rows by frame_num.
interpolated_rows.sort(key=lambda x: int(x['frame_num']))

#Write csv.
with open(output, mode='w', newline='') as file:
writer = csv.DictWriter(file,
    ↪ fieldnames=interpolated_rows[0].keys())
writer.writeheader()
writer.writerows(interpolated_rows)

```

# Bibliography

- [1] Department for Transport. (2023) Vehicle licensing statistics: 2022 - gov.uk. Accessed: July 24, 2023. [Online]. Available: <https://www.gov.uk/government/statistics/vehicle-licensing-statistics-2022/vehicle-licensing-statistics-2022>
- [2] ———. (2023) Road traffic statistics - about. Accessed: July 24, 2023. [Online]. Available: <https://roadtraffic.dft.gov.uk/about>
- [3] Retail Sensing. (2023) Vehicle traffic counting. Accessed: July 30, 2023. [Online]. Available: <https://www.retailsensing.com/vehicle-traffic-counting.html>
- [4] IntuVision. (2023) intu vision va - vehicle counting. Accessed: July 30, 2023. [Online]. Available: <https://www.intuvisiontech.com/solutions/vehicle-counting>
- [5] Department for Transport. (2023) Road traffic statistics information - gov.uk. Accessed: July 27, 2023. [Online]. Available: <https://www.gov.uk/guidance/road-traffic-statistics-information>
- [6] P. McGowen and M. Sanderson, “Accuracy of pneumatic road tube counters,” in *Institute of Transportation Engineers (ITE). Western District Annual Meeting, 2011*, 2011.
- [7] May 2006. [Online]. Available: <https://www.fhwa.dot.gov/publications/research/operations/its/06108/02.cfm>
- [8] Federal Highway Administration. (2014) A summary of vehicle detection and surveillance technologies use in intelligent transportation systems. Accessed: July 29, 2023. [Online]. Available: <https://www.fhwa.dot.gov/policyinformation/pubs/vdstits2007/04pt2.cfm>

- [9] S. Oh, S. G. Ritchie, and C. Oh, "Real-time traffic measurement from single loop inductive signatures," *Transportation Research Record*, vol. 1804, no. 1, pp. 98–106, 2002.
- [10] C. Antoniou, R. Balakrishna, and H. N. Koutsopoulos, "A synthesis of emerging data collection technologies and their impact on traffic management applications," *European Transport Research Review*, vol. 3, pp. 139–148, 2011.
- [11] G. Rajapaksha and S. Bandara, "Evaluation of accuracy of the infra-red traffic logger under mix traffic condition," in *2019 Moratuwa Engineering Research Conference (MERCon)*, 2019, pp. 491–497.
- [12] J. Fang, H. Meng, H. Zhang, and X. Wang, "A low-cost vehicle detection and classification system based on unmodulated continuous-wave radar," in *2007 IEEE Intelligent Transportation Systems Conference*, 2007, pp. 715–720.
- [13] T.-J. Ho and M.-J. Chung, "An approach to traffic flow detection improvements of non-contact microwave radar detectors," in *2016 International Conference on Applied System Innovation (ICASI)*, 2016, pp. 1–4.
- [14] O. Appiah, E. Quayson, and E. Opoku, "Ultrasonic sensor based traffic information acquisition system; a cheaper alternative for its application in developing countries," *Scientific African*, vol. 9, p. e00487, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2468227620302258>
- [15] M. A. Prasetyo, R. Latuconsina, and T. W. Purboyo, "A proposed design of traffic congestion prediction using ultrasonic sensors," *Int J Appl Eng Res*, vol. 13, no. 1, pp. 434–441, 2018.
- [16] X. Yu and P. D. Prevedouros, "Performance and challenges in utilizing non-intrusive sensors for traffic data collection," 2013.
- [17] B. Barbagli, G. Manes, R. Facchini, and A. Manes, "Acoustic sensor network for vehicle traffic monitoring," in *Proceedings of the 1st international conference on advances in vehicular systems, technologies and applications*, 2012, pp. 24–29.
- [18] D. A. Whitney and J. J. Pisano, "Autoalert: Automated acoustic detection of incidents," 1995.

