

Santa Clara University
Department of Computer Engineering
COEN 281 – Data Mining
Professor Ming-Hwa Wang, Ph.D
Winter 2016

Defining a Better Vehicle Trajectory With GMM

Christiane Gregory
Abe Millan

Contents

[Abstract](#)

[1. Introduction](#)

[2. Theoretical bases and literature review](#)

[3. Hypothesis](#)

[4. Methodology](#)

[5. Implementation](#)

[6. Data Analysis and Discussion](#)

[7. Conclusion](#)

[8. Future Work](#)

[9. Bibliography](#)

List of Figures

Figure 1: Valid paths found on 3 scenes.....	5
Figure 2: Highlight of paths and trajectories on scene 1.....	5
Figure 3: Errors on information due to poor cleaning of data.....	6
Figure 4: Output for 5 different videos of the software.....	9
Figure 5: Least Square Fitting Trajectories.....	9

Abstract

There are several methods to automatically analyse surveillance videos. The challenge is to find faster ways to detect, classify, and find trajectories so it can be used online. Our proposal is to many techniques to improve processing time and return great results.

Key-words: Video Surveillance, data mining, motion pattern, moving objects trajectories.

1. Introduction

Objective

To benchmark other techniques besides GMM to create an algorithm that uses the best features of all to make a more efficient vehicle trajectory detection algorithm.

What is the problem?

There is an abundance of surveillance cameras that are not being used efficiently because they must be manually supervised in order to detect anything interesting. Being able to automate surveillance systems will assist human operators as well as lower costs of labor and increase the reliability of the surveillance systems.

Why is this project related to this class?

To quote the textbook "The most commonly accepted definition of Data Mining is the discovery of "models" for data [1]. While it may be tempting to categorize trajectory analysis as a form of video processing, what we are really doing in this project is modeling the common trajectories of roads via clustering algorithms using video frames as data..

Why other approach is no good?

The motivation for GMM, a spatial distribution-method, comes from the observation that spatial distance-based methods lack a probabilistic explanation for abnormality detection, require the cluster number in advance, and have a high computational cost.

GMM is presented to be quite powerful. However, GMM can be improved by combined with the best features of similar approaches that has better performance in precise tasks. Our proposal is to find such algorithm, such tasks and improve GMM by substituting GMM on those tasks.

Area or scope of investigation

Our project will focus on comparing two existing algorithms from two different paradigm methodologies. We will focus on picking out the best features from both methodologies in order to create a more efficient, yet still accurate algorithm for trajectory detection.

2. Theoretical bases and literature review

Definition of the Problem

To be able to model object trajectories from real time video surveillance data.

Theoretical Background

The problem of trajectory analysis as mentioned above can be categorized into two classes: spatial distance-based methods and spatial distribution-based methods. Both these methods first use techniques such as background subtraction to extract and detect foreground images of objects. These objects are then used as input to both of these methods and output a model for trajectories for the scene.

Spatial distance-based methods:

These methods track and record the characteristics of trajectories for each object of objects by using consecutive frames. Once the trajectories for objects are collected, they are clustered by their similarity. These clusters are then used to represent the trajectories of the scene.

Spatial distribution-based methods:

These methods use the distributions of observations on the trajectories for trajectory analysis, but do not take into account the integrity of each cluster (such as LCSS) [4].

GMM:

GMM algorithm assumes that traffic trajectories are simple enough to be quadratic ($y = ax^2 + bx + c$) and so it describes each object's trajectory by a tracker T at time t by the tuple (a, b, c, v) where v is the direction of the motion. It then splits up the scene into $R \times C$ blocks. In each block the is modeled by a mixture of Gaussian distributions for trajectory tuples. This gives us a probabilistic way of describing the trajectories in the scene. This method then uses a graph-cut algorithm to group similar motion patterns and ultimately get the paths for the scene.

As seen in [3], many methods can be used to analyse trajectory clustering: K-means, Euclidean Distance, Symmetric Euclidean Distance (SED), HU distance, Principal Components Analysis (PCA), Dynamic Time Warping (DTW) and Longest Common Subsequence (LCSS). Even though the best combination shows to be PCA+Euclidean Distance, the author has a better result with SED.

According to Zhang [2] there are many methods to learn motion pattern, based on trajectory analysis. They can be divided in two groups: spacial distance-based methods (including Euclidean Distance, Hausdorff distance and Dynamic Time Warping-DTW) and spatial distribution-based method (such as GMM). Spacial distance algorithms has many drawbacks: they lack a probabilistic explanation for abnormality detection, riquer the cluster number in advance, have a high computational cost, and may not well approximate the true similarity. GMM is presented as the best solution to solve these problems.

