OPTICAL FLOW-BASED
VEHICLE DETECTION AND TRACKING

By

Ong Hen Ching

A REPORT

SUBMITTED TO

University Tunku Abdul Rahman

in partial fulfilment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONS)
INFORMATION SYSTEMS ENGINEERING

Faculty of Information and Communication Technology
(Perak Campus)

May 2010

DECLARATION OF ORIGINALITY

I declare that this report entitled "**OPTICAL FLOW-BASED VEHICLE DETECTION AND TRACKING**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : ______________________________

Name : ______________________________

Date : ______________________________

ACKNOWLEDGEMENT

I would like to give my special thanks to my supervisor, Mr. Tou Jing Yi for guide and support me throughout my research on FYP. He has provided me a chance to engage in computer vision field, which is a very interesting IT field that yet to be discovered. Without his guidance and support, I would not be able to deliver this project. I really appreciate his guidance and support. Thanks.

I would also like to thanks Mr. Ho Wing Teng for provided me very valuable comment and suggestion for my project. He has reviewed my project and suggested some solution that I have tried to seek for a long time. I also appreciate the data and information that he given to me, which is the video files that used in this research project. With his test video files, I managed to discover more fact about my project.

A very sincere thank for my friend, Alvin Yeo Zong Yun for commented on my works. He is also enrolled in project that is related to computer vision field, and always willing to share his knowledge with me, and helped my project a lot. He has also been a good partner in my university life. Million thanks for him.

A special thanks to my friend, Mr. Wang Hsin Jo for sharing his knowledge of computer programming. He even managed to provide me some information on computer vision field. And thanks my friends, Mr. Edward Cheang Kok Wei, Mr. Lim Sheng Peng and Mr.Yew Wei Hang, for lightening up my university life.

And finally, I would like to express my greatest thanks to my parents, Mr Ong Ban Sing and Mrs Chang Siew Hiang. They have been keep supporting me and encourage me throughout my university life. Their love, support, encouragement and patience, is and will be always the greatest treasure for my life.

ABSTRACT

Motion tracking is one of the most active research titles in computer vision field. Tracking of motion can be informative and valuable, and motion information is very useful for traffic surveillance system. Many proposed motion tracking techniques involves using template matching, blob tracking and contour tracking. A famous motion tracking and estimation technique, optical flow, however, is not being widely used and tested for the practicability on traffic surveillance system. Thus, to analyze the reliability and practicability of it, this research project proposed the idea of implementing optical flow in traffic surveillance system, and will evaluate its performance. Another aim of this research project is to assess the ability of OpenCV, a famous open source computer library, in traffic surveillance system. Three experiments were carried out in this research project to test the practicability of optical flow, and also to find the fact about optical flow. Also background subtraction technique available in OpenCV was assessed to assist on problem identification. The results of tracking using Lucas-Kanade (LK) optical flow is proving that optical flow is a great technique to track the motion of moving object, and has great potential to implement it into traffic surveillance system.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

2D	Two dimension(al)
LK	Lucas-Kanade
MoG	Mixture of Gaussians
OpenCV	Open Source Computer Vision

Chapter 1: Introduction

1.1 Background

In computer vision field, motion can be defined as the physical movement of pixels. Computer vision or some may refer to machine vision, means to enable computer to see, where see in here not simply refer to obtain image only, but also able to extract out the information that the image can provide. Given an image, human can easily identify and get most of the information out, such as background and foreground, motions, and events. However, computer do not possess such ability, and viewing an image as a matrix instead.

Vehicle tracking is one of the popular topics in computer vision field. It is also considered as one of the challenges in computer vision field, as motion tracking can be quite tricky. Unlike human, computer does not possess self-learning ability, thus also make them unable to track the motion as “smart” as human can. To overcome this, many researches has been done, and solution has been made. These solutions may not yet practicable, but still enough to prove that it is possible to tell computer how to track motions.

Many proposed solutions involve the combination of background and foreground subtraction and the template matching technique to keep track of the vehicles. The background subtraction is use to extract out the foreground object, and the template matching technique is use to find the object, which is vehicle in vehicle tracking case, and manage to track the object in next frame. The drawback for such combination method is template matching will require frequent update on template, and changing of the view of angle of camera will require adjustment also. The adaptability can be varying. To deal with such limitation, this project is focus on assessment of an alternative motion tracking method, optical flow method, which is believed to have better adaptability level compared with template matching and tracking.

1.2 Problem Statement

For many years, many video surveillance system implemented template or pattern matching to track the moving object. The main reason for implementing template or pattern matching to perform object tracking is because this method requires less resource than optical flow motion tracking. Simple template or pattern matching operation can produce acceptable result in object tracking. However, template or pattern matching requires sample of pattern or template before it can track any object. Object tracking can be difficult if the surface of the object do not remain same from frame to frame, such as reflection of light.

1.3 Motivation

Optical flow has varieties types of algorithms, and the very famous among them is Lucas-Kanade (LK) method. Originally proposed in 1981 (Lucas and Kanade, 1981), optical flow can compute the motion and its flow and producing quite accurate results theoretically. Yet, very few of traffic surveillance system implement this method, as optical flow was not easy to be implemented, and requires special optimized and designed system (Collins *et al.*, 2000). Still, the accuracy of optical flow, and the theory behind make it a considerable and arguable solution for motion tracking. Fortunately, a numbered of research and improvements on optical flow has been made. Therefore it would be necessary to re-evaluate its practicability and performance, which is one of the objectives in this research project.

OpenCV is becoming widely known in computer vision field for its library that mainly built for computer vision computation. Another notable feature is the library is cross-platform, which is usable in Windows, Linux and MacOS. Such high portability reduces burdens and works if port of system is needed. Nowadays, OpenCV has huge collection of popular computer vision computation algorithm, and optical flow is one of them.

1.4 Objectives

The objectives of this project are:

(1) To evaluate and assess practicability and ability of OpenCV in traffic surveillance system implementation.

(2) To introduce optical flow into traffic surveillance system to track vehicle movements.

(3) To develop a prototype of vehicle tracking system by implementing optical flow as major motion detection algorithm.

1.5 Scope

1.5.1 Functionalities

(1) Provide flow and direction of motion of moving vehicles

Vehicle moving on road usually only have one direction. In broader road, the road might contain two lanes where they have opposite direction, or actually same direction. Furthermore, flow of moving vehicle can also determine the traffic status of the road. All the motion detection will be done automatically, or with minimum interaction from user.

(2) Keep track of the moving vehicles

The system will be able to keep track of every moving vehicle once they are identified. Tracking vehicles can provide informative data such as their origin location, destination and moving behaviour. The system will also automatically track them once they are captured by camera.

(3) Able to export every data captured

Since this is one of the major components of complete traffic surveillance system, export of data is needed, to allow other components able to import the data and perform further processing.

1.5.2 Assumptions

The first assumption in this research project is the video stream or video is captured from static camera. This is because motion of camera itself will also affect the result of the system. Different movement of camera requires different motion tracking calculation approaches. Small motion can be ignored by applying thresholding to cancel out all the non significant motion, leaving significant motion behind. However this is not helpful if the motion of background is as significant as moving objects are.

Another assumption for this project is brightness will remain consistence from time to time, or only change smoothly and slowly from time to time. Huge increase on brightness can affect how optical flow designed to run and compute, result in poor detection and tracking on moving objects. Optical flow can still able to track movement if the changes of brightness are smooth and slow from time to time.

Chapter 2: Literature Review

2.1 Motion

Definition of motion can be unclear and confusing due to variety version of explanation and definition. One of the most common explanations of motion is the change in position of object over time to time. In physics, motion is further described in term of velocity, acceleration (rate of change of velocity over time), displacement (shortest distance of initial position to final position), and time (Zhang *et al.*). In computer vision, this explanation is further extended to change in pixel of a set of picture sequence over time to time.

However, computer natively does not have vision ability as human does. Without further processing, data or information, which motion of object (includes motion's properties) is being one of them, is not extractable. How computer percept our world is different from us, human does. Every scene that computer captured is digitally stored in 2D array, thus losing several information that only can obtained by third dimension such as depth of object, distance and etc. However if motion information is successfully extracted, it can be very useful for further processing such as shape of objects, speed or function.

Extraction of motion information is not as easy as just comparing image sequences. For example, in reality, when we move our eyes, it will result in getting different view of scene. Human knows there is no motion on object, instead is the motion of eyes that causing difference on scene captured. However, computer itself does not know the truth, as what it knows from image sequences captured is just simply change on pixels of images.