- [19] K. Abdulrahim and R. A. Salam, "Traffic surveillance: A review of vision based vehicle detection, recognition and tracking," *International journal of applied engineering research*, vol. 11, no. 1, pp. 713–726, 2016.
- [20] S. Abbes. (2022) Motion detection techniques. Accessed: July 28, 2023. [Online]. Available: <https://medium.com/@abbessafa1998/motion-detection-techniques-with-code-on-opencv-18ed2c1acfaf>
- [21] N. Paul, A. Singh, A. Midya, P. Roy, and D. Dogra, "Moving object detection using modified temporal differencing and local fuzzy thresholding," *The Journal of Supercomputing*, vol. 73, 03 2017.
- [22] Q.-L. Li and J.-F. He, "Vehicles detection based on three-frame-difference method and cross-entropy threshold method," *Jisuanji Gongcheng/ Computer Engineering*, vol. 37, no. 4, 2011.
- [23] Open Source Computer Vision. (2023) Opencv: How to use background subtraction methods. Accessed: July 28, 2023. [Online]. Available: [https://docs.opencv.org/4.x/d1/dc5/tutorial\\_background\\_subtraction.html#:~:text=Background%20subtraction%20\(BS\)%20is%20a,scene\)%20by%20using%20static%20cameras.](https://docs.opencv.org/4.x/d1/dc5/tutorial_background_subtraction.html#:~:text=Background%20subtraction%20(BS)%20is%20a,scene)%20by%20using%20static%20cameras.)
- [24] N. A. Mandellos, I. Keramitsoglou, and C. T. Kiranoudis, "A background subtraction algorithm for detecting and tracking vehicles," *Expert Systems with Applications*, vol. 38, no. 3, pp. 1619–1631, 2011.
- [25] H. Yang and S. Qu, "Real-time vehicle detection and counting in complex traffic scenes using background subtraction model with low-rank decomposition," *IET Intelligent Transport Systems*, vol. 12, no. 1, pp. 75–85, 2018.
- [26] Electronics and Computer Science University of Southampton. (2005) Computer vision demonstration website. Accessed: July 28, 2023. [Online]. Available: [http://www.southampton.ac.uk/~msn/book/new\\_demo/opticalFlow/#:~:text=Optical%20flow%20is%20a%20technique,in%20the%20next%20image%20sequence.](http://www.southampton.ac.uk/~msn/book/new_demo/opticalFlow/#:~:text=Optical%20flow%20is%20a%20technique,in%20the%20next%20image%20sequence.)
- [27] K. Mu, F. Hui, and X. Zhao, "Multiple vehicle detection and tracking in highway traffic surveillance video based on sift feature matching." *Journal of Information Processing Systems*, vol. 12, no. 2, 2016.

- [28] X. Chen and Q. Meng, "Vehicle detection from uavs by using sift with implicit shape model," in *2013 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2013, pp. 3139–3144.
- [29] J.-W. Hsieh, L.-C. Chen, and D.-Y. Chen, "Symmetrical surf and its applications to vehicle detection and vehicle make and model recognition," *IEEE Transactions on intelligent transportation systems*, vol. 15, no. 1, pp. 6–20, 2014.
- [30] S. Shujuan, X. Zhize, W. Xingang, H. Guan, W. Wenqi, and X. De, "Real-time vehicle detection using haar-surf mixed features and gentle adaboost classifier," in *The 27th Chinese control and decision conference (2015 CCDC)*. IEEE, 2015, pp. 1888–1894.
- [31] Y. Wei, Q. Tian, J. Guo, W. Huang, and J. Cao, "Multi-vehicle detection algorithm through combining harr and hog features," *Mathematics and Computers in Simulation*, vol. 155, pp. 130–145, 2019.
- [32] IBM. (2023) Convolutional neural networks. Accessed: July 30, 2023. [Online]. Available: <https://www.ibm.com/topics/convolutional-neural-networks>
- [33] Q. Fan, L. Brown, and J. Smith, "A closer look at faster r-cnn for vehicle detection," in *2016 IEEE intelligent vehicles symposium (IV)*. IEEE, 2016, pp. 124–129.
- [34] S.-C. Hsu, C.-L. Huang, and C.-H. Chuang, "Vehicle detection using simplified fast r-cnn," in *2018 International Workshop on Advanced Image Technology (IWAIT)*. IEEE, 2018, pp. 1–3.
