COMP2043.GRP Final Group Report

Traffic Monitor System Based on Computer Vision

Team 12

Supervised by: Qian Zhang

Member: Yuyan Liu(16522049), Zilin Song(16522063), Jingyu Luo(16521724), Jiawei Tian(16515499), Shiye Xu(16521987)

Word count: 7553 words

Date: 10th Apr. 2019

Abstract

The project has spent the team around 5 months to develop a computer vision-based traffic monitoring system, the main purpose of the software is to aid human operators better monitoring the traffic conditions, the functionalities will include vehicle detection, vehicle tracking, number counting and so on. In the design part, the client-end of the software is a web application, which allows better flexibility and compatibility, the implementation is based on the design and more details will be further discussed.

Table of Contents

1.0.Introduction

1.1. Purpose

The purpose of this document is to present a detailed description of the traffic monitoring system based on computer vision technique, and it reports the group work procedure and achievement during developing the system. The description part includes the purpose of the application under the specific background. Furthermore, the design details associate with the system features will be explained along with design and implementation constraints. Then, this document will analyze the requirements, both functional and non-functional. In addition, it will show some researches about the purposed features which may not be implemented, it exists as part of the original design. The future work and reflection part demonstrate the group work management and how the application can be improved.

1.2. Document convention

The document uses the following conventions:

UAV	unmanned aerial vehicle
CV	computer vision
ITS	intelligent transport system
GUI	graphic user interface

1.3. Background

Traffic activities has been growing very fast for the recent several years, the successive problems are that road condition has become more complex and harder to manage. As a result, the traffic monitoring systems have expanded, and the technology used to manage traffic activities also tends to become more mature. The basic idea of traffic monitoring is based on a large number of cameras deployed on the road. With the enhancement of the infrastructure, such as data accessibility, data analysis and video processing, altogether with the increasing computing performance, being able to apply to new technology such as computer vision on traffic monitoring, even to a heavily loaded road, has become more realistic, this kind of technology is referred to as intelligent transport system (ITS) [1][15].

1.4. Project scope

In our project, two main technologies will be applied, one is computer vision, and the other one is the unmanned aerial vehicle. Ideally, the video source will be from the UAV. By working with unmanned aerial vehicle (UAV), current traffic monitoring system can handle more conditions dynamically, as human can take control of the UAV and adjust the height of it, while the traditional fixed cameras are usually not adjustable. However, in our project, we will only consider a shooting point of camera looking down, as for the device that shoots the video, it is not necessarily have to be from a UAV, since the device can have a higher cost. A video from the website which is shot by a UAV or from a tall building might be considered as the source video. The other technology is computer vision. The main purpose of this project is to introduce

computer vision into traffic monitoring, and in some related applications, this technique is already been progressively used in ITS [1]. The concept of this project is to aid human regulators to understand the information that has been provided better, the cameras that have been deployed in a large scale provide too rich information for human to understand in a limited time, video processing and analysis can extract the very basic relevant information, ideally, by connecting to some database most of the information can be processed by computer automatically, and it is more reliable to process those mechanical and repeated using a computer.

The general idea of this project is a system that can provide users with an interface that allows them uploading video stream taken from UAV or road camera to the server, this system will also be integrated with a user-friendly frontend web interface, allowing users do some simple operations such tracking a single car on browser with their own computer by clicking the buttons.

2.0.Technical Research

To gain a better performance of the software, a wide range of tools selection including operating system, existing advanced algorithm and programming languages are considered, for the front end, the framework is also considered.

2.1. Front-end

Html. The front end is implemented as website, Hypertext Markup Language is an application under the standard universal markup language. It is also a specification, a standard, which marks each part of a web page to be displayed by a mark symbol.

Flask. Flask is a lightweight web application framework written in Python. For this application, as the server is implemented by python, Flask can better realize the data exchange between the client and the server.

CSS. A cascading style sheet is a computer language used to represent file styles such as HTML or XML. The use of CSS can make the web page look better.

JavaScript. JavaScript is a literal translation scripting language that is a dynamic type, weak type, prototype-based language, and built-in support type.

Canvas. The Canvas API is a new tag in HTML5 for generating images in real time on a web page and can manipulate image content. Basically, it is a bitmap that can be manipulated with JavaScript. This application use canvas to show the result and achieve video playback through high-speed refresh.

2.2. Back-end

2.2.1. Operating system

For the back end, Linux is preferred rather than Windows as the server, the first advantage is that Linux is an open source software operating system, whereas Windows is not, therefore, it has integrated many excellent functions from provided by people around the world. Second, compared to Linux, Windows' GUI can waste a lot of CPU and memory resources, however, a GUI is not necessarily needed for a server, so, more resources can be used to serve for the software itself. Third, the cost for using Linux is lower than that of Windows [2].

2.2.2. Programming language

As for programming language, we finally decide to use python, first, Python has simpler syntax than other languages, it is easy to learn, and it is much easier to program in Python [3]. However, the main reason for choosing Python is that it has some robust libraries which are useful for our project, those libraries can be extremely handy for developers choosing Python instead of other languages such as C++ or Java.

2.2.3. Libraries

TensorFlow is a machine learning system that operates at large scale and in mixed environments. Computation, shared state, and any operation that changes that state are represented by data-flow graphs in TensorFlow. It maps the nodes of a data-flow graph across many machines in a cluster, and within a machine across multiple computational devices, including multicore CPUs, general-purpose GPUs, and custom-designed ASICs known as Tensor Processing Units (TPUs). With this architecture, flexibility is given to the application developer: whereas in previous 'parameter server' designs the management of shared state is built into the system, TensorFlow enables developers to experiment with novel optimizations and training algorithms, this meets the need of our project: we will have to do experiments on the trained model. TensorFlow supports a variety of applications, with a focus on training and inference on deep neural networks. Several Google services use TensorFlow in production, it is released as an open-source project, and it has become widely used for machine learning research. It is a well-performing tool, to show this, the description of TensorFlow data-flow model and demonstration of the compelling performance that TensorFlow achieves for several real-world applications can be found on the website [4].

3.0.Design

3.1. Use case diagram

In figure 1 is the use case diagram for this application. When users access the client, they can upload video to the server immediately or read some related information displayed in the client. After uploading video, users can perform future operations. Due to the network speed limitations, it may take some time to wait for the video to be successfully uploaded to the server. Once the server successfully receives the video, it will automatically run the detect function immediately. When users click the detect button, the back end will send the result to the client and the client will show the result

on the video player. If users want user track function, they need to pause video and click the vehicle that they want track. The front end will send the number of the current frame and the current vehicle location to the back end, the back end will run the track function and sent the tracking result to the front end. If users want to download the processed video, they can click the download button and then the front end will send a message to the server, the server will send the processed video to users. This is the basic idea of the interaction between the front end and server of this application.