2.2 Background Subtraction Techniques

In current computer vision field, object segmentation (object extraction) is commonly highly related with motion tracking and analysis. Before any motion tracking can be performed, object should be clearly identified, and then apply tracking on them. One of the most common techniques (Mcivor, 2000) in motion detection study to isolate the object from the rest of the object is background subtraction. This has applied to several traffic surveillance systems too (Gupte *et al.*, 2002) (Cheung and Kamath, 2004).

Background subtraction, in simplest explanation, is to subtract out the background, which is always static (remain unchanged for a period of time), thus leaving active object for further analysis. For most of the video surveillance system, source of image sequences, if often captured by surveillance video camera that installed in a place and remained static when the system is operating. Background subtraction works very well in this situation, as identification of background became much easier due to the static background.

Background subtraction itself do not directly extract out motion information, instead, it helps to identify out the object in scene by separating foreground and background. Foreground is normally the interesting object that user wishes to analyze, while background normally always boring and repetitive, which is not so meaningful. Since background is defined as static scene (pixels), foreground item can be explained as object that possibly moving, thus making them different from background.

Input: $\mathbf{Image}_{\mathbf{t}}$ and $\mathbf{Image}_{\mathbf{t-1}}$, where $_{\mathbf{t}}$ representing time, $_{\mathbf{t-1}}$ representing previous time

Output: $\mathbf{Image}_{\mathbf{out}}$

For every pixel **[x,y]** in $\mathbf{Image}_{\mathbf{t}}$ and $\mathbf{Image}_{\mathbf{t-1}}$,

Image out [x,y] = $\mathbf{Image}_{\mathbf{t}}$**[x,y]** – $\mathbf{Image}_{\mathbf{t-1}}$**[x,y]**

If **Image out [x,y]** > threshold value, then **Image out [x,y]** = 1

Else **Image out [x,y]** = 0

Figure 2.1 Simple explanation on background subtraction algorithm

Most of the current surveillance system is applying this technique to help them identify interest object (usually moving objects), due to its characteristics and performance. Accuracy of this technique can be varying, as this technique highly relies on background model it obtained or learned. In other word, background model is very critical for background subtraction technique. If good background model is provided, the accuracy of object segmentation can be very high. But in opposite, the accuracy could be very bad if background model obtained or learned is not strong enough.

2.2.1 Absolute Differential Motion Detection

Also known as frame differencing, differential motion detection is one of the most basic yet sometime effective techniques. Motion of an object can be simply defined as the difference of initial position and final position of the object in two images. Assume that P_{t1} is the initial position, and P_{t2} is the final position, if P_{t1} is not equal to P_{t2}, thus difference occurred, and then the difference may tell that the object has been moved from one position to another position.

Given two images that captured during different time (t & t-1), differential motion analysis methods will finds the difference between two images, which possibly is the motion of objects (Sonka *et al.*, 2008).

This approach can sometime be surprisingly effective and efficient in certain circumstances. In different circumstances, this approach will produce different interesting result. To explain this, two example circumstances will be tested.

Circumstance A is taking two images from an image sequence, which image sequence is holding properties of low frame per-second (FPS). Absolute frame differencing can provide clear initial position and final position of the object, but do not tell the trail of the moving object.

Circumstance B is having two images from an image sequences that have a high FPS value. Due to the minor difference between frame $_{t-1}$ and frame $_{t}$, this approaches captured the contour of the object.


(a) Frame $_{t-1}$


(b) Frame $_{t}$


(c) Result

Figure 2.2 Absolute frame differencing in Circumstance A


(a) Frame $_{t-1}$


(b) Frame $_{t}$

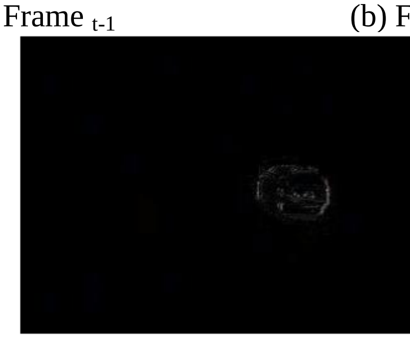

(c) Result

Figure 2.3 Absolute frame differencing in Circumstance B

2.2.2 Advanced Background Subtraction

The drawback of absolute frame differencing is background must remain static over time. As it just simply comparing the current frame with the previous frame, the system itself actually do not know any background model. In reality, many background scenes contain complicated moving things, such as trees waving is also result in pixel changing, which will greatly affect the accuracy of absolute frame differencing (Bradski and Kaehler, 2008). Providing the background model is very helpful for producing better background subtraction result.

(a) Background model

(b) Current frame

(c) Result

Figure 2.4 Background subtraction by providing static background model

By providing a good background model, even when the object is not moving, it still can be detected by subtracting (comparing) every pixel on current frame with the background model provided.

2.2.3 Background Modelling

Provide a constant background to perform background subtraction would only works in certain circumstance. When dealing with environment that brightness is changing from time to time or lighting is dynamic, constant background subtraction will not works well. Such constraints will greatly reduce the usability and practicability of the video surveillance system, as video surveillance system always deals with dynamic brightness and lighting environment, such as outdoor.

A proper solution for such environment is to model background and maintain the background frame dynamically (Sonka *et al.*, 2008). Background model will be collected after *n* frames is collected, and taken the median intensity of them. By doing so, lighting that changes slowly and steadily can be updated into background model.

Input: **n Frame, Frame $_{n+1}$,** where **n** representing period (time)
Output: Background model

(1) Get **n Frames** and get the median intensity
(2) Get **Frames** $_{n+1}$, differentiate **Frames** $_{n}$ & **Frame** $_{n+1}$
(3) Remove noises (thresholding)
(4) Remove small regions (which is usually false foreground object during background subtraction)
(5) Update median intensity measurement
(6) Repeat Step 2

Figure 2.5 Simple explanation on background modelling and maintaining algorithm

In algorithm explained in figure 2.5, value *n* is the key factor to determine the quality of background model. Algorithm in figure 2.5 assumes that value *n* is remaining constant over whole process. By holding this assumption, algorithm will then is vulnerable to various objects moving speed. Setting the value *n* high may cause slow moving object accidentally updated into background model, and in reverse, low value of *n* will causes many non-background object false recognized as background.

The better background modelling algorithm to tackle such problem is Mixture of Gaussian (Stauffer and Grimson, 1999). This algorithm proposes that every single pixel is modelled as a mixture of Gaussians, and they may be representing either foreground or background of the scene. Weights is taken into consideration, and updated at every new frame.

2.3 Motion Tracking

After object has been identified, further processing can be applied to track and obtain the motion information on that object.

2.3.1 Motion Field

In computer vision, motion field is defined as a 2D array of 2D vectors representing the motion of 3D scene points (Shapiro and Stockman, 2001). In other words, it is the projection of 3D motion vectors on image plane.

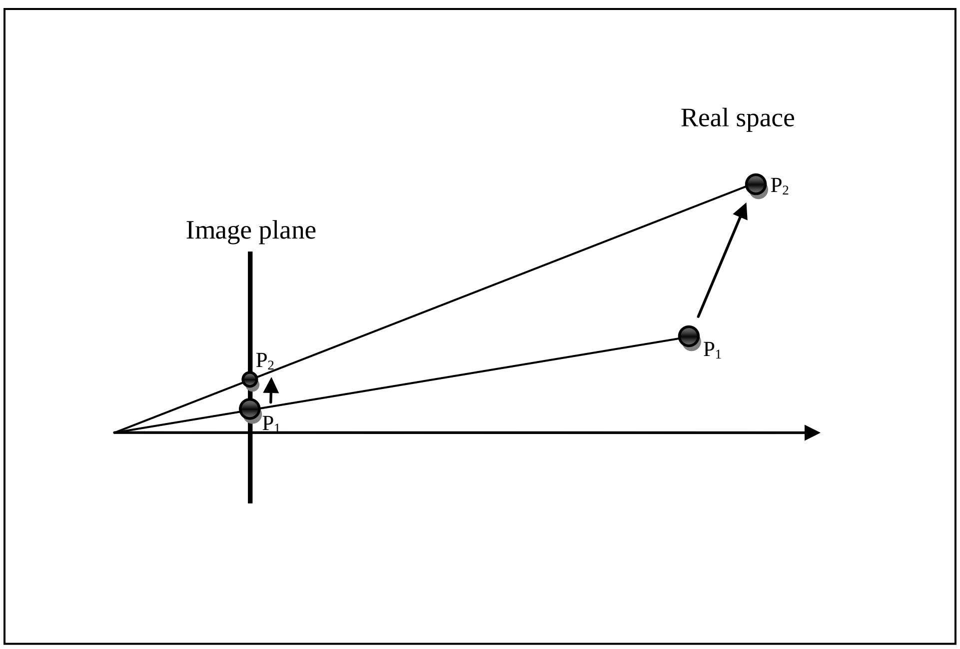


Figure 2.6 3D motion projections over 2D image plane

Since the captured scene is actually a 2D image, which loose a dimension, motion field in more appropriate explanation, is not directly computable and measureable, but can only approximate it.