- [35] K. ZHANG, R. Hang, W. Yongquan, and G. Jun, "Multi-target vehicle detection and tracking based on video," in *2020 Chinese Control And Decision Conference (CCDC)*. IEEE, 2020, pp. 3317–3322.
- [36] S. Li, W. Zhang, G. Li, L. Su, and Q. Huang, "Vehicle detection in uav traffic video based on convolution neural network," in *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, 2018, pp. 1–6.

- [37] S. Chen and W. Lin, “Embedded system real-time vehicle detection based on improved yolo network,” in *2019 IEEE 3rd advanced information management, communicates, electronic and automation control conference (IMCEC)*. IEEE, 2019, pp. 1400–1403.
- [38] Y. Zhang, Z. Guo, J. Wu, Y. Tian, H. Tang, and X. Guo, “Real-time vehicle detection based on improved yolo v5,” *Sustainability*, vol. 14, no. 19, p. 12274, 2022.
- [39] T.-N. Doan and M.-T. Truong, “Real-time vehicle detection and counting based on yolo and deepsort,” in *2020 12th International Conference on Knowledge and Systems Engineering (KSE)*. IEEE, 2020, pp. 67–72.
- [40] C. Patel, D. Shah, and A. Patel, “Automatic number plate recognition system (anpr): A survey,” *International Journal of Computer Applications*, vol. 69, no. 9, 2013.
- [41] DrMax. (2022) How does optical character recognition work. Accessed: July 31, 2023. [Online]. Available: <https://www.baeldung.com/cs/ocr>
- [42] Police.uk. (2023) Automatic number plate recognition (anpr). Accessed: July 31, 2023. [Online]. Available: <https://www.police.uk/advice/advice-and-information/rs/road-safety/automatic-number-plate-recognition-anpr/>
- [43] Driver and Vehicle Licensing Agency. (2023) Vehicle enquiry service (ves) api guide. Accessed: July 31, 2023. [Online]. Available: <https://developer-portal.driver-vehicle-licensing.api.gov.uk/apis/vehicle-enquiry-service/vehicle-enquiry-service-description.html>
- [44] B. Coifman and S. Krishnamurthy, “Vehicle reidentification and travel time measurement across freeway junctions using the existing detector infrastructure,” *Transportation Research Part C: Emerging Technologies*, vol. 15, no. 3, pp. 135–153, 2007.
- [45] M. Ndoye, V. Totten, B. Carter, D. M. Bullock, and J. V. Krogmeier, “Vehicle detector signature processing and vehicle reidentification for travel time estimation,” Tech. Rep., 2008.

- [46] B. Hellenga and L. Fu, "Assessing expected accuracy of probe vehicle travel time reports," *Journal of transportation engineering*, vol. 125, no. 6, pp. 524–530, 1999.
- [47] H. Al-Deek, A. Mohamed, and A. Radwan, "Operational benefits of electronic toll collection: case study," *Journal of Transportation Engineering*, vol. 123, no. 6, pp. 467–477, 1997.
- [48] Coriel. (2023) Accessed: July 28, 2023. [Online]. Available: <https://coriel.co.uk/automatic-vehicle-identification>
- [49] A. Haghani, M. Hamed, K. F. Sadabadi, S. Young, and P. Tarnoff, "Data collection of freeway travel time ground truth with bluetooth sensors," *Transportation Research Record*, vol. 2160, no. 1, pp. 60–68, 2010.
- [50] T. Tsubota and T. Yoshii, "An analysis of the detection probability of mac address from a moving bluetooth device," *Transportation Research Procedia*, vol. 21, pp. 251–256, 2017.
- [51] M. Carlier. (2023) Average car and van occupancy in england 2002-2018 statistic... Accessed: July 28, 2023. [Online]. Available: <https://www.statista.com/statistics/314719/average-car-and-van-occupancy-in-england>
- [52] G. Leduc *et al.*, "Road traffic data: Collection methods and applications," *Working Papers on Energy, Transport and Climate Change*, vol. 1, no. 55, pp. 1–55, 2008.