GMM is presented to be quite powerful. However, GMM can be improved by combined with the best features of similar approaches that has better performance in precise tasks. Our

proposal is to find such algorithm, such tasks and improve GMM by substituting GMM on those tasks.

3. Hypothesis

We want to find the best (a more efficient) algorithm to define valid trajectory paths to cars. We believe if we find the fastest method to show the trajectory paths, finding abnormalities will be faster.

Our main goal is to find a most efficient algorithm to find valid paths than GMM alone.

There are many algorithms to solve this problem, we selected the best in our scope so we can find the best and worst features of each.

Given a surveillance traffic video, we want the software to be able to find all valid paths based on motion pattern of the cars. Based on papers related, GMM shows to have superior result. We want to improve the approach, by combining other techniques (Contours, Hungarian algorithm) to make it even better. We will measure computational time to find one path, all paths, how many paths are there and level of accuracy.

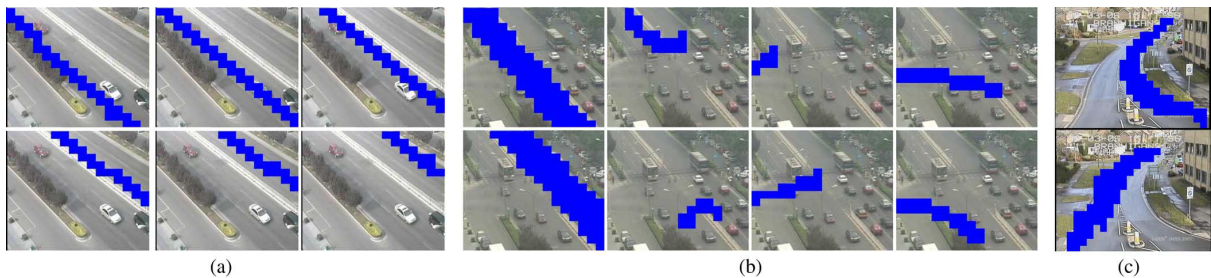


Figure 1: Valid paths found on 3 scenes. (a) 6 Valid paths found in scene 1. (b) 8 Valid paths found in scene 2. (c) 2 Valid paths found in scene 3.

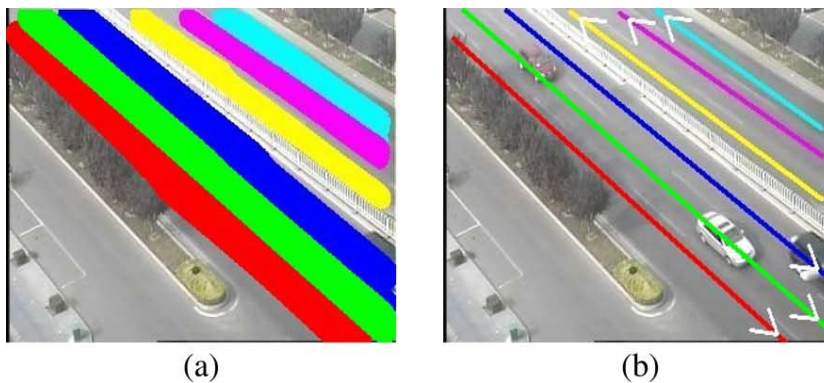


Figure 2: Highlight of paths and trajectories on scene 1.

(a) 6 Valid paths found and highlighted side by side in scene 1. (b) 6 Valid trajectories defined on those paths of scene 1.

4. Methodology

We solved the problem by improving the GMM technique and finding a more efficient way to define the valid paths.

The data used was collected from the internet, from many traffic surveillance websites and from websites containing other projects on traffic as well such as Nagel [5].

One important step was to filter out useless data. There can be too much unnecessary information that can be eliminated in order for a better result. We eliminate information collected that were not within some threshold. The following images show the information extraction without the boundaries causing the data to be “polluted”.