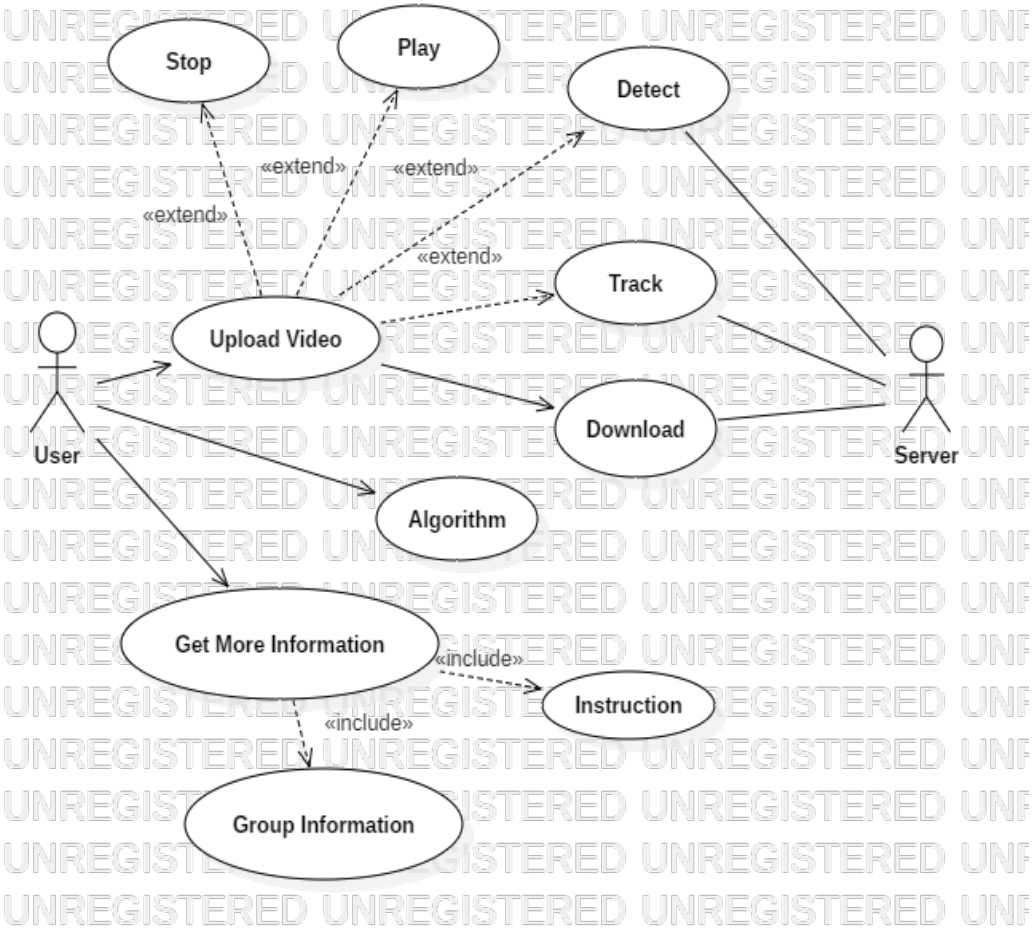


Figure 1 use case diagram

3.2. Front-end design

For the front end, in order to make it easier for users to use this app, we decide to use HTML to implement it. HTML5 is a markup language to build a website and it is supported by almost all the browsers. Compared with GUI, HTML may run slower as everything in the website is on the server, and the video player on the website may not perform as well as the GUI. By using HTML, user can use this application with a browser on any platform, they do not need to consider whether the system is compatible with this software. The diagram above is use case diagram which represents some operations users can do within this application. As the diagram shows, user needs to upload the video for the next step. After uploading the software, users can play the uploaded video and use functionality of the application. There are several buttons that users can use to use these functions. When the whole video has been processed, users can download the processed video if they want. Users can also understand how this

application runs by visiting the website or clicking some useful links or buttons on the website.

3.2.1. Activity diagram

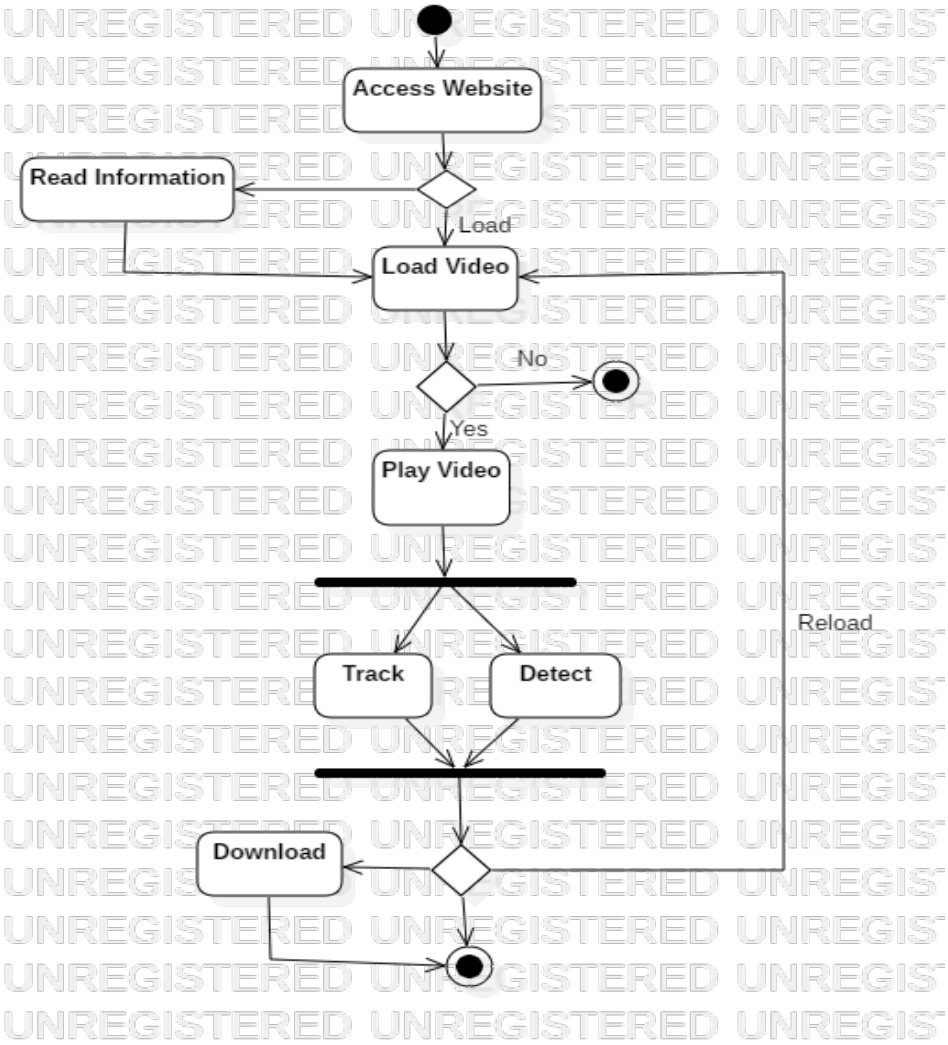


Figure 2 activity diagram for front-end

Figure 2 is the activity diagram of the client. It shows the activities that users can perform in the client and the order of each activity. This graph is used to help reader to better understand this application. As the graph shows, if user want to real use this application, they must upload a video to the server first, and then they can use the other features of the app.

In conclusion, the main purpose of the front-end design is to make it easier for users to use the app. HTML is chosen as it can be accepted by all browsers. To reduce the waiting time, server only send the result to the client. All the design is help user to adapt the application as soon as possible.

3.2.2. Home page

To make it easier for users to use, we divided the website into several parts. When users access the website, the background of the web page will automatically play a processed video. The processed video is a video that been processed by this

application already. By playing this video, users can intuitively know the effect of this software. In addition to the processed video, there are three buttons that users can click. The first button is used to load video. Users can click the button to choose video they want to process in their computer. The second button is used to submit the video. When the users select the video, they need to click this button to upload the video to the server and jump to the video paly site automatically. The third button is used to show some detail information about this application. The last button connects the website which displays some information about the whole group. There is also some information on the website that can help people to general idea this application. When users first access the website, they can read this information to understand and use the application as soon as possible.

3.2.3. Video player page

When users successfully upload video, they will automatically jump to the video play site. To improve users' experience, canvas is used to replace the video player. Even html has its own player, it can only play video and transport video form backend to front end will takes a lot of time. By using canvas, the server will send the processed result as pictures one by one and the canvas will refresh at least 24 times per seconds. In this way, users will still feel watching video instead of sperate pictures. In addition to the video player, there is an information display box at the top of the player. The box is used to represent some useful information such as the speed of the car being tracked, the number of vehicles in the current screen and so on. We added this feature to give users a better understanding of the video being processed. There are also some buttons under the video player which users can use to do some operations to the video. The list of buttons are as follows: stop button, detection button, tracking button, stop tracking button download button and upload button.

3.2.4. Information display page

Another page of the front end is used to show more details and give some useful instruction about this application Even if the first page has some simple explanation about this application, it does not provide more specific information to give users a deeper understanding. In this website, users could find a more detailed description of this application. This page also provides users an instruction for use. If users feel confused about this application or do not know how to use this application, it is really helpful to read the instruction on this page. Since the implementation of this application feature is based on YOLOV3, if any user interest in it, they can find the relevant information about YOLOV3.

3.3. Data transmission

In order to make the information transfer more convenient and easier between client and server, flask is used in the front end. Flask is a lightweight web application framework written in Python. By using flask, the front end will send the video to the server and then call the detecting function. After detecting the video, server will send the processed result to the front end as .json file. This is the data transform for detecting

function. As the tracking function, when users click the car, they want to track, the client will get the relative position of the corresponding numbered frame and send it to the server. The server will send the detecting result as the .json file and the player will display the result.

