

CSCI-B 657

Computer Vision

## **Speed Detection of Moving Vehicles in a Video**

Project Report

Spring 2016

Indiana University Bloomington

Bipra De

Ghanshyam Malu

Suhas Gulur Ramakrishna

| [bde@indiana.edu](mailto:bde@indiana.edu)

| [gmalu@indiana.edu](mailto:gmalu@indiana.edu)

| [suhgulur@indiana.edu](mailto:suhgulur@indiana.edu)

May 02, 2016



## Contents

Introduction .....	2
Background and related work.....	2
Experimental Setup.....	2
Methodology.....	3
Technology Stack .....	4
Challenges Faced.....	4
Results/Evaluation .....	5
Limitations.....	6
Future Work.....	7
Conclusion.....	7
References.....	7

## Introduction

In recent years, the number of traffic accidents have increased exponentially. Currently, speed detection is done in most of the areas using RADAR technology which uses Doppler Effect. Monitoring speeding vehicles manually is not feasible round the clock everywhere. Automation of speed detection is need of the hour. Given a traffic surveillance video, we aim to detect the speed of the moving vehicles.

## Background and related work

- Pornpanomchai<sup>1</sup> discusses about detecting the position of a moving vehicle in a frame and finding the reference points and calculating the speed of each static image frame from the detected positions.
- Abbott and Williams<sup>2</sup> discuss a method to extract the background image, find the foreground pixel mask to extract blob.
- Jun, Aggarwal, Gokmen<sup>3</sup> discuss Background Subtraction and blob segmentation using Gaussian background model (MoG).
- Martin, Philipp and Gerhard<sup>4</sup> propose a method for foreground segmentation which follows a nonparametric background modeling paradigm, thereby the background is modeled by a history of recently observed pixel values.

## Experimental Setup

- We define an entry point detection range ( $S$ ) and exit point detection range ( $E$ ) in the video in advance. When the detected vehicles coordinates lies within  $S$ , we note this entry frame number and when it lies in  $E$  we note the exit frame number.  $S$  is denoted by two green horizontal lines

near the bottom of the video and  $E$  is denoted by two cyan lines near the top of the video as shown in the image below.

- We select a traffic video taken by a still mounted camera with known real-world distance between our points of reference i.e.  $S$  and  $E$ .
- We currently are detecting vehicles in a single lane. Thus, we define a range in the video outside which we ignore all moving objects.

## Methodology

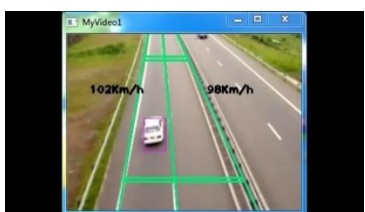

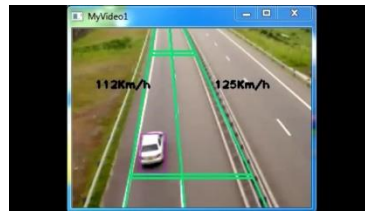
The following steps are followed to detect the speed of a moving vehicle in a video obtained from a still mounted camera on a highway.