2.3.2 Optical Flow Computation

This method detects the motion by analyzing the motion of pixels, either in bulk or in individual. It reflects the image changes due to motion during a time interval *dt*.

The idea of optical flow can be said derived from how human observe motion and objects. Human able to determine motion in the scene captured by analyze gradient, brightness, lights reflected and etc (Lee and Kalmus, 1980). However optical flow computation do not provide functionality to automatically classifying and recognize moving objects, though it can be done after optical flow computation has been done to observe what object is moving and probably will moving (estimation).

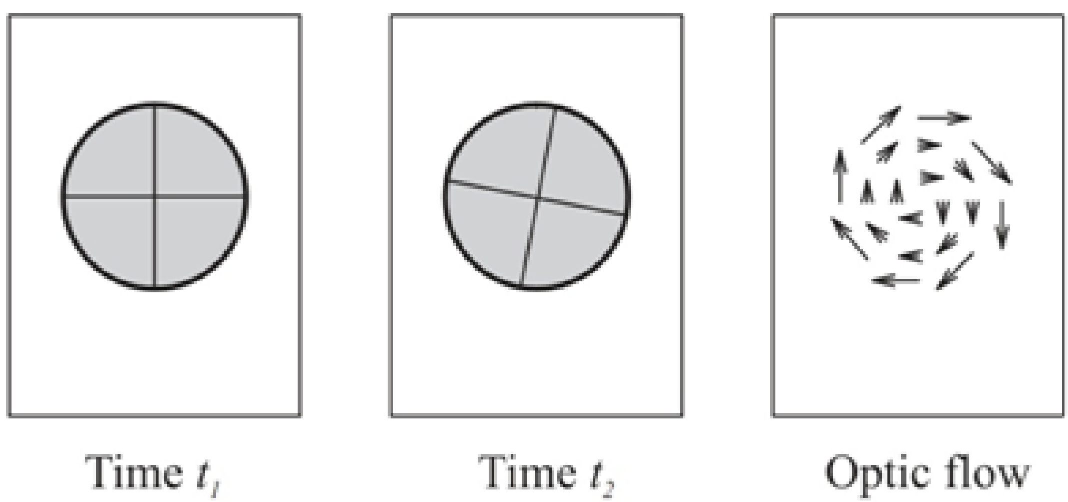
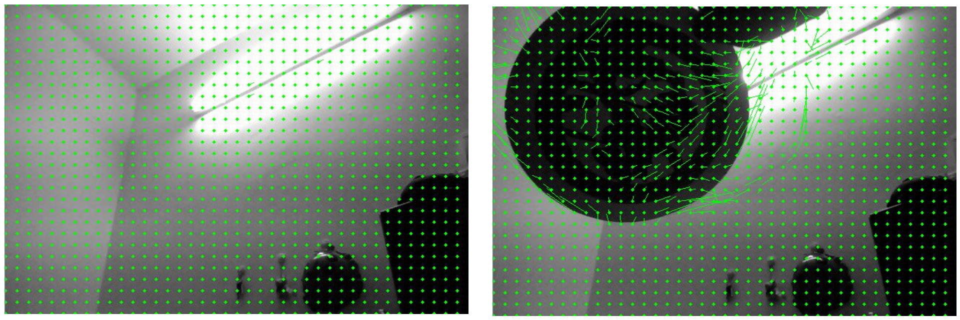


Figure 2.7 Illustrated optical flow. (Sonka *et al.*, 2008).

Optical flow has two main methods for tracking, which are dense optical flow, and sparse optical flow.

Dense optical flow suggested that every pixel in an image should be associated a velocity value, and compute them to get the flow of motion.

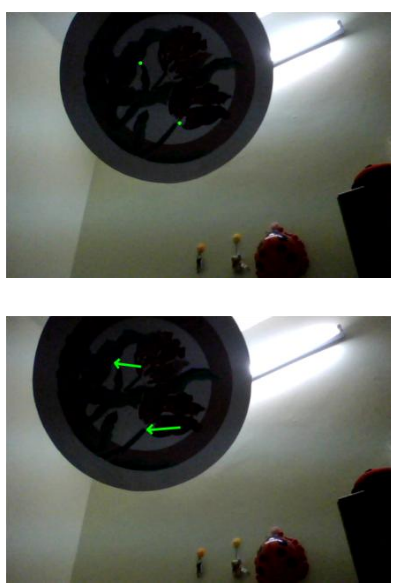


(a) Frame $_{n}$ **(b) Frame** $_{n+1}$

Figure 2.8 Example of dense optical flow using Farnebäck algorithm

Dense optical flow can provide visualize form of motion field, as shown in figure 2.6. Appearance of the object in figure 2.6(b) is detected and calculated, showing the motion field of scene. However this can be very resource hogging, and harden the process to extract out the certain object motion (if whole image is actually moving).

Sparse optical flow will only tracking on point of interest (or subset of points), thus able to produce robust, reliable and direct informative result.



(a) Frame $_{n}$

(b) Frame $_{n+1}$

Figure 2.9 Example of sparse optical flow using LK method

2.3.2.1 Lucas-Kanade Method (LK-Method)

Originally proposed in 1981 by Bruce D. Lucas and Takeo Kanade (1981), LK optical flow method has become one of the most frequent used sparse optical flow techniques. The reason of this optical flow technique has been widely used is because it can easily applied to a subset of the point in the input image. For an example, in dense optical flow techniques, optical flow will compute every pixels velocity even user only request for tracking one vehicle motion. In LK-method, after assigning the object as the interest object, LK-method will only compute on the region that has been specified (in this example, the vehicle).

LK-method made three assumptions to ensure the effectiveness of the algorithm (Bradski and Kaehler, 2008).

(1) Brightness constancy

Interest point from the image should not change from frame to frame, as change in brightness is actually changing its appearance.

(2) Temporal persistence ("small movement")

Motion is happen slowly in time, or smoothly over frame to frame. In other word, it means the difference of position of object should not be too much.

(3) Spatial coherence

Neighbouring point of the interest points should have similar motion and staying neighbour from frame to frame.

2.4 OpenCV

OpenCV, abbreviation for Open Source Computer Vision, is an open source computer vision library that originally developed by Intel (Bradski and Kaehler, 2008), and now being actively supported by Willow Garage. It contains numerous number of computer vision library that is open and free for everyone. The computer vision function includes image processing, face detection, object segmentation, user interface and much more.

OpenCV is well known for its portability, and support wide range of C and C++ compilers. Natively, it supports Windows, Linux, and Mac OS X on Intel and AMD architecture (Bradski and Kaehler, 2008). Now it also supports several popular operating systems such as Android, iOS, and FreeBSD. This made the system developed by using OpenCV library even more flexible in term of portability.

Improvement of OpenCV can be said dramatic and rapid. In year 2006, official released the first release version 1.0. Now the latest version of OpenCV is version 2.2, which was released on December, 2010. Numerous improvements have been made, such as improvement on background subtraction method by changing the algorithm to MoGs algorithms. Performance and accuracy of most of the computer vision function has also increased for past 5 years, making it a great computer vision library today.

Chapter 3: Methodology

3.1 Introduction

In this research, most of the computer vision tasks are supported by using OpenCV computer library. OpenCV consists of a lot of image processing related function and method. In this chapter, the major software architecture will be discussed, and methodology of the system will be explained.

3.2 Design of Software

3.2.1 Vehicle Tracking System Overview

The software structure for video surveillance system is not always consistent. Still, there are some major components or function that is commonly seen and used.