3.4. Server design

Back-end is designed to take control of object detection and tracking part. It should keep communicate with front-end client and react to it. Four functional requirements will be satisfied in the back-end:

- **Vehicle detection**: by using the state-of-art detection algorithm, this system can detect the type of the vehicle from a video stream (especially from a UAV camera) and coordinates them with bounding boxes so that it will be easier to identify them. During detecting, different types of vehicle have different colored boxes and the type of vehicle will be displayed above the box. At the website, people can choose whether show the detection result. There is a button on the website that helps user to do this operation.
- **Vehicle counting**: while tracking, the server will count the number of different types of vehicles on the current video frame and show the numbers on the upside of video. It will be refreshed by every specific time period.
- **Vehicle tracking**: users can track a specific vehicle by clicking the vehicle on the screen while detection. Then, all other objects' boxes will disappear expect the selected vehicle, which will be easier for user to identify. The detailed information of that vehicle will be displayed on the top of the video.
- **Vehicle speed monitoring:** when user click specific car and begin to track it, the system will start to calculate the speed of this car. User will see the speed at the upside of the screen. When user do not want to track the car or decided to track another car, the speed will automatically disappear or change.

3.4.1. Activity diagram

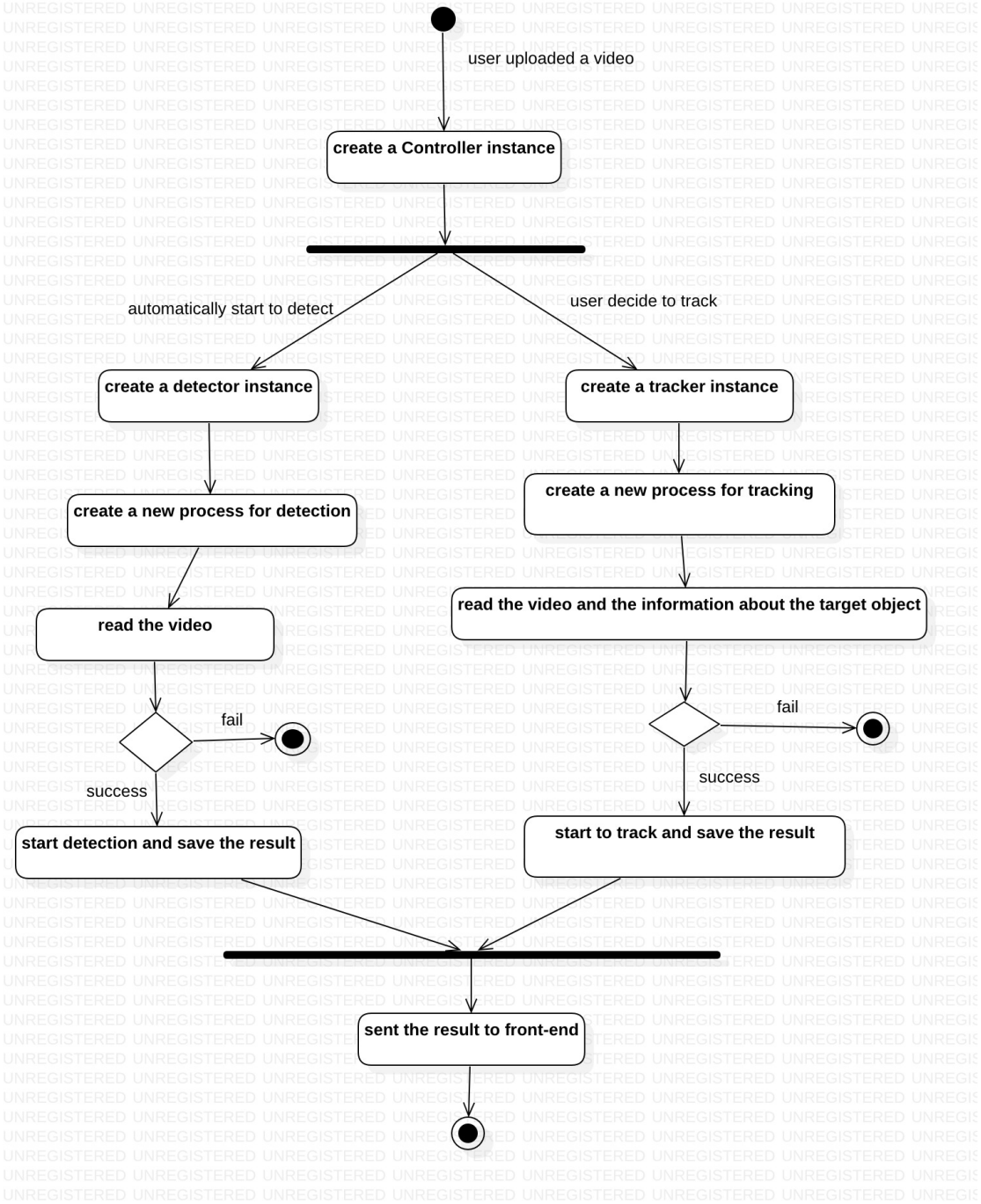


Figure 3 activity diagram for back-end

When the video source is updated, the front-end will notify the back-end. The detection will automatically start while tracking need to wait for the requirement sent by the user. Since the tracking process is needed only when the user wants to. Front end will create a controller instance first. Then all the functions can be called by it.

3.4.2. Sequence diagram

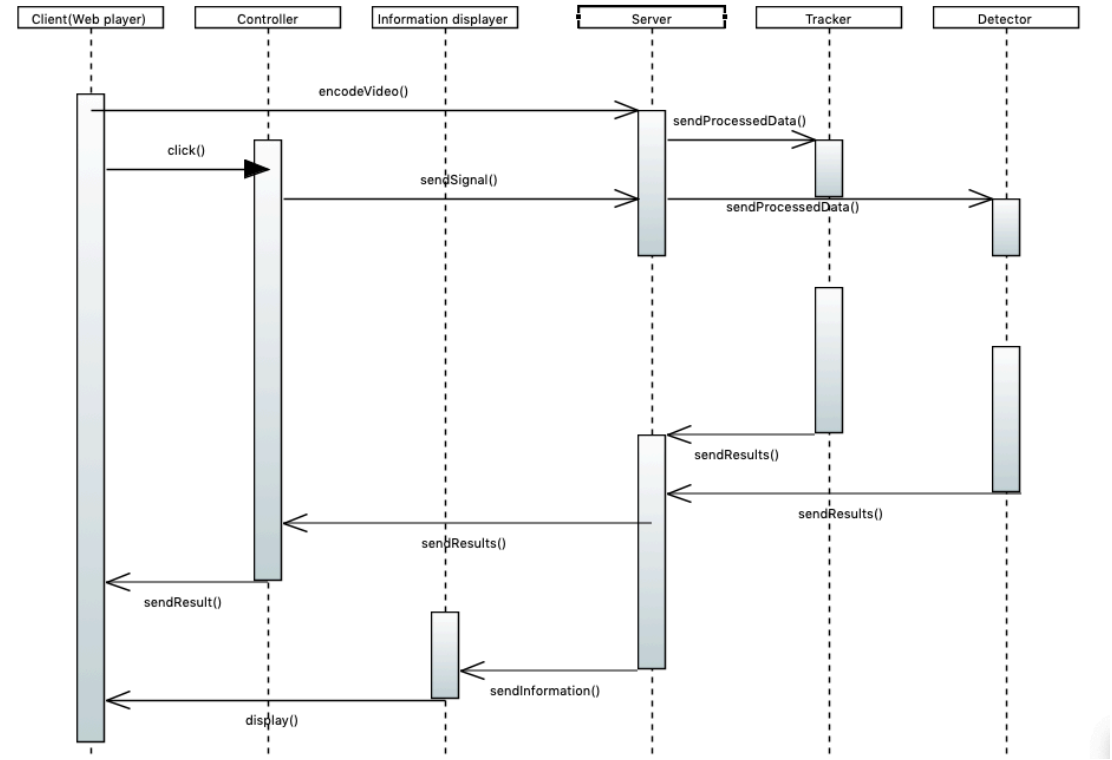


Figure 4 sequence diagram

This diagram shows how different components interact with each other.

3.4.3. Two threads

In our design, we separate detection and tracking into two threads. We need tracking and detection to run at the same time. Detection result of the same video won't change; therefore, detector will start to work as soon as the video is updated. However, tracker will work only when the user selects an object. In addition, user can change a target or cancel tracking at any point in time. Based on this requirement, they must run in two different threads. They will share the detected box data, since initializing a tracker requires a frame and the bounding box points of that object.

While the video is read frame by frame, the detector and tracker will calculate the result (bounding box and class information). At the end of each loop, the result will be sent to the front end.

3.4.4. Class diagram

Based on that, we designed the back-end as three components: controller, detector, and tracker. Their class diagrams are shown in the figure respectively.