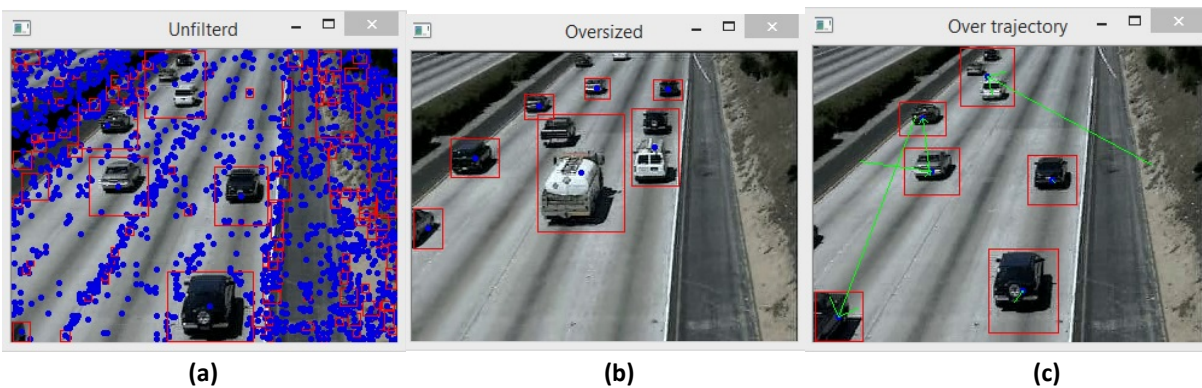


Figure 3: Errors on information due to poor cleaning of data. (a) All motion blobs detected. (b) Oversized blobs allowed. (c) Unlimited distance between bounding boxes.

The development was made using C++, OpenCV and Visual Studio.

Our output represents the vehicles detected, the paths detected, and the valid paths found. The final algorithm is a combination of all techniques combined resulting in a more efficient method.

To test against hypothesis should answer the question “did we find a better solution than GMM alone for defining valid trajectory paths?”