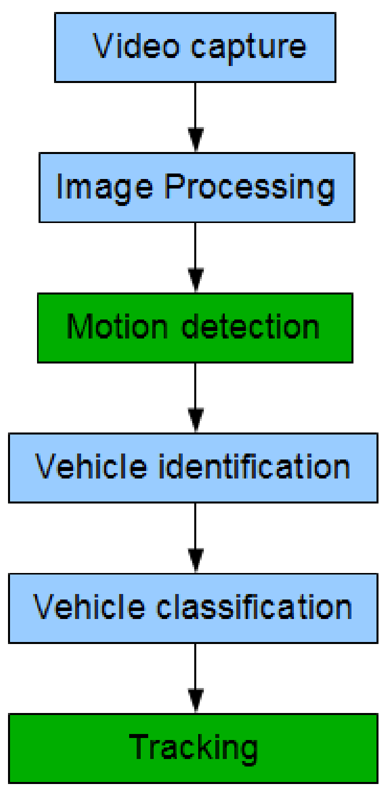


Figure 3.1 Simple diagram of vehicle tracking system component and process flow

In this project, the research area is concentrate on motion detection and tracking, which is highlighted in green in figure 3.1. The vehicle identification and classification process will be briefly discussed, but does not include in the project primary objectives.

Motion detection and tracking process is also consists of many sub component and function. Different approaches of motion detection and tracking may have different component design. The most frequent used approach to detect motion in image sequences is background subtraction. In this project, no specify motion detection technique will be used. Instead, any possible solution available in OpenCV library that is capable of achieving the main objectives, which is to track the vehicle, is tested and evaluate.

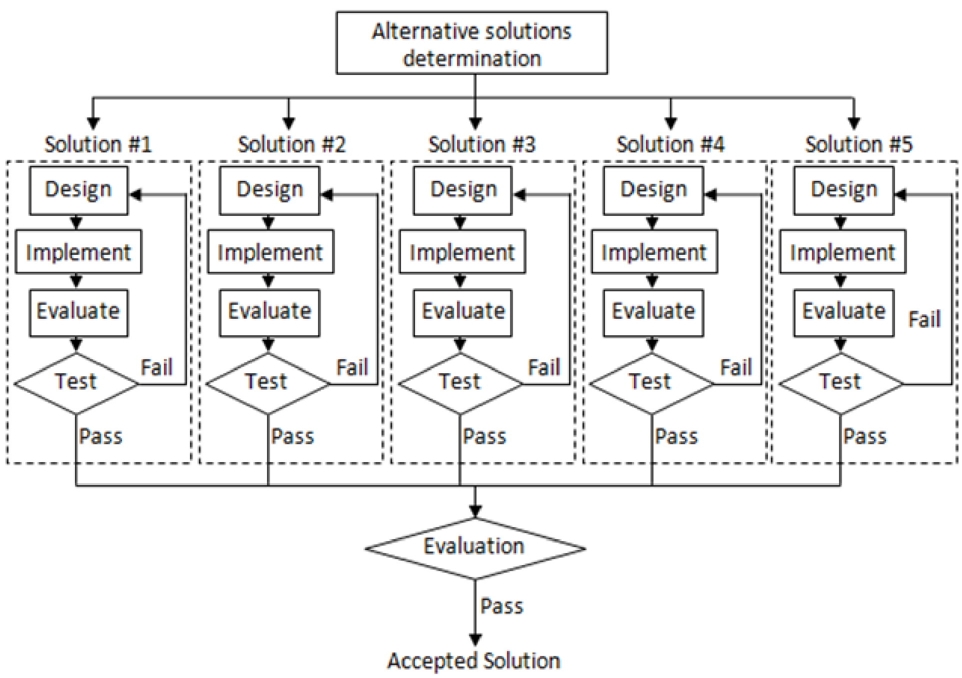


Figure 3.2 Flow diagram explains methodology to select and evaluate solution

3.2.2 Motion Detection and Tracking Design Overview

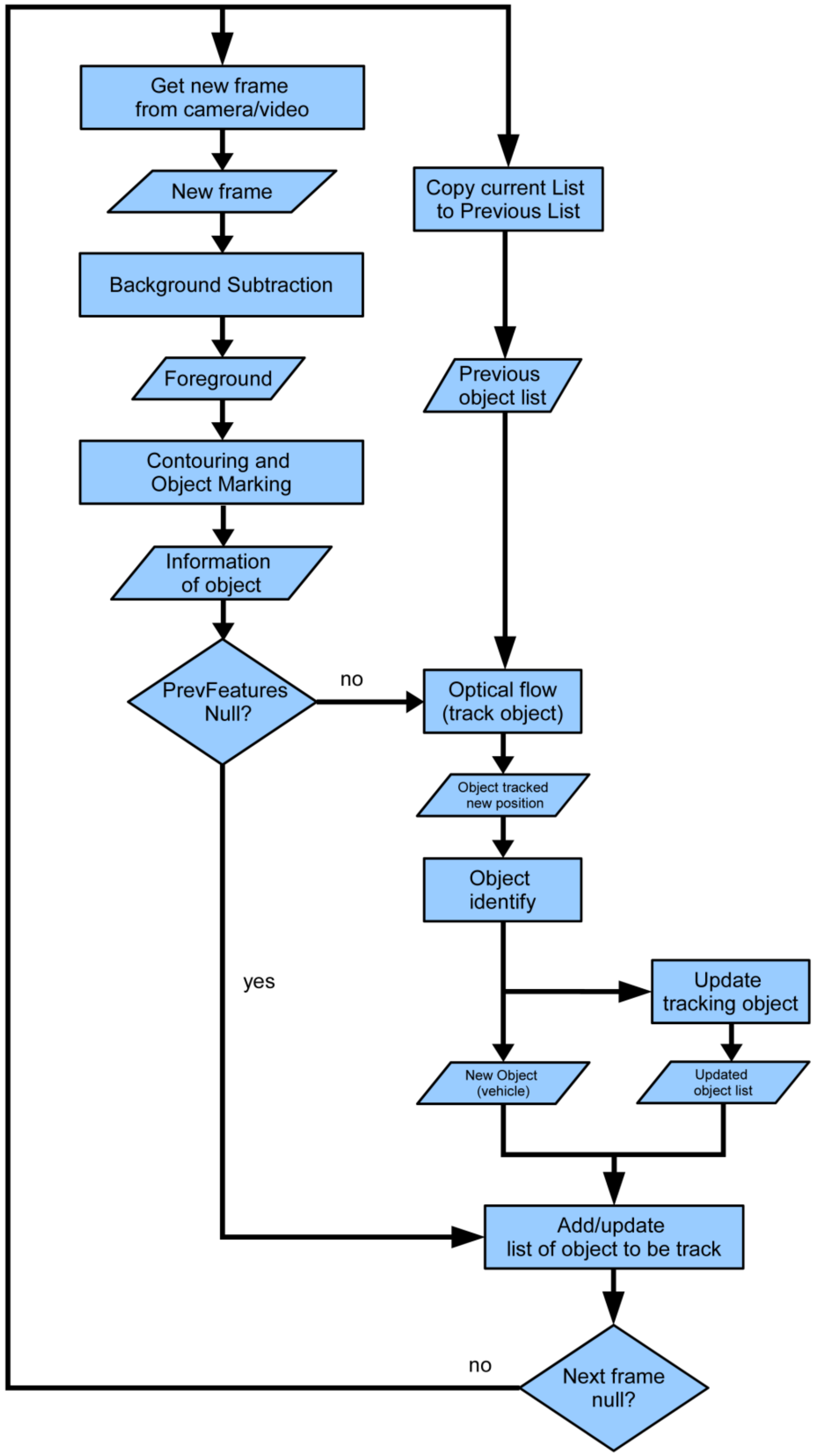


Figure 3.3 Flow chart of motion detection and tracking

After several testing on techniques available in OpenCV, the workable solution is found. The motion detection and tracking will consist of 3 major techniques:

(1) Background subtraction

(2) Contouring and object marking

(3) LK optical flow method

Background subtraction will help the system to extract out the foreground object. Contouring and object marking will get the raw foreground from background subtraction function, and then apply contours and object marking on any object found. LK optical flow will hold the biggest responsibility to track the object, and provide system some motion information of object such as direction of motion and trace of motion.

In between motion detection and tracking (background subtraction and LK optical flow tracking), there is supposed to be vehicle identification and classification process before any tracking can be performed. While this research project is focus on motion tracking using optical flow approach, the vehicle identification and classification process will be replaced by contouring and object marking. Contouring and object marking will mark and analyze the object to provide better information for further operation.

After information of detected object is gathered, it will compare with the optical flow detected object, to see whether the new object detected from background subtraction is already being tracked by optical flow. The object will be deleted if it is already being tracked by optical flow.

This project will be implemented by using C++ programming language. The main reason of using C++ as the programming language for this project is the compatibility with OpenCV. C++ is one of the language that being supported by OpenCV. The other two programming languages are C and Python. Python interface in OpenCV is still being developed and has less support than C++.