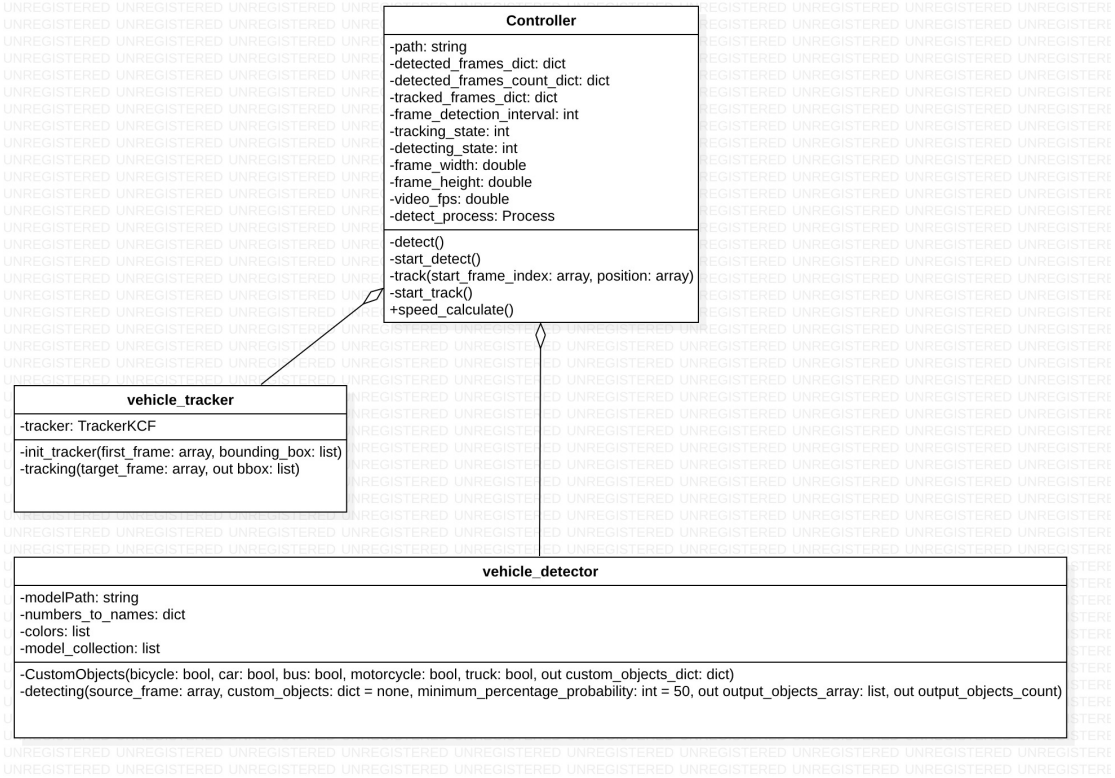


Figure 5 class diagram

3.4.5. Controller

Controller will interact with the front-end client directly. All the processes start from it.

Attributes

- path: stores the path of video source that will be used to read the video. It has a default value, the received video stream uploaded from the front end should be save to that folder.
- detected_frames_dict: a hash map whose keys are bonded to frame number, and values stores the results from detector.
- detected_frames_count_dict: a hash map storing the total number of a certain type of vehicle on a frame.
- tracked_frames_dict: a hash map whose keys are bonded to frame number, and values stores the results from tracker.
- frame_detection_interval: detect or track once several frames, default set to 1
- tracking_state: records whether tracking process is running. If so, it will be set to 1, 0 on the contrary.
- detecting_state: records whether detecting process is running. If so, it will be set to 1, 0 on the contrary.
- frame_width, frame_height, video_fps: the basic information about the video source.
- detect_process: a process object, can be started or stopped by the controller.

Main functions

- detect(): reads the video, detects objects on each frame and save the results. When it runs, it will set detecting_state to 1, 0 when finished.

- start_detect(): the detection function that should be called by the controller directly, it will check the state of detection process first. If the old process is running, it will restart the process. This is designed for cases that the user refreshes the page or want to load another video.
- track(), start_track(): same with detect() and start_detect().
- speed_calculate(): this function can be called only while track() is running. It will calculate the speed of the target object and save the result.

3.4.6. Detector

Detector is a class that will be implemented with the detection logic. In our design, it will receive the video stream from the front-end, and use the detection algorithm to calculate the information of each vehicle, such as its bounding box position, type, and the result confidence.

Attributes
- mode_path: stores the path of trained model, used to load the model.
- numbers_to_names: store the names of classes that can be recognized by the model and give each of them a number.
- colors: a list stores the colors of bounding box for each class.

Main functions
- detecting(): In our design, this function receives the frame to be detected from the controller, and the target classes, the minimum percentage probability that can be accepted. It will pre-process the image, such as reshape and re-format, and send it to the model. After calculation, the result will be stored and sent to the front end.

3.4.7. Vehicle Tracker

Similar to detector, all the tracking will be done by this class.

Main attributes
- tracker: an object from openCV library, we will use it to do the calculation.

Main functions
- init_tracker(first_frame, bounding_box): tracking algorithm needs the first frame and the bounding box of that object, these parameter will get from the controller.
- tracking(): after initialization, this function will update the position of bounding box frame by frame, and save the result, send it to the front end.

3.4.8. Data Structure

In our design, there are communications between front-end and back-end, as well as different classes.

Unified data structures
- Bounding_box: [x1(float), y1(float), x2(float), y2(float)], a list or similar data structure containing the left top point and right bottom point of the box.

- Object_information: {class_name(string), percentage_probability(float), box_points(Bounding_box)}, a hash map or similar data structure containing the above information of each detected object.
- Frame_information: a list or similar data structure containing all Object_information within one frame.
- Frame_dictionary: a hashmap whose keys are the frame numbers.

4.0.Requirement Analysis

This part demonstrates a detailed requirement analysis of the whole system, it is comprised of two main parts: functional requirements and non-functional requirements. Functional requirements are based on the purpose of this project, it mainly describes the functionalities of the application, therefore, the design part will largely refer to functional requirements. Non-functional requirements are expectations of this application, it is realized and perfected through multiple iterations of improvement during the development period.

4.1. Functional requirement

4.1.1. System features

Vehicle detection
In the input video, vehicles are detected as they are bounded by boxes, which are rectangles with a colored edge, the edge is to signal that the current functionality is detection. This function is purposed to aid human observer to recognize vehicles, especially when a large number of vehicles are in the screen at the same time. The video to be processed should be taken from a valid angle, in this project, it should be taken from high above the road, especially from a UAV, however, a viewing angle from a high building is also valid as it simulates the UAV's view, and instead of moving along the road, the viewing point should be fixed.

Vehicle tracking
The system can track a single vehicle at one time. Tracking is differentiated from detection by that it takes record of the tracked vehicle's movement track from time to time, whereas detection can only recognize a vehicle at each frame.

Vehicle counting
The system keeps monitoring the number of vehicles appears on the screen, the number is displayed on the user interface and refreshes from time to time as vehicles enter and leave the scene.

Speed monitoring
This feature is based on vehicle tracking since only the tracked vehicle is recognized by the system for all the time, once the tracked vehicle is selected, on the server side, the speed is calculated automatically and displayed on the user interface.

4.1.2. Human computer interaction

Upload video from local files

The application is built on a website, on the user interface, by clicking the upload button, a file selection dialog box will pop up and user can choose a video that he or she wants to process, the video is uploaded to and processed by the server. After that, the video is displayed in the video player as the user can watch it.

Information display
After uploading the video, the system will automatically start detecting, and the count of vehicles is displayed on the screen, however, user can choose to hide all the information by click the button on the interface, the boxes will disappear and as so the original video will be played.

Track a vehicle
When user wants to track a vehicle from the video, he or she has to pause the video and click the target vehicle as to track it, when the system starts tracking, all other boxes around other vehicles except the tracked one disappears, the tracked vehicle's detailed information is displayed to user.

Video playback
When playing the video, users can playback the video whenever they want. After playback, the detection result will disappear, and users can click the detection button to achieve detection function. It also allows users to track a car, but the previous tracking results will disappear.

Download video
This App allows users to download the processed video if they want. Users can download the video by clicking the download button on the website. However, downloading the video may take some time due to the speed limit.

4.2. Non-functional requirement

Performance
The system is required to be a real time system as the only time user spends on waiting is for video upload and sent back to the client. Ideally, the system can immediately calculate the detection results and add boxes around the vehicles, however, the model of the neural network can bring a massive workload to the computer, only a high performance computer can process the input data with an ideal speed, for normal personal computer with 1.6GHz processor, it is very difficult to run in real time, the requirement of performance can be considered to be reduced at the discretion of this.