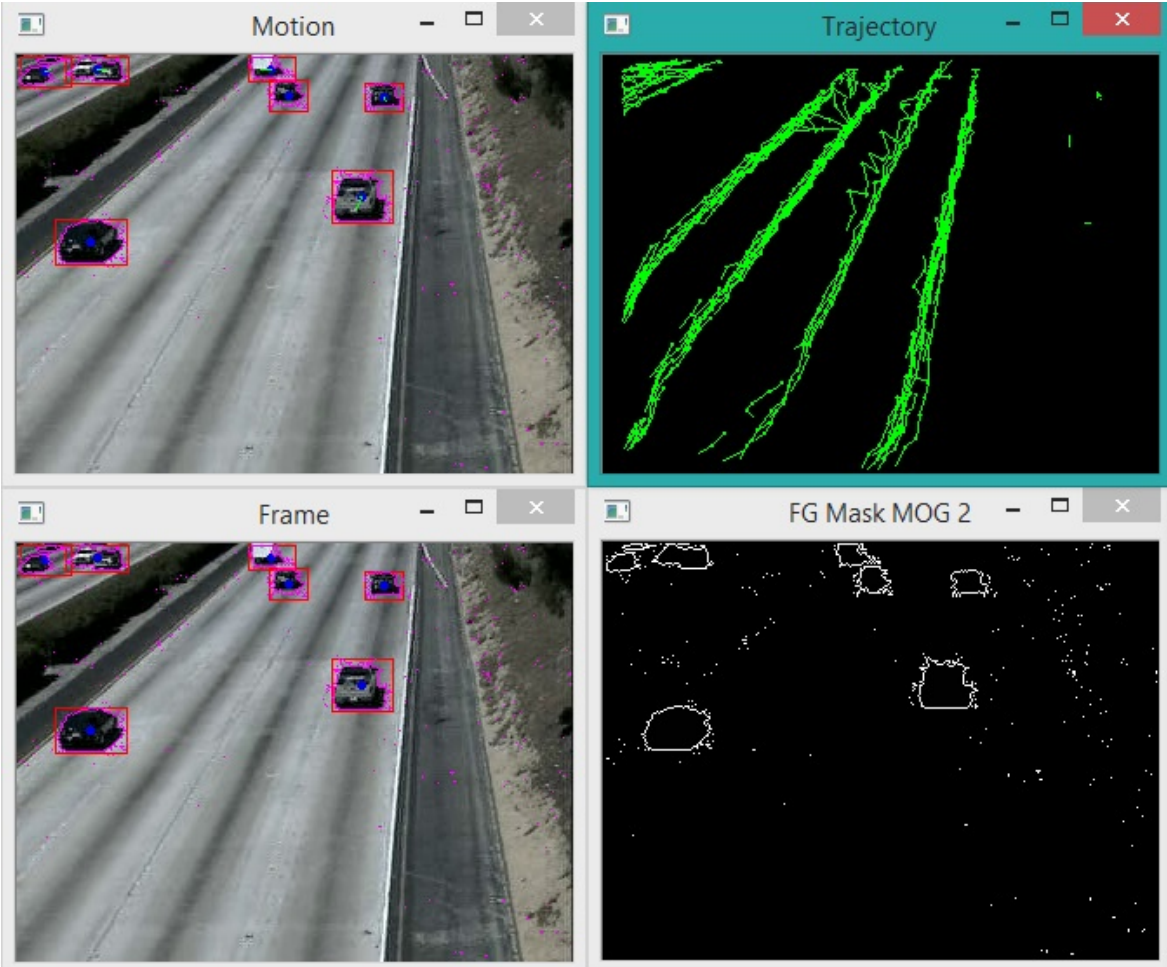
5. Implementation

The implementation was made using reading a video file and extracting information frame by frame, with any assumptions to obtain mostly good data. In each frame we detect the vehicles and its center. From frame to frame the the distance between the center of the vehicles are collected until each goes out of frame. At that moment we calculate the Least Square to find a quadratic regression of the trajectory (of the format $ax^2 + bx + c = 0$). This is modeled by a quadratic because it is assumed that traffic behavior is simple enough.

All the cars we track are stored into an array of trajectories, which we call the Cars Class. Using the Cars class we insert new car objects based on a hungarian matrix algorithm. Once a unique car is tracked we continue to update it frame by frame by adding it's centroids to its list. When the car disappears from the scene or it can no longer be tracked it is marked as finished through the data structure. As we are collecting cars we have GMM algorithm waits to train the model until it sees that a sufficient amount of cars in the Cars list. For use this number is around 200. Once that limit is reached, we traverse through the Cars list and simply its list of centroids by the parameters (a, b, c) we obtain using Least Squares. The whole scene is then partitioned into R X C blocks, where R X C for us was 8 X 8, and each trajectory parameter is placed into to a block if it has ever visited that corresponding block. This is done via our Video Blocks class. Once the trajectories have been sorted to their corresponding block, a GMM classifier is implemented on each of the blocks. We implemented GMM via the machine learning library that OpenCV provides.