3.3 Motion Detection

3.3.1 Background Subtraction

Before any motion can be tracked, system must first extract out the object. The common technique to obtain the object from 2D image is background subtraction. The main reason of using background subtraction is background subtraction works well if the background is static for long time. In traffic surveillance system, camera is often remaining static. By taking this characteristic, background subtraction should works well the most.

OpenCV provide several techniques to remove static background from frame captured. One of the techniques is MoG background modelling. MoG function provided in OpenCV has been updated and revised for several time. In the previous version of OpenCV (version 2.1, updated in April, 2010), MoG function has been rewritten for better accuracy and performance. OpenCV's MoG function is based on an improved version of MoG algorithm (KaewTraKulPong and Bowden, 2001). Although in OpenCV version 2.2, the shadow detection proposed in the journal is not implemented yet, it doesn't affect the result much as the shadow will be considered part of the object as well.

Modelling of background using MoG can be performed by calling *cvCreateGaussianBGModel*. The *cvCreateGaussianBGModel* require at least one parameter, which is the current frame captured from camera or image sequences (video).

```
cvCreateGaussianBGModel (
        IplImage*                       frame,
        CvGaussBGStatModelParams*       param      = NULL
);
```

Code snippet 3.1 OpenCV MoG background modeling function: cvCreateGaussianBGModel

The second parameter, *param* is optional, and will be auto declared with default values if NULL is passed. The default value for *param* is shown in code snippet 3.2.

```
#define    CV_BGFG_MOG_BACKGROUND_THRESHOLD    0.7
#define    CV_BGFG_MOG_STD_THRESHOLD           2.5
#define    CV_BGFG_MOG_WINDOW_SIZE             20
#define    CV_BGFG_MOG_NGAUSSIANS              5
#define    CV_BGFG_MOG_WEIGHT_INIT             0.05
#define    CV_BGFG_MOG_SIGMA_INIT              30
#define    CV_BGFG_MOG_MINAREA                 15.f
```

Code snippet 3.2 Default value of CvGaussBGStatModelParams

Background subtraction will only responsible for extracting out the foreground object. In other word, it does not have ability to identify whether the object found is new object or old object that already found in previous frames. The identification of new or presenting object will be later carried out by comparing the latest foreground object list and recorded object list.

3.3.2 Contouring and Object Marking

After background subtraction successfully extract out the foreground, it will draw them into a binary image. However in this state, the system still does not able to recognize, or identify the object in the binary image. The system cannot interpret the white area of pixels in the binary image yet.

To make these objects more recognizable and informative to the system, these objects should be marked and recorded. Contouring is the process of marking out the edge of the object, making it more recognizable and informative. In computer vision field, edge of the object is usually explained as the area of pixels that is contrast with its neighbour pixels or area of pixels motion (Shapiro and Stockman, 2001). Provided binary image that object is coloured in white, process of contouring can be quite straightforward. It is suggested that the binary image provided should contain no or minimal false object, as contouring process do not hold responsibility for filtering out false object.

Function of finding and connecting contour in OpenCV is very useful, as it also will provide additional information of object such as size, centre point, and points that can later use to draw bounding box that will bound to the object. The centre point of object is very useful for the next process, object tracking, as it can act as a good feature point for object.

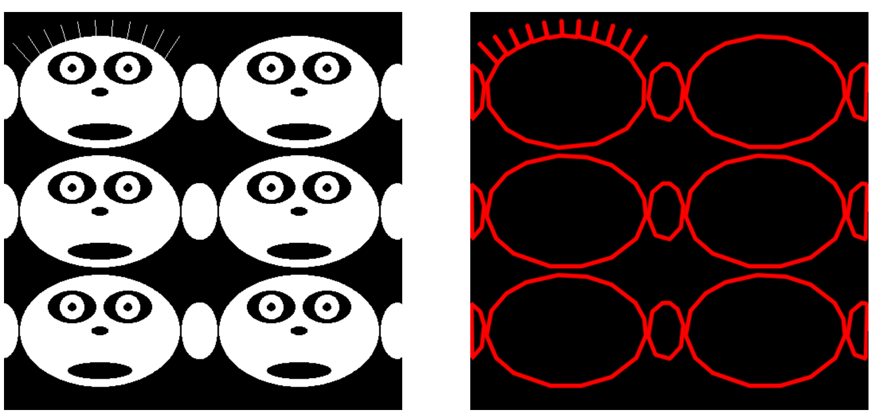
Figure 3.4 Output of contouring and object marking

Contouring function in OpenCV is using data structure that is similar to linked list to store the contour node, where the contour node in here usually means curve or corner of the object. Before any contours can be draw, function cvFindContours should be performed to get the list to the contour node.

```
int  cvFindContours(
     IplImage*                  img,
     CvMemStorage*              storage,
     CvSeq**                    firstContour,
     int                        headerSize     = sizeof(CvContour),
     CvContourRetrievalMode     mode,          = CV_RETR_LIST,
     CvChainApproxMethod        method         = CV_CHAIN_APPROX_SIMPLE
);
```

Code Snippet 3.3 OpenCV find contours function: cvFindContours

The way of contouring may be differ depend on the value of parameter *mode* passed into the function. The *mode* variable defines the level of retrieval, which is also understandable as depth level of contour will be retrieved and presented. The higher the level is, the more interior contour will be discovered and presented. As the details of vehicle are not required and informative for this research project, the value of parameter mode will set to only 1 (defined as CV_RETR_LIST), which will retrieve only extreme outer contour.

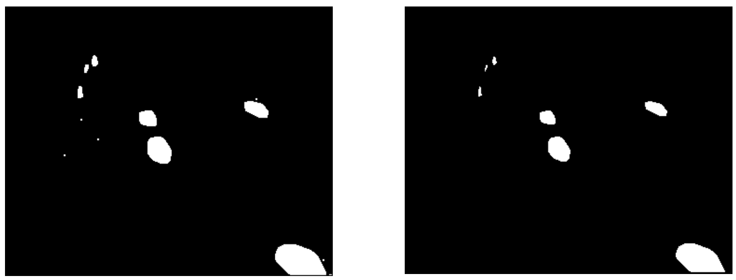


(a) Source (b) Result

Figure 3.5 Input and output of result when retrieval level is 1

3.3.3 Image Morphology

Background subtraction rarely gives clean output. Most of the time, background subtraction will made some error identification, and leaving some small area of pixel which is wrongly identified as foreground object. “Tiny object” in most situations, is merely just mistakes (or error) of background subtraction. In order to eliminate them, cvErode can be use to perform erosion on image passed in. Erosion is image morphology technique that will reduce the object region (or size of the object). Result of erosion is shown in figure 3.6 (b).



(a) Before erosion morphology (b) After erosion morphology

Figure 3.6 Binary image after erosion to remove “tiny object”

cvErode requires four parameters to perform erosion morphology, which are stated as:

```
void cvErode (
        IplImage*          source,
        IplImage*          dest,
        IplConvKernel*     B             = NULL,
        int                iterations    = 1
);
```

Code snippet 3.4 OpenCV erosion morphology function: cvErode

Parameter *source* act as the input for the function, and *dest* is the output binary image after erosion processing is complete. *B* will be a 3-by-3 kernel with anchor at its centre if NULL value is passed into the function. Parameter *iterations* will determine how many times the function will repeat in a single call. The larger the value of *iterations* is, the more regions will get reduced.

3.4 Motion Tracking

3.4.1 LK Optical Flow

Background subtraction itself does not perform any tracking operation on object extracted. In this methodology, optical flow is chosen to track the object, as well as to retrieve its motion information such as direction of moving object and motion history (path).

As a sparse optical flow method, LK optical flow will require a set of features as the input before it can perform any operation. The algorithm itself does not perform any feature detection. OpenCV has an additional function *cvGoodFeaturesToTrack*, to help user automatically detect the feature rather than require them to input them manually. However, the features computed from *cvGoodFeaturesToTrack* may be not representing any object, as *cvGoodFeaturesToTrack* is taking any pixel (or area of pixel) that is good to identify and track, rather than identifying features that is good to representing an object.