Security and safety
Any sensitive or private information is not suggested to be uploaded to our web application, information such as license plate number that appears in any uploaded video will be protected and will not be disclosed to third party.

Compatibility
The application is a web application, user is able to access it via any main stream web browser of the up-to-date version such as Chrome and Firefox, therefore it should be compatible on almost all main stream operating system such as Windows 10, OS X and Linux.

Quality
The resolution and frame rate of the video should be 720p and 24fps, in addition, the video transmission rate should be fast enough to transmit data as soon as the last frame is processed, another component needs to sync with the video is the bounding boxes, it should appear at the same frame and position as the target vehicle.

5.0.Implementation

5.1. Front-end

To make it easy to use the application, the client is designed to implement as website. The website is divided into four parts.

Video label
The first part is the page users access directly. To help users better understand the application, a video will play on the background. To achieve this function, the video target is used as shown in figure 6

```
<video loop id=bg-video class=fullscreen-video>

    <source src="{{ url_for('static', filename='video/home.mp4') }}" type="video/mp4">

</video>
```

Figure 6 code fragmentation

Upload_file() function
There are serval buttons that users can use in the first page. The first button is “Load Video” button and the second button are submitting button. When user wants to use this application, they can click Load video button to select a video and then click submit button to upload. The submit button function is implemented by using Flask framework. To achieve the function of uploading files to the server, a function called upload_file() is created. When users click the button, the button will call this function and detect whether this file is exit. When the file is successfully uploaded, the server will detect the file and perform the corresponding calculation.

Button label
In this application, button is especially important as user needs to da operation by click button. Some buttons are connected to hyperlinks, some can call corresponding function such as upload file, some can return its parameter to the server and server will call the corresponding function. At first, the design is when users click the button, the button can call the function directly and get the result. However, it is not realistic as the button cannot call function in the server. In order to achieve the corresponding function of the button, flask is used. To make button looks better, many css style and icon is used. Ionic is an open source, free code base for developing hybrid mobile apps.

The web pages
The first page contains three buttons and some basic information that users can general understanding of this application by reading. The page looks like the in figure 7

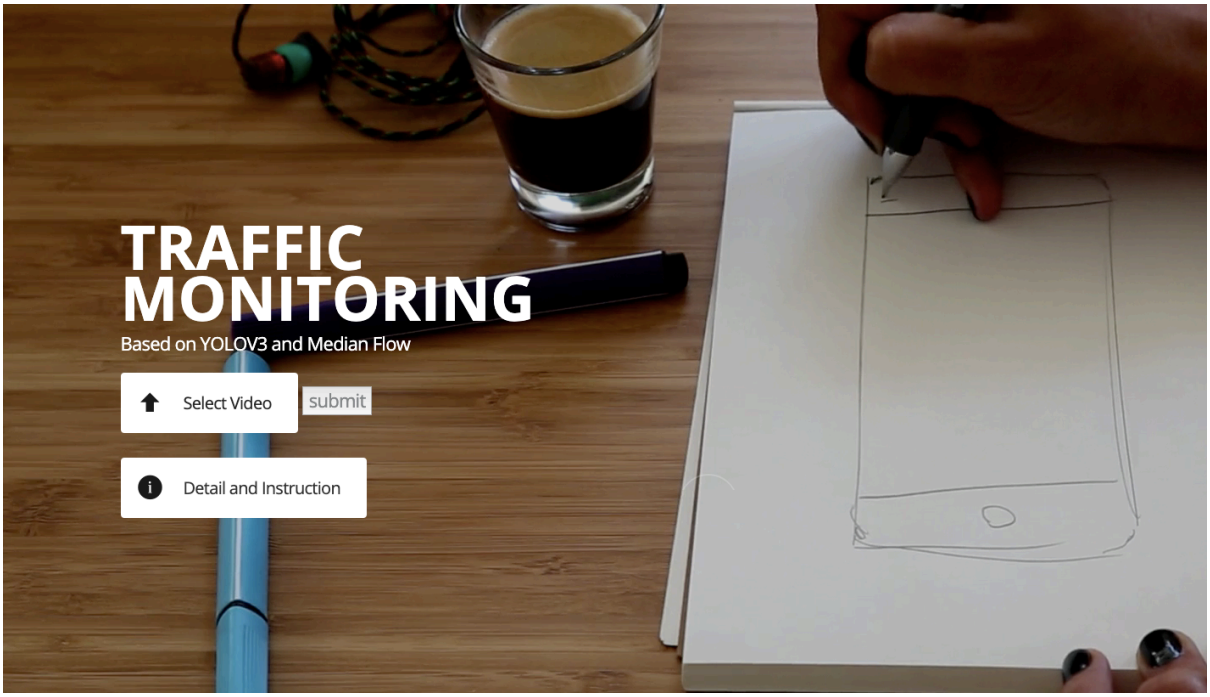


Figure 7 web page

The second part contains some information more specific and some relevant information that users may interest in. For example, this application relies on YOLO3, users can find some information about YOLO3 and if they really interest in YOLO3, they can even find the hyperlink to the official website of YOLOv3. It also contains some instruction that users can read to better use this application. The pages look like as in figure 8.

Figure 8 web page

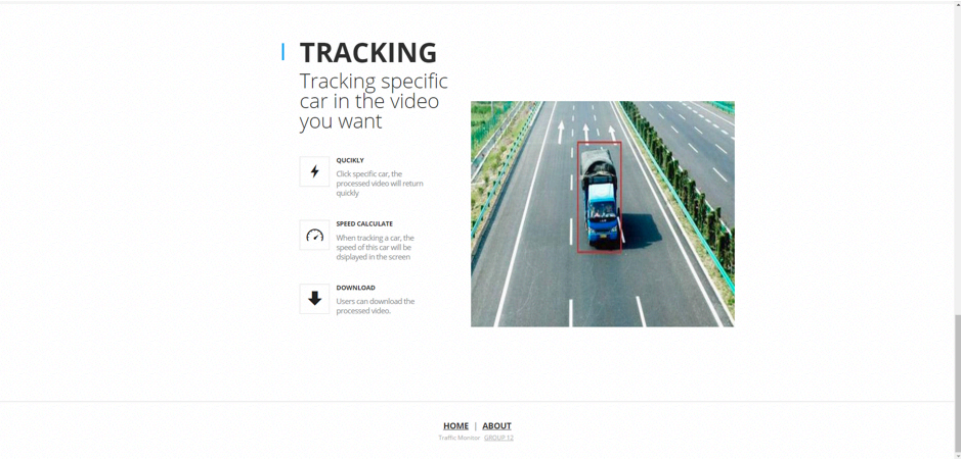


Figure 9 web page

The information displaying page is shown in figure 10, it is an introduction of the idea of our project, the picture on the right is the architecture design and the left-hand side is a summarization of the project.

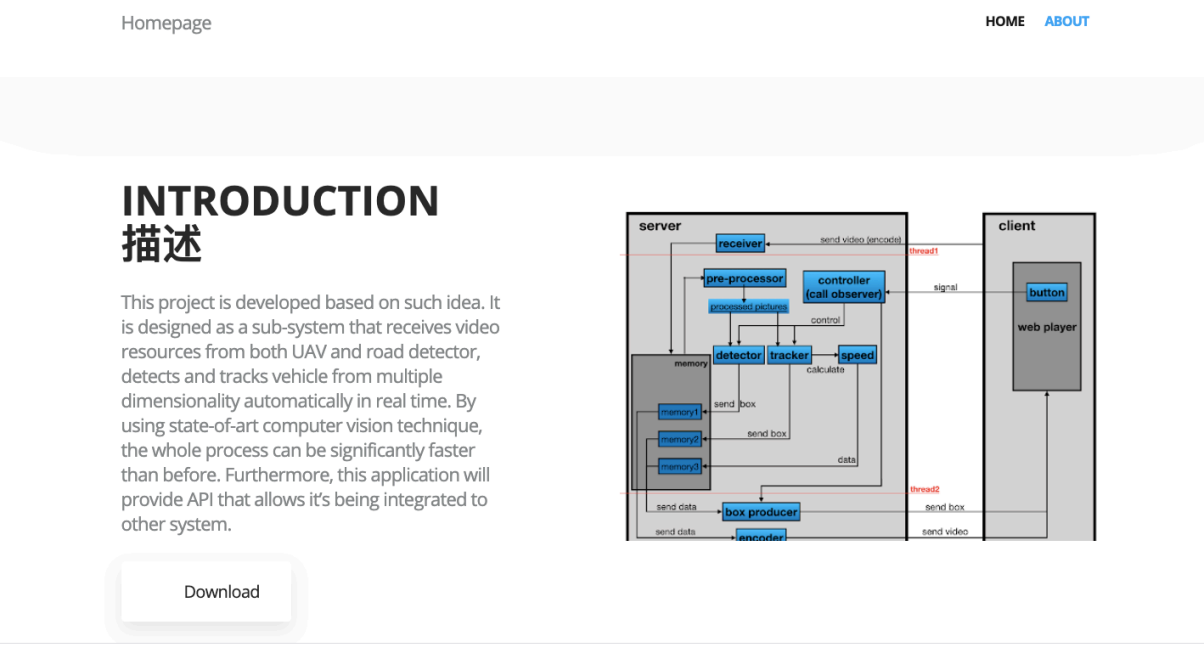


Figure 10 web page

The video page as shown in figure 11 and 12, for detection and tracking respectively, mainly contains a play window and several buttons that users can control video playback. For easy operation, user just needs to click the button below the player. On the top of the video player, it can display some information that user may interests.