- [53] A. Faghri and K. Hamad, "Application of gps in traffic management systems," *GPS solutions*, vol. 5, pp. 52–60, 2002.
- [54] C.-H. Lo, W.-C. Peng, C.-W. Chen, T.-Y. Lin, and C.-S. Lin, "Carweb: A traffic data collection platform," in *The Ninth International Conference on Mobile Data Management (mdm 2008)*. IEEE, 2008, pp. 221–222.
- [55] X. Zhou, W. Wang, and L. Yu, "Traffic flow analysis and prediction based on gps data of floating cars," in *Proceedings of the 2012 International Conference on Information Technology and Software Engineering: Information Technology*. Springer, 2013, pp. 497–508.
- [56] J. Weitz. (2009) Arterial traffic available on google maps. Accessed: July 29, 2023. [Online]. Available: <https://maps.googleblog.com/2009/08/arterial-traffic-available-on-google.html>

- [57] D. Barth. (2009) The bright side of sitting in traffic: Crowdsourcing road congestion data. Accessed: July 29, 2023. [Online]. Available: <https://googleblog.blogspot.com/2009/08/bright-side-of-sitting-in-traffic.html>
- [58] OpenStreetMap US. (2016) Adding traffic to openstreetmap – david emory. Recording of the presentation given at the State of the Map US 2013 conference (SotM-US 2013). [Online]. Available: <https://www.youtube.com/watch?v=example>
- [59] G. Salvo, L. Caruso, and A. Scordo, “Urban traffic analysis through an uav,” *Procedia-Social and Behavioral Sciences*, vol. 111, pp. 1083–1091, 2014.
- [60] M. A. Khan, W. Ectors, T. Bellemans, D. Janssens, and G. Wets, “Uav-based traffic analysis: A universal guiding framework based on literature survey,” *Transportation research procedia*, vol. 22, pp. 541–550, 2017.
- [61] P. Posea. (2023) Best drones with longest flight time. Accessed: July 29, 2023. [Online]. Available: <https://dronesgator.com/drones-with-long-flight-time/>
- [62] The UK Civil Aviation Authority. (2019) The drone and model aircraft code. Accessed: July 29, 2023. [Online]. Available: <https://register-drones.caa.co.uk/drone-code/where-you-can-fly>
- [63] Ultralytics. (2024) Object detection datasets overview. Accessed: January 22, 2024. [Online]. Available: <https://docs.ultralytics.com/datasets/detect/#ultralytics-yolo-format>
- [64] Edouard Bekvak. (2024) Textrecognitiondatagenerator. Accessed: January 23, 2024. [Online]. Available: <https://github.com/Belval/TextRecognitionDataGenerator>
- [65] Driver and Vehicle Licensing Agency. Vehicle registration numbers and number plates. Accessed: January 23, 2023. [Online]. Available: [https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment\\_data/file/359317/INF104\\_160914.pdf](https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/359317/INF104_160914.pdf)
- [66] S. Suzuki *et al.*, “Topological structural analysis of digitized binary images by border following,” *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32–46, 1985.

- [67] R. U. Projects, “License plate recognition dataset,” <https://universe.roboflow.com/roboflow-universe-projects/license-plate-recognition-rxg4e>, dec 2022, visited on 2024-01-29. [Online]. Available: <https://universe.roboflow.com/roboflow-universe-projects/license-plate-recognition-rxg4e>
- [68] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and real-time tracking,” in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3464–3468.
- [69] D. Vedhaviyassh, R. Sudhan, G. Saranya, M. Safa, and D. Arun, “Comparative analysis of easyocr and tesseractocr for automatic license plate recognition using deep learning algorithm,” in *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, 2022, pp. 966–971.
- [70] Jaied AI, “Easyocr,” <https://github.com/JaiedAI/EasyOCR>.
- [71] Department for Transport and Driver and Vehicle Licensing Agency. (2023) Vehicle licensing statistics: April to june 2023. Accessed: January 28, 2024. [Online]. Available: <https://www.gov.uk/government/statistics/vehicle-licensing-statistics-april-to-june-2023>
- [72] J. Valmadre *et al.*, “Local metrics for multi-object tracking,” <https://arxiv.org/abs/2104.02631>, 2021, accessed: 2024-01-29.