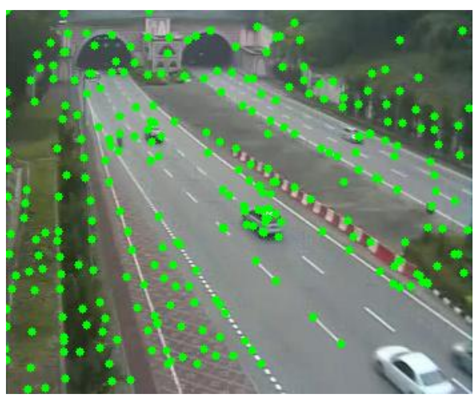


Figure 3.7 Features found using cvGoodFeaturesToTrack

To provide features that are capable of representing and identifying object in the scene, the result from contouring and object marking is required. In this project, the centre point of the object is used as the feature to representing and identifying object. Centre point of object can be obtained during the process of contouring and object marking, thus reduce the work load of seek for good feature in the object.

Figure 3.8 Centre point of object as features

OpenCV have two LK optical flow function, which are *cvCalcOpticalFlowLK* and *cvCalcOpticalFlowPyrLK*. The difference between them is the latter additionally uses image pyramid to improve the quality of result. Image pyramid is a collection of image that the entire image inside pyramid is arising from a single original image.

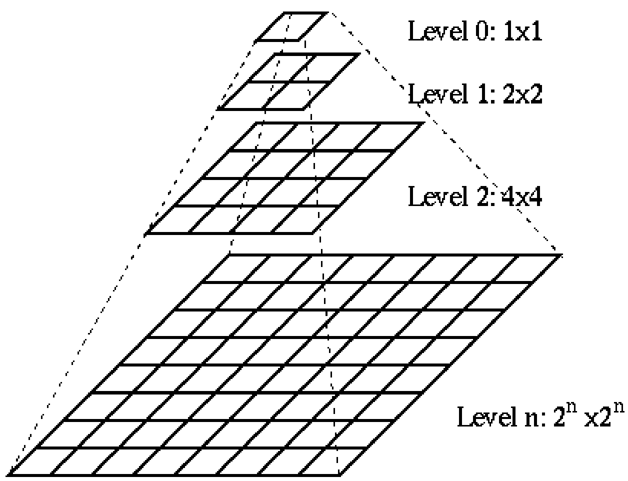


Figure 3.9 Illustration of image pyramid (Wang, 2004)

As discussed in 2.3.2.1, LK optical flow method is rests on three assumptions:

(1) Brightness constancy

(2) Temporal persistence (movement should be small)

(3) Spatial coherence

Reason of making such assumption is because LK optical flow method relies on small local window to track the sparse content. For every feature, LK optical flow will only perform calculation on some of its nearby area, which is known as local window, and is usually small. While this brings better performance compared with optical flow (dense optical flow) that relies on global information, another problem derived. Any movement that moves larger than the window size will make the algorithm failed to find its destination.

To make the algorithm more robust and adaptable, image pyramid is suggested. Every layer of image pyramid will be taken into optical flow to find the feature, and using same local window, find the motion if possible. As different layer of image pyramid have different size but share a same local window size, it is possible to mitigate the limitation of sparse optical flow.

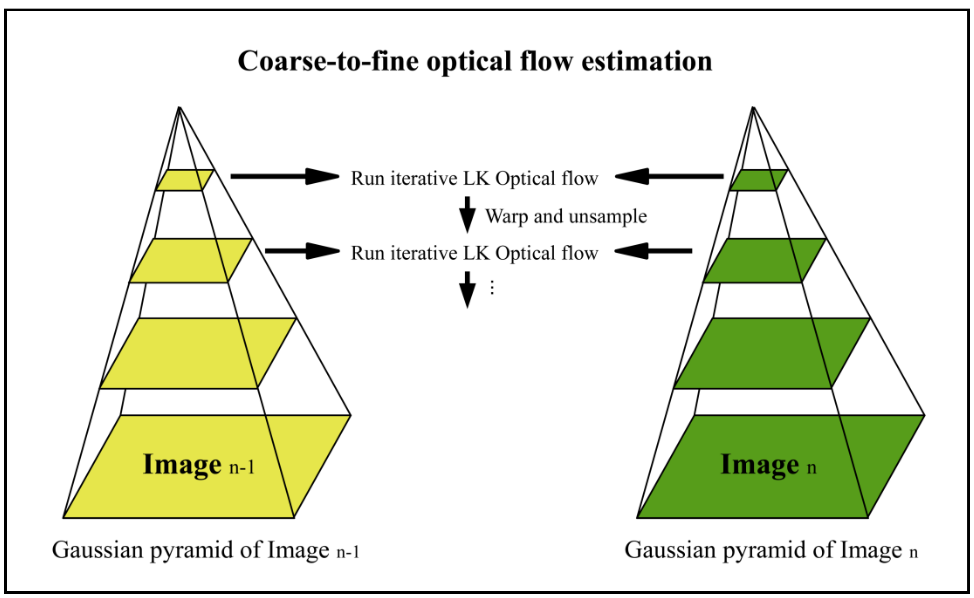


Figure 3.10 Illustration of Pyramid LK Optical Flow (Bradski and Kaehler, 2008)

The *cvCalcOpticalFlowPyrLK* accepted numbered of parameters. The structure of *cvCalcOpticalFlowPyrLK* is shown in code snippet 3.5.

```
void cvCalcOpticalFlowPyrLK(
        const CvArr*        imgA,
        const CvArr*        imgB,
        CvArr*              pyrA,
        CvArr*              pyrB,
        CvPoint2D32f*       featuresA,
        CvPoint2D32f        featuresB,
        int                 count,
        CvSize              winSize,
        int                 level,
        char*               status,
        float*              track_error,
        CvTermCriteria      criteria,
        int                 flags
);
```

Code Snippet 3.5 Structure of cvCalcOpticalFlowPyrLK

The *imgA* and *imgB* is the image frame to the current frame (frame $_{n}$) and previous frame (frame $_{n-1}$). The *pyrA* and *pyrB* is the image pyramid for *imgA* and *imgB*. They can be NULL and the system will automatically allocate for them, but it would be risk and not performance wise (Bradski and Kaehler, 2008). The next parameter, *featuresA* is an array of the data type CvPoint2D32f, which is a structure use to store point in OpenCV. The *featuresA* is the point array that found in *imgA*, and will try to be found and track them in *imgB*. Integer data *count* is the number of features stored in *featuresA*.

The *winSize* is the size of local window that can be changed if user wishes to. The larger the *winSize* is, the larger the motion can be tracked. The next parameter, *level* is an integer value that indicates level of pyramid that wished to perform optical flow on. Value 0 will tell the function not to perform any optical flow operation using image pyramid. The *track_error* parameter is a parameter that use for error tracking. Error is explained as the motion that has unusual difference on distance of source and destination. The *track_error* will record the distance of source point and found destination point. User can set the criteria for distance, thus eliminate errors. The *criteria* parameter is use to tell the function criteria of iteration termination. And the last parameter, *flags* is use to control the function routine's internal bookkeeping.

3.5 Summary

The main techniques used in this project can be category into two, which are motion detection and motion tracking. In between, the contouring and object marking will be implemented instead of vehicle classification and identification. Motion detection will be performed by using background subtraction technique, as it works well when the scene captured is static. Contouring and object marking technique will provide information of foreground object extracted from background subtraction. And finally the motion tracking will be carried out using LK optical flow. Every technique has its own requirement and functionalities. The summary of their functionalities is shown in table 3.1.

Techniques	**Input**	**Output**
Background subtraction	Frame from camera/video	Binary image that had background subtracted
Contouring and object marking	Background subtracted binary image	Object information (point, size, centre point, area of pixels)
LK optical flow	Centre point of objects as features	Tracked object motion information (direction, motion path)

Table 3.1 Summary of functions required input and their output

Chapter 4: Experiments and Tests

4.1 Introduction

This chapter will demonstrate the experiments have been done during the research of the project.

4.2 Evaluation Data

This research project is targeted to detect and track the motion of vehicle that captured by camera, which is usually captured at outdoor environment. To assess their ability and practicability on vehicle tracking, 7 video files that are captured from a static camera that containing moving vehicle objects will be the test data for this research project.

Every video is captured in different location and different scene. Some of the scene is challenging, such as night scene with multiple vehicle lights turned on that might affect the accuracy, unintentionally captured rain drop that fall on the camera itself, instable scene due to camera shake, and more. The resolution of these video samples is fixed at 320x240, and their length is fixed at 20 seconds. The characteristics of every video will be explained in table 4.1. The screenshot of every video is also available in appendix A.