Figure 11 detection

Figure 12 tracking

Canvas label

Canvas label is used to display the result to the user. At first, the design is to display video directly and push the result to user in the form of a barrage. When implementing this function, as the number and direction of the barrage is determined by the track, when playing the video, the barrage can only move horizontally which is not meet the requirements of this application. The second program is just playing the processed video directly. However, the problem the program needs longer time to process the video and it cannot change while play the processed video so it cannot achieve tracking function. The last method to make it is to use the canvas. By using canvas, the server can send the result one frame at a time to the client and the client will display the result

at a high-speed refresh which will looks as video. As a result, users will not have to wait a long time to get the result of the process and when user uses tracking function, the corresponding result will display in time.

WebSocket
By using canvas, the client can show the result of the process to the use in a timely manner. As the client needs to receive the result continuously, WebSocket is used. WebSocket is a protocol for full-duplex communication over a single TCP connection. By using WebSocket, the client can continuously accept the test results from the server.

The Flask implementation
As the server is written in Python and the client is implemented by html, to achieve the data transform between the client and the server, flask is used. To implement, several questions need to be solved. When implementing the client, flask was not considered, as a result, some code on the front end needs to be revised to make sure the website can be used in conjunction with flask. By using Flask, server can monitor the user's actions on the client and get the button's parameter when the button is clicked and then call the corresponding function. For instance, after uploading the video, when users want run detect function and click detect button, the website will return 2 to the server and the server will run the detect function and return the detecting result to the client.

5.2. Back-end

Based on the design, backend consists of three classes. Detector and Tracker are responsible for computation while Controller should interact with the frontend and manage the communication between Detector and Tracker.

5.2.1. Detector

To balance the performance and accuracy, we chose the YOLOV3 algorithm to detect objects. At first, we want to implement the algorithm by ourselves. After some research, we found an object detection library called imageAI, which currently supports object detection, video detection and object tracking using RetinaNet, YOLOv3 and TinyYOLOv3 trained on COCO dataset. It provides powerful interfaces for engineers to use some state-of-the-art Deep Learning and Computer Vision algorithms by using simple and few lines of code. However, this library was highly capsulated, we cannot use it to satisfy our functional requirement directly. For example, we want to the detector send the boxes of detected vehicles to the front every frame, but the interface does not have such function. Therefore, we decided to use the trained YOLOv3 model provided by this library to build our own detector.

We first set the default parameter values, such as set IoU (Intersection-over-Union) to 0.5, input image size to 416 * 416 and only accept the results whose percentage probability is greater than 0.5. When the video is uploaded, the Controller will read and pass a frame to the detector at each iteration. Before putting this image to the model, the following steps should be done: (1) zoom the image to the input size; (2) normalize the image array; (3) expand it into one dimension. Then we use the interface to run the model. After that, the model will return the detected boxes, scores and classes to the controller.

5.2.2. Tracker

Based on the design, after tracker was initialized with a bounding box and the first frame, it should update the bounding box frame by frame. We found that OpenCV 3.4.4 provides 8 different trackers. Compare the trackers from a practice standpoint, we found that the Kernelized Correlation Filters (KCF) tracker is more accurate and faster than MIL, in addition, it showed a higher performance when reports tracking failures than other algorithms. Therefore, we decided to use the KCF tracker to track the vehicle.

The tracker provides two functions. The first one is the init(), which requires two parameters: frame and box, to initialize the tracker. Another one is update(), which receives a new frame, and return the tracked bounding box.

The following steps shows how we used the API to track an object:
When the user clicks an object on the screen, the front end will pass the frame number and the coordinate of the click to the controller. Then the controller will read the video stream from that frame, determine the bounding box, and pass those data to initialize the tracker. After that, the tracker can update the bounding box continuously.

5.2.3. Controller

Front-end cannot access to the tracking and detection API directly, instead, it will use the interfaces provided by the Controller to run functions. Some features are achieved as follows:

Parallel computation: At the beginning, we decided to create two threads in the controller, one for detection and another for tracking, to run them in different cores. And these two threads should share the dictionary of detected bounding box, since the tracker need use the result to initialize.
However, when we tried to implement this design, we found that both of the two threads only ran on the same core. This issue was caused by global interpreter lock in CPython, which prevents multithreaded programs from using the multiprocessor systems in certain situations. And currently this problem cannot be solved. Therefore, we had to give up using multi-threading.
Instead, we put the detection and tracking into two processes. Multiprocessing library also provided some data structures that can be shared by different processes. We use this library to achieve our design and found that the result reached our requirement. Tracking and detection can run at the same time on different cores, and they can share the detected boxes.

Tracking object determination: Controller is also responsible for pass the bounding box to the tracker, but the front end will only send the coordinate and frame number. We designed an algorithm that can determine which car is more likely to be tracked. Firstly, we select all the candidate boxes according to the coordinate and the number. The pseudocode is:

```
Input: a coordinate (x, y), the frame index i
Output: a list of candidate boxes C
C = [ ]
detected_box = detected_frames_dict[i]
for box in detected_box do
      if (x > box[0] and x < box[2] and y > box[1] and y < box[3]) then
            C.append(box)
return C
```

If a box covers the coordinate, it will be added to the candidate array. Then the algorithm will select the most matched one.
Given a coordinate (x, y), a candidate box [(x1, y1), (x2, y2)], where (x1, y1) is the coordinate of the upper left point, (x2, y2) is the coordinate of the lower right point. The probability score S of that box is defined as below:

$$S = 1 - \frac{\left|x - \frac{x1 + x2}{2}\right|}{|x2 - x1|} + \left|\frac{|y - \frac{y1 + y2}{2}|}{|y2 - y1|}\right|$$

Since x and y are in the range of (x1, x2) (y1, y2), the range of S is [0,1]. If the center of a box is closer the (x, y), it will be given a higher score.
After all the candidates were visited, we can select the box with the highest score to track.

5.3. Speed calculation

Due to the time limit, this feature cannot be implemented, however, the group member had done some research for this part, and in this section, the design and an ideal implementation will be demonstrated as an idea, also, there will be analysis of speed calculation based on several papers.

5.3.1. Methodology

Since the camera captures the images from a certain angle above the road, the image is somehow distorted, the curb line in the distance tends to converge, this results in the car far away from the camera seems to be moving slower than those are closer to the camera even in reality they are driving at the same speed [6], also, in another method [7], view calibration is introduced, integrated with vehicle tracking, the speed of a vehicle can be estimated based on its location variation over contiguous number of frames.

Furthermore, in both aforementioned researches, some pre-requisite to estimate speed is mentioned, in Park et al.'s research, the geometry and dimension of the road should be known in advance, then is the conduct of view calibration, a transition of 3-D real world coordinates of road to the 2-D abstract coordinate is obtained by view calibration. In Hua et al.'s research, it adopts a data-driving method, which requires several pre-conditions and strong assumptions to calculate the speed, first, the camera should be static, second, the maximum speed limit of the road is known and at least one vehicle is driving at the maximum speed in the footage.

5.3.2. Speed estimation

In the second method mentioned above, to estimate the speed, the calibrated coordinate and the timestamp for the video are needed, in one of the methods, a one-second based velocity is used to smooth and eliminate the noise of timestamp and errors of view calibration [7], in this project, only some simple steps will be introduced.

$$v_{inst,t} = \frac{D(r_t,\ r_{t-1})}{(T_t - T_{t-1})}$$

In the equation, $v_{inst,t}$ is the instantaneous speed at frame t, D(,) is the Euclidean distance between the two given points, here is position change from the previous frame to the current, and T_t is the time stamp at frame t. The aforementioned one-second based velocity is simply adjusting the frame number from t-m to t to make the time interval close to one second, which is simply written as:

$$v_{1,t} = \frac{D(r_t,\ r_{t-m})}{(T_t - T_{t-m})}$$

The $v_{1,t}$ is basically velocity at each second, then, to smooth the noise and error, an average of one-second based velocity from frame t-m to t will be calculated. Again, this is just a simple description of speed estimation, more detailed steps are not included.