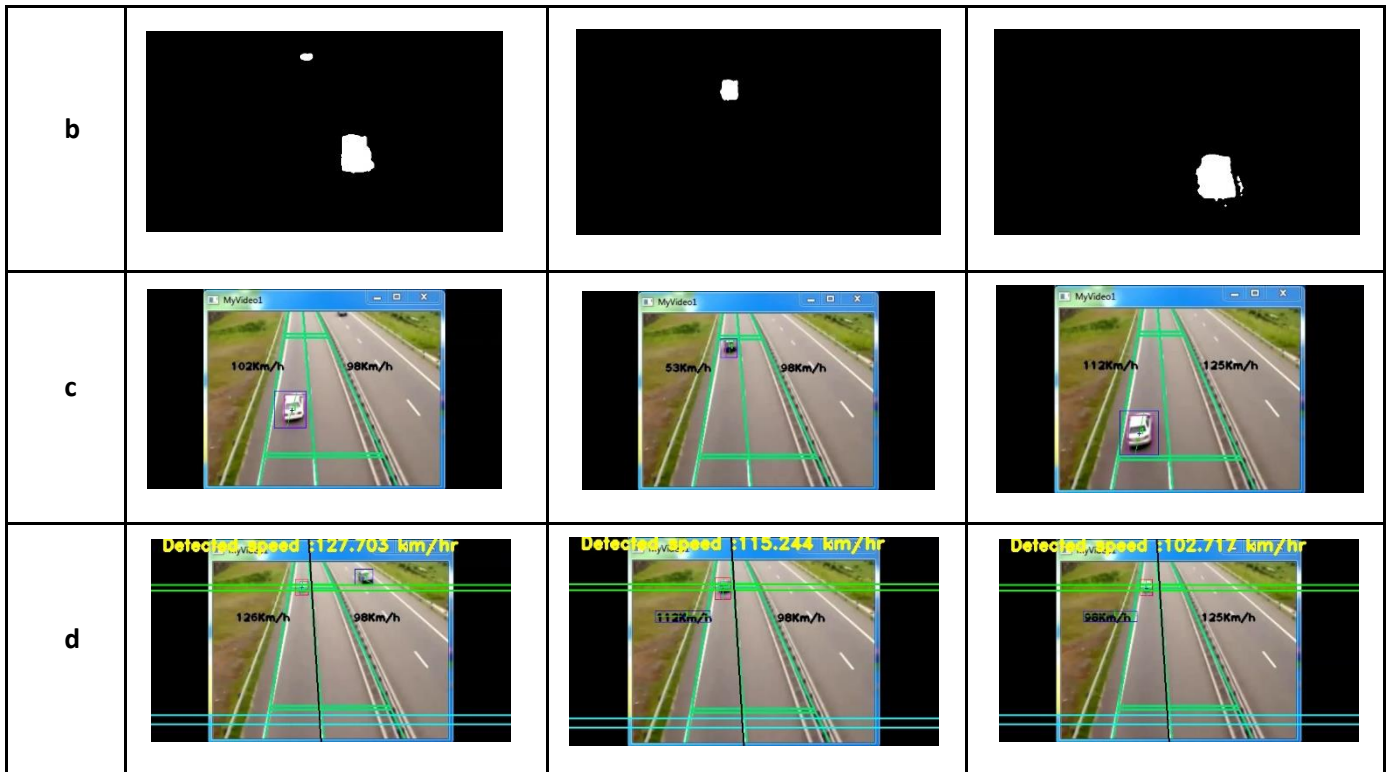
1. Extract frames from the given traffic surveillance video.
2. Find the foreground image by using Background Subtraction method.
3. Detect the moving objects by comparing consecutive foreground images and generate a blob object for each moving vehicle.
4. Filter out the blobs outside the selected lane or having size outside the selected area range.
5. Track the entry and exit frames for each moving vehicle/blob by storing the frame number when the centroid of the blob lies in the entry point detection range and exit point detection range respectively.
6. Calculate the number of frames lapsed between the entry and exit points of the moving vehicle.
7. Compute the speed of the vehicle using the equation below.

The mapping of the frame coordinates to real world coordinates of the video is given( $d$ ).

### Speed Detection:

- For each lane, we maintain a queue which contains the entry point of the vehicle.
- As the vehicle leaves the exit point, we look at its entry frame from the queue and determine the number of frames elapsed( $n$ ).
- Speed is computed using the equation  $speed = \frac{n * d}{FPS}$  .

	Video – 1	Video – 2	Video - 3
a			



- a - Frames from the original video.
- b - Foreground Image using Background Subtraction Method.
- c - Blob detected shown in blue.
- d - Start and end lines shown in green, cyan. Detected speed shown in yellow.

## Technology Stack

We use following technology for our implementation

- C++
- OpenCV for image functionalities.
- PBAS and cvBlob<sup>5</sup> open-source packages.
- Video of any format but should have FPS (frames per second) encode and contain 3-channels.

## Challenges Faced

Following are the challenges faced

### Magic Parameters

There were many magic parameters to tune. Few of those listed are

- Back Ground Subtraction

- Blob Detection
- Entry point range ( $S$ )
- Exit point range ( $E$ )
- Kernel width and sizes
- Size of Blobs to filter
- Lane limitation and
- Amount of frames to skip once the Blob is detected
- And many more.

Tuning these parameters for the existing dataset was time consuming and challenging.

Mapping Blobs across different frames:

- There was great challenge in mapping the blobs detected in one frame and the blobs detected in next frame.
- Basically the Blobs had no identifier to keep track of. Few things like finding the Tracks from cvBlob<sup>5</sup> package didn't help our cause.
- Finally we used Queue data structure to keep track of Blobs entering and exiting the ranges. However, this came with the cause of unable to detect blobs if vehicles changes lane or overtakes each other.