Video	Characteristics
1	Night scene with multiple vehicle lights turned on
2	Inside the tunnel, where the entrance is covered in high brightness
3	Day scene, Road with medium density of car
4	Day scene, but vehicle may be occlude other vehicle
5	Night scene, instability frame due to strong wind
6	Day scene, road with low density of car (little occlusion happen)
7	Day scene, camera is near to ground, size of object is big

Table 4.1 Explanation of test subjects characteristics (7 video files)

4.3 Experimental Phases

While this research project featured three major technique (background subtraction, contouring and object marking, and LK optical flow tracking), every of them is heavily dependent on each other output, except MoG background subtraction as it directly take the frame from video as input. In other word, it is necessary to assess them separately if any problem occurred or inaccuracy in result. Therefore, there will be three experimental phases and will individually carried out, to assist the problem analysis in the end of experiment.

The first experimental phase is focus on assessment of LK optical flow tracking. Since this research project is focusing on assessing optical flow, most of the experimental effort will be put on this phase. The second experimental phase will assess the ability of MoG background subtraction, contouring and object marking. This is to help on problem identification and further result analysis. The third phase will assess the whole operation by integrating background subtraction, contouring and object marking, and LK optical flow tracking.

4.3.1 Experiment Phase 1

To individually assess the performance of LK optical flow, the experiment will be run by manually input the features point into the algorithm. This is to ensure the result is not affected by the automatic features point detection mechanism, as wrong appointed features point will result in inaccuracy of tracking and result.

There will be 3 tests in this phase, involving the variation value of local window size. The first test will test with using window size 5x5, the second will use window size 10x10, and the last test will try with window size 15x15.

In every test, numbered of vehicles will be selected as tracking target. The tracking is considered fail if the tracking stopped half way, or the tracking tracked the wrong object. The tracking is considered success if the tracking is tracking the correct object from the start to the end.

Video	Vehicles	Tracked	Hit rate	Main reason of tracking failure
1	16	12	75.00%	Wrong point tracked when exit
2	17	10	58.82%	Wrong point tracked when exit
3	13	7	53.84%	Wrong point tracked when exit
4	20	14	70.00%	Wrong point tracked when exit
5	10	8	80%	Wrong point tracked
6	16	15	93.75%	Occlude at border
7	7	5	71.42%	Occlude at border
Average hit rate:			71.83%	

Table 4.2 Result of LK Optical Flow using window size 5x5

Video	Vehicles	Tracked	Hit rate	Main reason of tracking failure
1	16	15	93.75%	Loss of tracking (tiny object)
2	17	12	70.00%	Wrong point tracked when exit
3	13	12	92.30%	Wrong point tracked when exit
4	20	17	85.00%	Occlude at border (colour leak)
5	10	9	90.00%	Hard to find strong feature
6	16	15	93.75%	Occlude at border (tiny object)
7	7	6	85.71%	Occlude at border
Average hit rate:			87.21%	

Table 4.3 Result of LK Optical Flow using window size 10x10

Video	Vehicles	Tracked	Hit rate	Main reason of tracking failure
1	16	15	93.75%	Loss of tracking (tiny object)
2	17	13	76.47%	Wrong point tracked when exit
3	13	11	84.62%	Hard to maintain feature quality
4	20	17	85.00%	Occlude at border (colour leak)
5	10	9	90.00%	Hard to find strong feature
6	16	15	93.75%	Occlude at border (tiny object)
7	7	6	85.71%	Occlude at border
Average hit rate:			87.04%	

Table 4.4 Result of LK Optical Flow using window size 15x15

The difference success rate of test using window size 10x10 and test using window size 15x15 is not much. However, the window size 5x5 shows weaker results, and making similar error in all the video test subjects. The result also concludes that strength of feature, colour and size of object is important to LK optical flow.

(a) Vehicle on tracking (b) Vehicle tracking stopped

Figure 4.1 Tracking error due to size of object is too tiny

In figure 4.1(b), the red circle representing the actual location of vehicle being tracked in figure 4.1(a). The tracking process stopped at the end of green colour trail, as optical-flow failed to identify such small object which is also even hardly identified by human. Despite such error, the tracking on normal object is still achievable and provides satisfactory result.

4.3.2 Experiment Phase 2

In this phase, background subtraction and contouring and object marking will be put into test. Given the same video input, the background subtraction will first identify the background and subtract them out, leaving foreground (object) on the image, and finally pass it to contouring and object marking process to extract out the object information. The object information includes centre point of object, which is later become the feature for object in LK optical flow. Therefore, the quality of object detection will directly affect the quality of object tracking.

Two tests will be carried out in this phase. The difference between them is the second test will perform additional erosion morphology, while the first one simply gives the final result without any image processing or morphology.

Video	Correct Detection (%)	False detection (%)	Main reason of error
1	75.00%	67.00%	Vehicle light interruption and object size too small
2	63.76%	63.51%	Too many fragment (noise)
3	76.83%	28.69%	Object size too small
4	79.32%	46.00%	Occlusion of vehicle is identified as one object
5	64.56%	69.53%	Vehicle light interruption
6	82.00%	23.20%	Object size too small, occlusion of vehicle
7	93.35%	17.26%	False detection (noise)

Table 4.5 Result of background subtraction and contouring and object marking

Video	Correct Detection (%)	False detection (%)	Main reason of error
1	60.00%	53.27%	Vehicle light interruption and object size too small
2	54.37%	46.26%	Object density too high
3	67.33%	20.51%	Object size too small
4	70.00%	31.82%	Occlusion of vehicle is identified as one object
5	56.83%	32.96%	Vehicle light interruption
6	82.00%	13.51%	Object size too small, occlusion of vehicle
7	93.35%	7.00%	False detection (noise)

Table 4.6 Result of background subtraction and contouring and object marking with erosion morphology

Moving object that is correctly identified and detected was categorized into correct detection. If the detection is false, for an example, wrongly detect background or noise as the moving object, this was categorized into false detection.

The results show that erosion morphology can help to mitigate the false detection rate of background subtraction, but will also decrease the accuracy of object marking, except in some scene where the object size is good for detection, and density of vehicle on road is not too high. The background subtraction also does not perform much well, especially in the scene where vehicle density is high, and the object is too small to be detected.

4.3.2 Experiment Phase 3

The final phase of the experiment is merging motion detection (background subtraction and contouring & object marking) with motion tracking (LK optical flow) together into one partial complete work, and assesses its practicability.

After the assessment in experiment phase 1, the window size for LK optical flow in this test is decided to be 10x10, as it holds the highest success rate compare with window size 5x5 and 15x15. The erosion morphology will be used in this test, as tracking error features is resource consuming and confusing.

	Object detection (with erosion morphology)		**Object Tracking (LK optical flow with window size 10x10)**
Video	**Correct Detection (%)**	**False detection (%)**	**Result evaluation and summary**
1	60.00%	53.27%	Noisy due to false detection
2	54.37%	46.26%	Improvement is suggested (false detection)
3	67.33%	20.51%	Improvement is suggested (false detection)
4	70.00%	31.82%	Improvement is suggested (false detection)
5	56.83%	32.96%	Noisy due to false detection
6	82.00%	13.51%	Occlusion problem
7	93.35%	7.00%	Minor error, acceptable

Table 4.7 Result of integrating objects detection and object tracking

As the result, the performance of background subtraction has greatly affected tracking operation by LK optical flow. Object detection provided too much error feature point to LK optical flow, thus making LK optical flow calculate on error feature points as well.

Figure 4.2 Sample screenshot of experiment phase 3 (video 6)

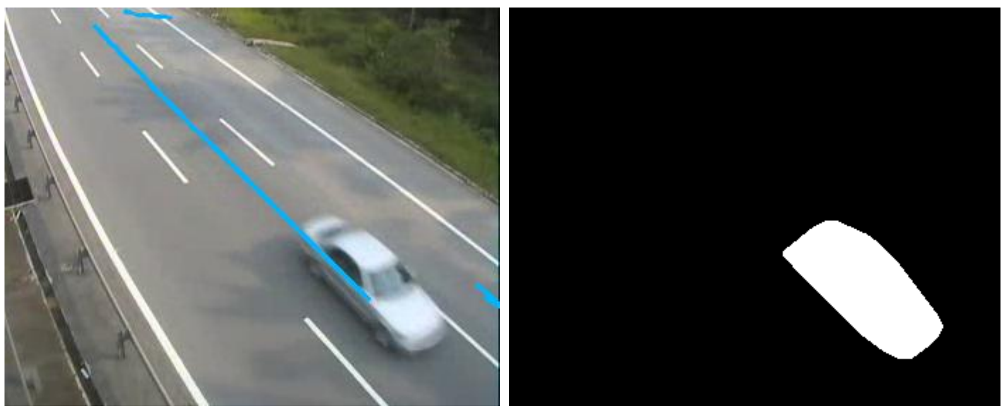


Figure 4.3 Sample screenshot of experiment phase 3 (video 7)