6.0.Evaluation

In order to figure out the performance if this software, we did several tests on detection and tracking, and the results are evaluated in this part.

6.1. Detection

Two different videos are used as input, video NO.1 has screen size of 854*480 and frame rate 29.97 fps, video NO.2 has screen size of 640*360 and frame rate 25 fps, and their lengths are different so running time should be respectively compared within the same video. these two videos are put into the programs which set the detection image size to 416*416, 320*320 and 208*208 respectively. The results are shown in table 1.

Input video	image size setting	Minimum frame rate (fps)	Average frame rate (fps)	Running time
Video 1 (854*480, 29.97fps)	416*416	18.29	19.92	12.13
	320*320	20.59	23.50	10.66
	208*208	24.68	28.54	9.27
Video 2 (640*360, 25fps)	416*416	16.20	20.59	107.29
	320*320	18.88	24.32	91.58
	208*208	23.71	29.96	75.18

Table 1 results for detection

The table shows that as the size of detection images becomes smaller, the frame rate becomes higher, also the running time becomes shorter.

6.2. Tracking

in table 2, a single video is used as input, two different algorithms are used for testing, the image detection size is set to 208*208 for both algorithms, although Median Flow seems to have better results, in our project we choose KCF due to its accuracy and reliability, this can be seen from the table that the average frame rate of Median Flow varies largely from its minimum value, this is because at some frames the tracking is lost, as a result the frame rate as those frames are abnormally large.

Input video	Tracking algorithm	Minimum frame rate (fps)	Average frame rate (fps)	Running time
Video 1 (854*480, 29.97fps)	KCF	8.61	10.92	16.55
	Median Flow	19.06	48.45	7.03

Table 2 results for tracking

Overall, the results meet the expectation and are acceptable.

7.0.Future work and reflection

The purpose of this part is to explain the defects, shortcoming of our work and the finished product, it is also a chance to look ahead and discuss more possibilities of the project, so, some researches and concepts which can perfect the software will be explained and shown.

7.1. Future work and further study

One part of this project that can be further improved is speed calculation, a rough sketch of the methodology presented, some detailed method of calculation is presented in relevant papers. However, those materials found are focus on a view much higher above the road, our results appear to be parallel to the road surface, so it can be difficult to apply such methods onto our application. If an overlooking view can be implemented or a method to estimate speed from a parallel view can be found, speed calculation can be more feasible and possible.

7.2. Reflection on technical development

The core technology of the project is based on a well-trained neural network, so a crucial part of implementation is the neural network model training. The model should better be obtained by training it ourselves if time permits, however, the access to HPC, which is essential for model training, was delayed due to the first application is rejected. Later, other missions and items are scheduled so this part is put on hold. At the current stage, a ready-made model is handier and time-saving, so we decide to use a ready-made model.

The second to mention is the dataset, to make the project more original, it is better to create a dataset by ourselves, concretely, we could use UAV or shoot from some high

buildings to capture some pictures for training, and if allowed, we can shoot some films of traffic transporting for testing the application.

7.3. Reflection on team work and management

Overall, we spent some effort during the whole process of the project, especially in the last semester, we communicated a lot in order to come out with some idea and refine the project. To conclude the insufficiency of our work, first, we can be more efficient on report writing and the progress of initial preparation, such as design and HPC access, the report had spent us too much time, our working pattern, which is to revise the report word by word together seems to be inefficient, and another problem is the process is reversed in the early days, we should finish the design perfectly instead of starting discussing the technical details without any plan.

8.0.Conclusion

First, our group have produced a complete application, although it does not meet some requirements initially mentioned, the final product is very similar to what is required, the most core functionalities including vehicle detection and tracking, vehicle counting and type recognition, for the front-end, uploading and downloading videos, presenting the boxes and clicking a certain vehicle to track, are implemented in our product. Besides, for those features that are not implemented, a demonstration of the idea is presented.

During the process of the project, the group members have learnt some useful techniques by themselves in a limited time, this is a chance to improve an individual's learning skills and abilities. Moreover, except for the technical skills, the interpersonal skills of all member are also improved, since we have to communicate frequently and work on the same target for a long time. By this project, we experienced a whole software development process and learnt how to collaborate with people in a team, it helps us to be prepared for the future.

Reference

[1] Buch N., Velastin S., Orwell J., A Review of Computer Vision Techniques for the Analysis of Urban Traffic, (Sep. 2011) IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 12, NO. 3

[2] Economides N. Katsamakas E. Linux vs. Windows: A Comparison of Application and Platform Innovation Incentives for Open Source and Proprietary Software Platforms, (2006) Elsevier B.V.

[3] Python Software Foundation. Python Language Reference, version 3.6. Available at http://www.python.org

[4] TensorFlow, Available at: https://ai.google/research/pubs/pub45381

[5] MosesJohn Olafenwa, ImageAl. Available at: https://github.com/OlafenwaMoses/ImageAI

[6] Hua S., Kapoor M., Anastasiu D., Vehicle Tracking and Speed Estimation from Traffic Videos, Department of Computer Engineering, San José State University, San José, CA.

[7] Park M., Kim J., Lee Y., Park J., Suh W., Vision-based surveillance system for monitoring traffic conditions, (Oct. 2017) Springer Science and Business Media New York

[8] Joseph Redmon and Ali Farhadi, YOLOv3: An Incremental Improvement (8 Apr 2018). Available at: https://arxiv.org/abs/1804.02767

[9] Ionicons. Available at: https://github.com/ionic-team/ionicons

[10] Li Jiea,b, Henk J. van Zuylenb,a* ,Road Traffic in China ,Procedia - Social and Behavioral Sciences 111 (2014)107 – 116, Available at www.sciencedirect.com

[11] Yilmaz, A., Javed, O., and Shah, M. 2006. Object tracking: A survey. ACM Comput. Surv. 38, 4, Article 13 (Dec. 2006), 45 pages. DOI = 9.1145/1177352.1177355. Available at http://doi.acm.org/10.1145/1177352.1177355

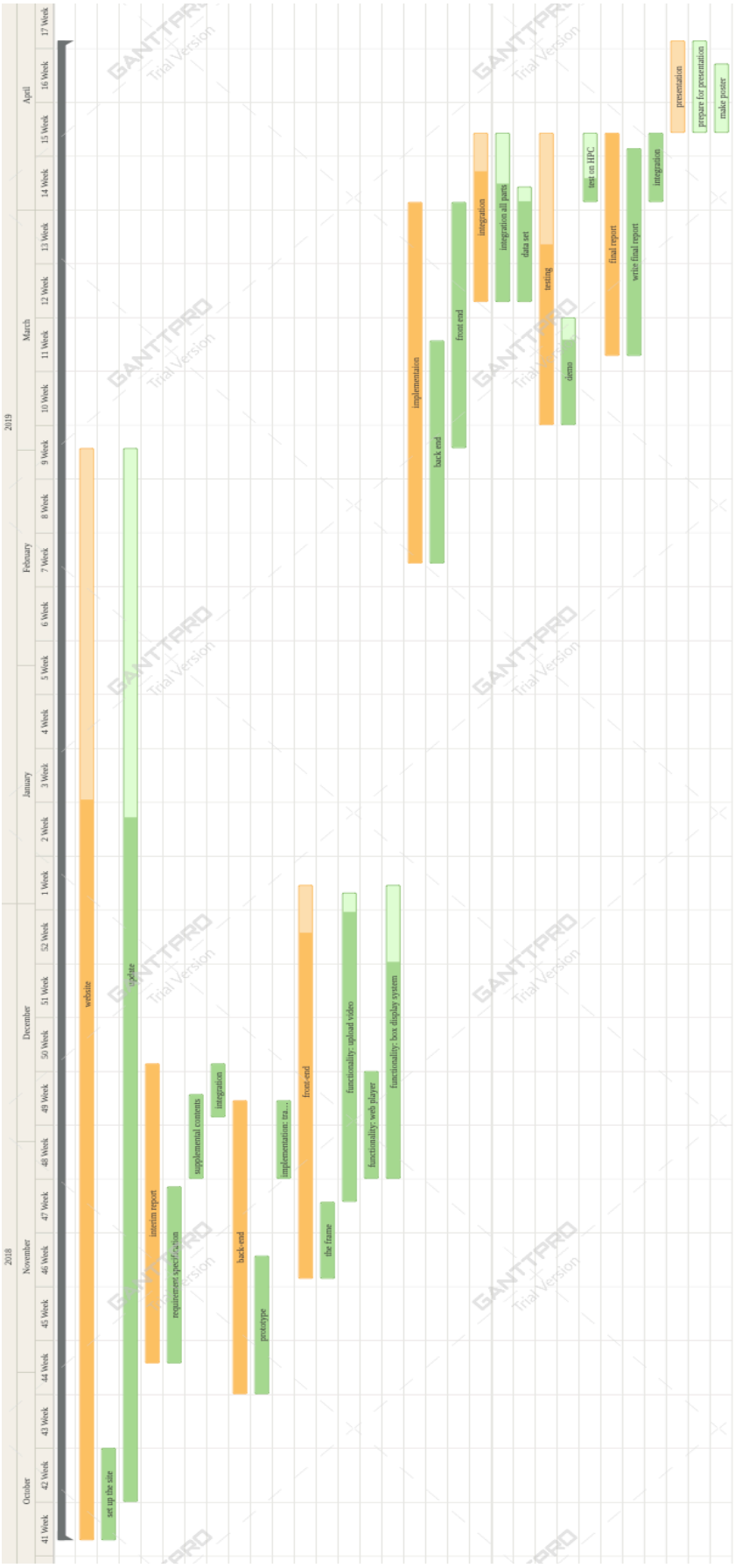
[12] Kaiming H., Xiangyu Z., Shaoqing R., Jian S., Deep Residual Learning for Image Recogenition(10 Dec 2015). Available at https://arxiv.org/abs/1512.03385