Getting a proper dataset

We needed dataset which has the distance between  $S$  and  $E$ . It also needs to contain the Ground truth speed of the vehicles. Finding such dataset was challenging and luckily we found one<sup>6</sup>. There were other datasets whose videos were corrupted like no frames per second data from the video or missing channels, etc.

## Results/Evaluation

For testing, we have considered a dataset containing videos<sup>6</sup> for which the distance between the start and end lines used for speed detection is given. It also has the speed of the moving vehicles which acts as the ground truth for our evaluation.

Video 1			
	Speed Computed	Actual Speed	Error %
Car 1	126	127.703	1.33%
Car 2	86	82.8947	3.75%

Video 2			
	Speed Computed	Actual Speed	Error %
Car 1	112	112.5	0.44%
Car 2	107	115.244	7.15%
Car 3	102	105	2.86%
Car 4	126	131.25	4.00%

Video 3			
	Speed Computed	Actual Speed	Error %
Car 1	98	102.717	4.59%

Entry point detection range ( $S$ ) and exit point detection range ( $E$ ) is a range of pixels. We detect the objects irrespective of where we have detected in that particular range. However, the actual distance is between the centers of  $S$  range and  $E$  range. If the blob misses this sweet spot (most likely it will) then we can expect some margin of error as the actual distance is not accurate. This is evident from the results we obtained.

## Limitations

- The algorithm would not work as expected in the following situations :
  - If the centroid of the detected blob doesn't pass through the region of the decided detection area (start min - start max and end min - end max).
  - If a vehicle changes lane anytime in the video (as we maintain a queue per lane for object tracking across frames)
  - If the vehicles overtakes another vehicles. The whole logic of mapping blobs across frames breaks.
- Computation is slow. Both open source packages we use – PBAS and cvBlob<sup>5</sup> that involve huge mathematical computation thereby slowing down the complete process.
- The road elevation in view of the camera has to be constant and shouldn't have difference in elevations.
- We have filtered the blobs by area size. This size is convenient for cars and bigger trucks. But this fails sometimes when a huge trailer is on the road. Our algorithm might detect more than one blobs for one trailer which is wrong. However if we increase the limit on the size of blobs will cause errors like merging two adjacent blobs (vehicles) into one single entity. After lot of experimentation we have kept ones which work best for our project.

## Future Work

Kalman filter can be used for object detection across frames and this approach's accuracy can then be compared against our current approach.

We can look for faster background subtraction methods apart from PBAS as it is very computation intensive. This saves some time and can be used for live video analysis. The same can be applied for cvBlob<sup>5</sup> as well however there might be some compromise on the accuracies.

Moreover, the current project can be enhanced further in also detect the number plates which in turn has many real world applications including assisting investigative authorities. Basically, this can be huge start to automating traffic speeding to greater extent.

## Conclusion

We found in our experimentation that the Background Subtraction works well in detecting moving vehicles. We noticed that we need to properly tune the range in which a blob area can lie as there can be lot of unwanted movements in the video like shadows, leaves, trash, etc. whose speed we don't want to compute. The accuracy of speed detection depends heavily on how the moving objects are tracked across frames. So we have to assume that no vehicle changes lane as object tracking algorithms didn't do very well in such scenarios. Overall, each phase involved in our algorithm worked fairly and there is a scope for improvement.

We feel that this project shows that there is huge scope for automating speed detection and can be used in practical cases with most accurate results. Background subtraction and blob detection method suits our needs exactly. This start can be big initiative for better approaches to limit the errors and improve the quality. Huge inroads needs to be done to make this an instantaneous speed detection as the current one has lot of benchmarks to be met.

## References.

- <sup>1</sup> Vehicle speed detection system - *C. Pornpanomchai*
- <sup>2</sup> The Speed Detection Algorithm Based on Video Sequences - *W. Liang*
- <sup>3</sup> Detecting Global Motion Patterns in Complex Videos - *Min Hu, Saad Ali, Mubarak Shah*
- <sup>4</sup> Background Segmentation with Feedback: The Pixel-Based Adaptive Segmenter - *Martin, Philipp, Gerhard*
- <sup>5</sup> cvBlob - *Fabrice de Gans-Riberi* (<https://github.com/Steelskin/cvblob>)
- <sup>6</sup> Dataset - <https://www.youtube.com/watch?v=tqeDene7r74>
- <sup>7</sup> Vehicle Detection, Tracking and Counting - *Andrews Sobral*