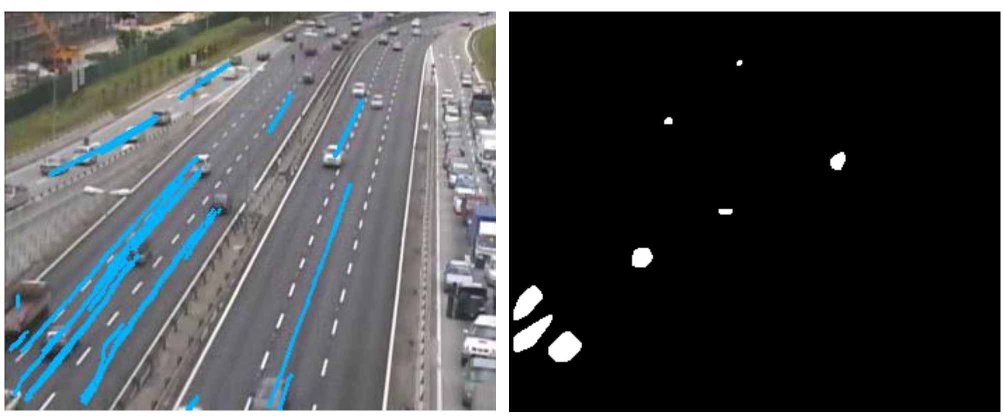


Figure 4.4 Sample screenshot of experiment phase 3 (video 3)

4.4 Summary

Three main experiments have been carried out in this research project to assist the identification of problem and assessment of motion detection and tracking techniques. 7 video that recorded traffic scene with different characteristics is the test data in all the three experiments.

The objective of the first experiment phase is to assess the ability of LK optical flow, the proposed technique that use to perform motion tracking in this research project. The result of experiment phase 1 is good, proving its ability and practicability.

The next experiment phase will try to test how well background subtraction, contouring and object marking can do in the videos given. The result is not so optimistic, as object identification is not proposed and reduces the accuracy of them.

The final phase will observe how good when motion detection (background subtraction, contouring and object detection) is integrate into motion tracking (LK optical flow) to automatically provide features for LK optical flow. The result explains that motion detection will require further improvement if motion tracking can provide more informative and better result.

Chapter 5: Conclusion and Future Works

5.1 Analysis and Findings of Research

LK optical flow is proposed long time ago, yet not many of the vehicles tracking system uses it to do the tracking job. One of the reasons of using other tracking technique is because optical flow is too complex and expensive to run compared with other techniques, and thus requires special optimized and designed system (Collins *et al.*, 2000). However, ability of hardware has been greatly improved since year 2000. To test the optical flow ability to tracking vehicle in current time, this research project is proposed and done.

From the results explained in chapter 4, experiment phase 1, which is to focus on assessment of LK optical flow on accuracy of tracking object, LK optical flow can be pretty accurate if the feature point given to it is strong and informative. Therefore the input to the LK optical flow is very sensitive, as LK optical flow itself do not have ability to identify bad feature or good feature. Bad feature may result in lower success chance of tracking the feature, as the bad feature do not strong enough to represent the object, and failed to become the identity of the object. If the good feature is provided, the tracking process can be very accurate, even is able to track the object that size is so small until it is hardly identified by human.

Experiment phase 2 shows that MoG background subtraction in OpenCV may require further improvement in future before it can put into practice use. MoG background subtraction works well in scene that background is static, often seen, and is not affected by other element such as dynamic lighting. The results in video 5 shows that MoG background subtraction is very sensitive to dynamic lighting, causing a lot of object fragment, and later become the bad feature (false feature). The object extraction is also heavily depending on the size of object and density of object on the road. MoG background subtraction does not work well when the background availability is low, which is clearly shown in experiment phase 2, tested using video 1 and 5.

The impact of MoG background subtraction is demonstrated in experiment phase 3. Although LK optical flow prove its strength in object tracking in experiment phase 1, it tracks also every features passed by MoG background subtraction, including the false features which do not representing any object. This made the LK optical flow provide false information as it is tracking “ghost object”. Despite such flaws, LK optical flow still proves its ability in video test 6 and 7, where it successfully tracked every vehicle.

Most of the works in this research project is carried out using OpenCV library. OpenCV provides a lot of good computer vision function, but also requires further improvement to provide more functionality as well. Some function such as contrast adjustment, image sharpen is not available, though they can be done by manually coding the algorithm. Documentation for OpenCV also requires further improvement, as currently some new function is not explained in any online documentation. Still, OpenCV is still proven an excellent tool for computer vision. With the open-source community, it is believed will continue growing and improving.

5.2 Future Works

In order to make the whole work more practicable, vehicle identification and classification can be implemented after the object has been identified. In this research project, neither background subtraction with contouring and object marking or LK optical flow has acknowledgment of object identity. This is one the reason that background subtraction provides false features, as the object detected is not further identified. If the object is correctly identified as vehicle, then it can eliminate false detection, thus providing better feature set for LK optical flow.

Currently some research (Culibrk *et al.*, 2007) shows the availability of using machine learning technique to build a better background model. Such technique can help to deal with scene that has light dynamic lighting, as it can learn the background more efficiently. Knowing that the quality of background subtraction is important to provide good features for LK optical flow, improving the background subtraction technique will definitely bring advantages to whole implementation.

The system will become better if road detection technique is applied. By providing the area which is identified as road on an image, LK optical flow can restrict the tracking within the region. This can be very informative for traffic condition analysis, by setting criteria that any motion that have move into non-road region can be possibly cause traffic accident. This can also assist the process the object detection, if the target object is restricted to vehicle only.

REFERENCES

Bradski, G. R., & Kaehler, A. (2008). *Learning OpenCV* . Farnham: O'Reilly.

Cheung, S. S., & Kamath, C. (2004). Robust techniques for background subtraction in urban traffic video. Visual Communications and Image Processing 2004, 5308(1), 881-892.

Collins, R. T., Lipton, A. J., Wixson, L., Kanade, T., Fujiyoshi, H., Duggins, D., et al. (2000). *A System for Video Surveillance and Monitoring*, Final Report, Robotics Institute, Carnegie Mellon University.

Culibrk, D. k., Marques, O., Socek, D., Kalva, H., & Furht, B. (2007). Neural Network Approach to Background Modeling for Video Object Segmentation. *Neural Networks, IEEE Transactions on*, 18(6), 1614 – 1627.

Gupte, S., Masoud, O., Martin, R. F., & P.Papanikolopoulos, N. (2002). Detection and Classification of Vehicles. IEEE Transactions on Intelligent Transportation Systems, 3(1), 37-47.

KaewTraKulPong, P & Bowden, R. (2001). An Improved Adaptive Background Mixture Model for Real-time Tracking with Shadow Detection, *European Workshop on Advanced Video Based Surveillance Systems*

Lee, D., & Kalmus, H. (1980). The optic flow field: the foundation of vision. Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences, 290(1038), 169-179.

Lucas, B. D., & Kanade, T. (1981). An Iterative Image Registration Technique with an Applicaiton to Stereo Vision. *International Joint Conference on Artificial Intelligence*, 3, 674-679.

Mcivor, A. M. (2000). Background subtraction techniques. In Proc. of Image and Vision Computing, 1, 147-153.

Shapiro, L. G., & Stockman, G. C. (2001). Motion from 2D Image Sequences. *Computer vision* (pp. 254-255). Upper Saddle River, NJ: Prentice Hall.

Stauffer, C., & Grimson, W. (1999). Adaptive background mixture models for real-time tracking. *1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2, 246-252.

Zhang, Y., Finger, S., & Behrens, S. (n.d.). Chapter 1. Introduction to Mechanisms. *School of Computer Science, Carnegie Mellon*. Retrieved March 11, 2011, from http://www.scs.cmu.edu/~rapidproto/mechanisms/chpt1.html

APPENDIX A

Example Screenshots of Video Samples

Video 1: Night scene with multiple vehicle lights turned on

Video 2: Inside the tunnel, where the entrance is covered in high brightness

Video 3: Day scene, Road with medium density of car

Appendix A: Example Screenshots of Video Samples

Video 4: Day scene, but vehicle may be occlude other vehicle

Video 5: Night scene, instability frame due to strong wind

Video 6: Day scene, road with low density of car (little occlusion happen)

Appendix B: Example screenshots of experiments results

Video 7: Day scene, camera is near to ground, size of object is big