[13] Jun X.,Chia-Wen L., Ming-Ting Sun., Digital Video Transcoding, Proceedings of the IEEE (Volume: 93 , Issue: 1 , Jan. 2005) 84 - 97. DOI: 10.1109/JPROC.2004.839620 Available at: https://ieeexplore.ieee.org/document/1369700

[14] Dilip Kumar Prasad, Object Detection in Real Images(August 2010). Available at: https://arxiv.org/ftp/arxiv/papers/1302/1302.5189.pdf

[15] Messelodi S , Modena C M , Zanin M . A computer vision system for the detection and classification of vehicles at urban road intersections[J]. Pattern Analysis and Applications, 2005, 8(1-2):17-31.

Appendix A Gantt chart



Appendix B Minutes

Formal meeting 1

Date of meeting: 17th Oct.
Duration: 40min

Agenda items:
Responsibility of each team member
Responsibility of supervisor
The aim of the project

Discussion:
This is the first supervised group meeting, we confirmed what we will build for the project –a computer vision-based monitoring system, the supervisor guided us to take our responsibilities and also stated her role in the project.

Conclusion:
Start researching and collecting materials for building the software.

Unsupervised meeting 1

Date of meeting: 24th Oct.
Duration: 30min

Discussion:
We discussed the website and start working on it
Assign tasks to each member which part one should do for the website

Conclusion:
The prototype of the website will be done by next week

Formal meeting 2

Date of meeting: 26th Oct.
Duration: 1hr

Agenda items:
Everyone should hold the meeting in turns, as to enhance group work skills
Everyone should report what he/she has done last week
Tasks need to be divided into small tasks and deliver to each member

Discussion:
Group members reported the work they did for last week
Initially divide tasks into client part and server part

Conclusion:
Need to further distribute tasks and everyone should take record of what they have done.

Formal meeting 3
Date of meeting: 31st Oct.
Duration: 30min

Agenda items:
Decide what tools we should use
Introduce some useful resources to group members

Discussion:
Based on the project itself and advices from the supervisor, we decide to use YOLO for the training model, and ImageAI is also considered for usage, for model training data set, our supervisor advises us to use DOTA data set.

Conclusion:
The project becomes clearer since some concrete tools are settled.

Unsupervised meeting 2
Date of meeting: 31st Oct.
Duration: 1hr
In attendance:

Discussion:
How to collect and analyse user requirements? Interview people to collect user requirements.
Analyse functional and non-functional requirements, discuss what the software is able to do, such as to detect a car, to show relevant information.
Discuss the user interface, to implement a GUI or web application, how user is able to interact with the system.

Conclusion:
Requirements specification and prototype design, the preliminary steps of software engineering.

Formal meeting 4
Date of meeting: 9th Nov.
Duration: 1hr30min

Agenda items:
Must have systematical design of every aspect of the software.
Clarify the steps of building the software
Flow chart of tasks

Discussion:
After several weeks working on the software, there is a problem with the current workflow, we don't have systematical design and steps to follow, so the supervisor asks us to use the software engineering method and first design a prototype as soon as possible.
The workflow should be like: requirements → design → implementation → testing, however, so far, we just want to rush for a quick result.

Conclusion:

We should consider the software engineering process seriously or there will be a great chance of fail.

Unsupervised meeting 3

Date of meeting: 11th Nov.
Duration: 2hrs

Discussion:
The first thing is the website of the project's information, the website has almost done, and we need keep updating the information.
The main content of this meeting is about the report, we find some material and guides for writing reports and discussed about the structure and segments, how they should be divided and arranged in the report. We also distributed tasks for each member to start writing the report.

Conclusion:
This meeting is mainly about documentation of the project.

Formal meeting 5

Date of meeting: 16th Nov.
Duration: 1hr

Agenda items:
The report
The design

Discussion:
The design and architecture is very important in a proper project, every actual working and implementation will be based on these design, and there will be difference between the prototype and the final outcome, so we have to design the whole thing systematically and present it in the report clearly, in the meantime, we should be care of what may be changed during the process and how to handle it.

Conclusion:
We should reflect on the process of building a software.

Unsupervised meeting 4

Date of meeting: 18th Nov.
Duration: 2hrs

Discussion:
We still work on the report and integrate and revise the parts of the report wrote by group members word by word, to adjust the structure, enrich the content and refine the language.
We discussed about the HPC access.

Conclusion:

Revision of interim report

Formal meeting 6

Date of meeting: 3rd Dec.
Duration: 1hr

Agenda items:
Specific problems when implementing

Discussion:
unable to track when playing back, but it's ok for failing to implement, we have to show the effort we made. If user wants to speed up playing, the tracking information may be lost, it depends on the algorithm.
The output, should be the whole video or only the frames? The whole video may cause low efficiency, so implement the frame and put it on the screen.

Conclusion:
need more consideration on specific problems.

Unsupervised meeting 5

Date of meeting: 7th Dec.
Duration: 1hr

Discussion:
How may user download the processed video or the results? 1) the whole video 2) a format of file which acts like bullet curtain and displays the boxes.
How can the front end communicate with the back end? The front end needs to read boxes and send the click messages, should also consider the frame rate and appearance.

Conclusion:
Specific implementation may have problems

Formal meeting 7

Date of meeting: 13th Dec.
Duration: 1hr

Agenda items:
Data transmission
Sub-group should be divided on different works

Discussion:
What to use to transmit data? Socket, UDP, encoding, h.264 and data stream should be considered, to ensure transmission speed and avoid frame lost.
Can sacrifice accuracy to ensure speed.
There should be 2 sub-groups for data transmission and core algorithms

Conclusion:

It can be difficult to satisfy the requirement of accuracy and speed at the same time

---

Unsupervised meeting 6

Date of meeting: 17th Dec.
Duration: 2hrs

Discussion:
how to implement speed estimation? In the pre-processing part, what should be done for detection and tracking, how to implement the floating boxes?
In the back end, the controller should control which parts and how it is set in the system, what parts should it interact with? An idea is to make it an observer and observe any actions, such as the mouse click.

Conclusion:
Specific implementation problems about speed calculation, back end control and so on.

Formal meeting 8

Date of meeting: 21st Dec.
Duration: 2hr

Agenda items:
Citation and reference
Knowledge about the algorithms
Source and explanation

Discussion:
we should have clear knowledge about the algorithms we use in the project, instead of simple use them without knowing how they work.
We should read some papers to justify the algorithms we used in the project and state the source and explain it by ourselves base on the familiar of the tools.

Conclusion:
Need further study on the algorithms we use

Unsupervised meeting 7

Date of meeting:3rd Mar.
Duration: 30min

Discussion:
The report should include more diagrams
Communication between the front and the back end

Conclusion:
Some details on implementation