6. Data Analysis and Discussion

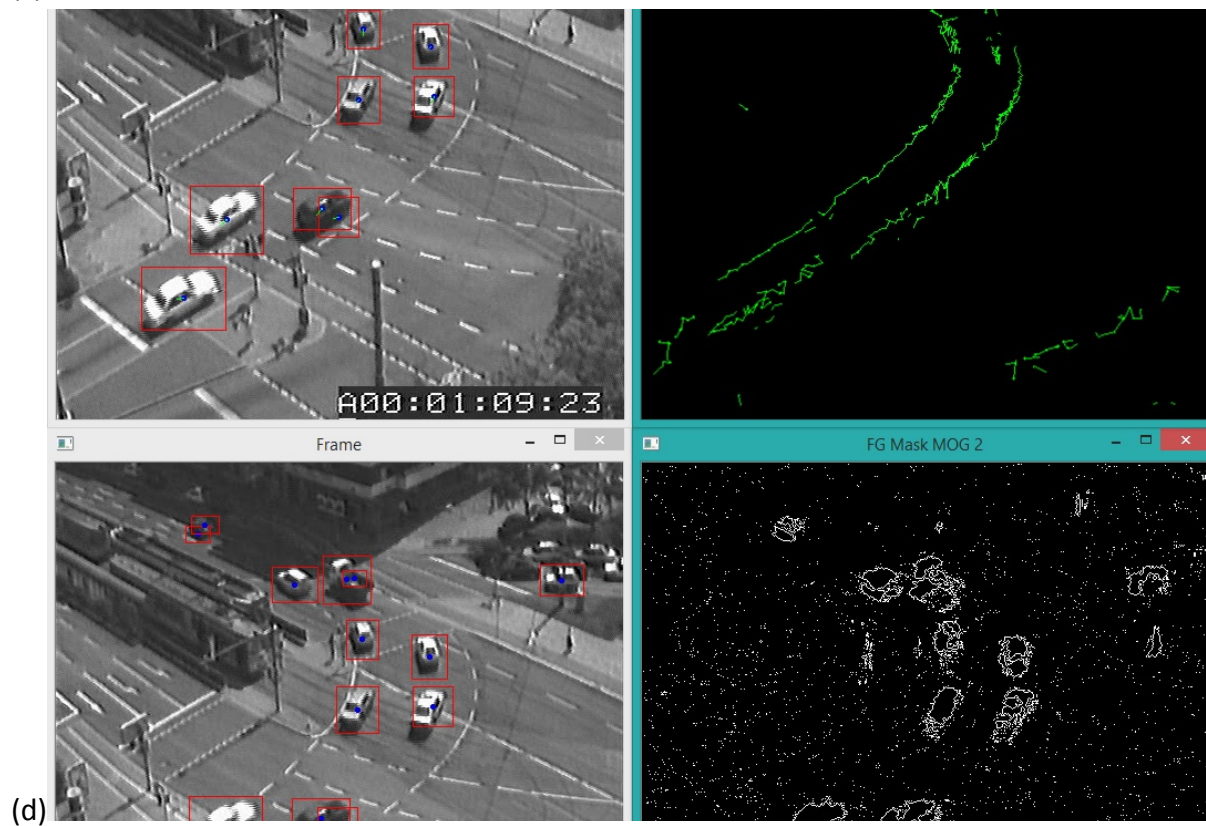
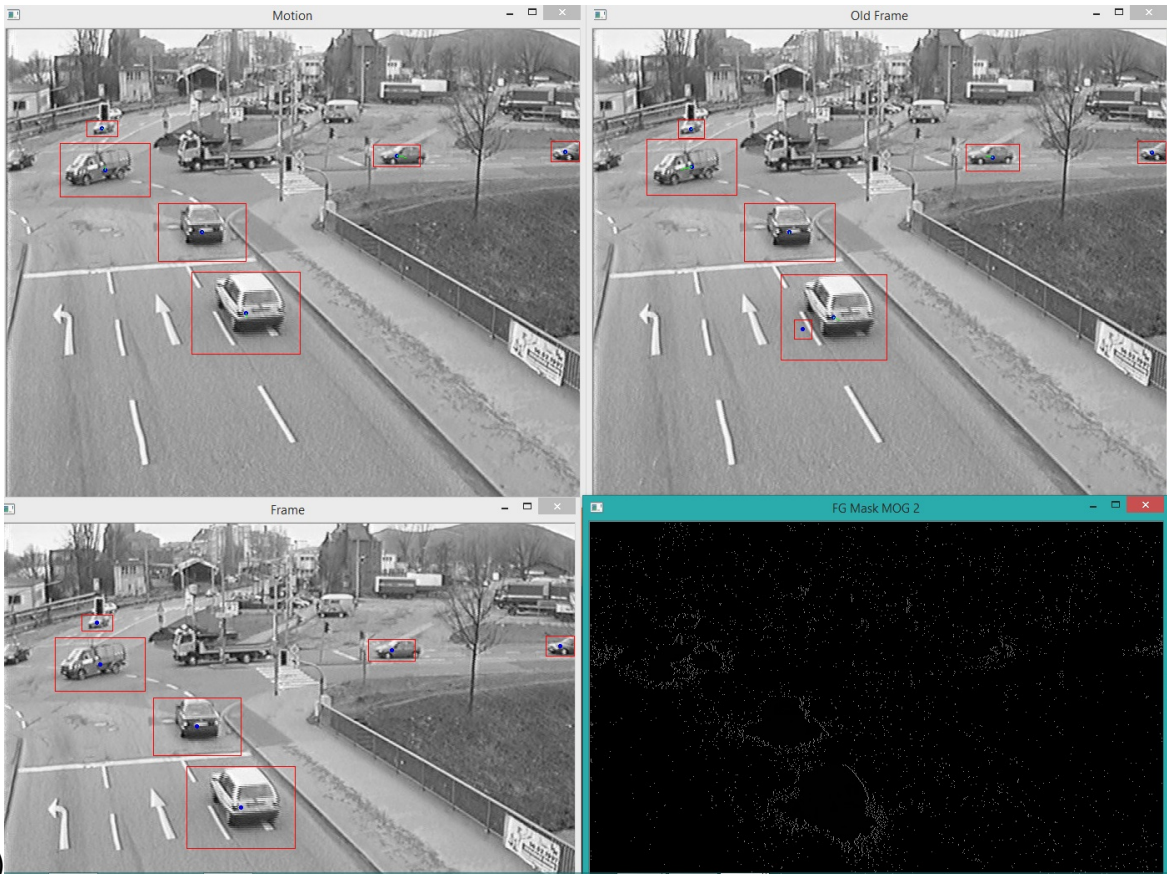
More than 10 different videos were tested from different font.s The algorithm has a great performance on most of the videos tested except for large videos. The output data was satisfactory within the scope of the project. The following images show results generated with the software developed.



(a)



(b)



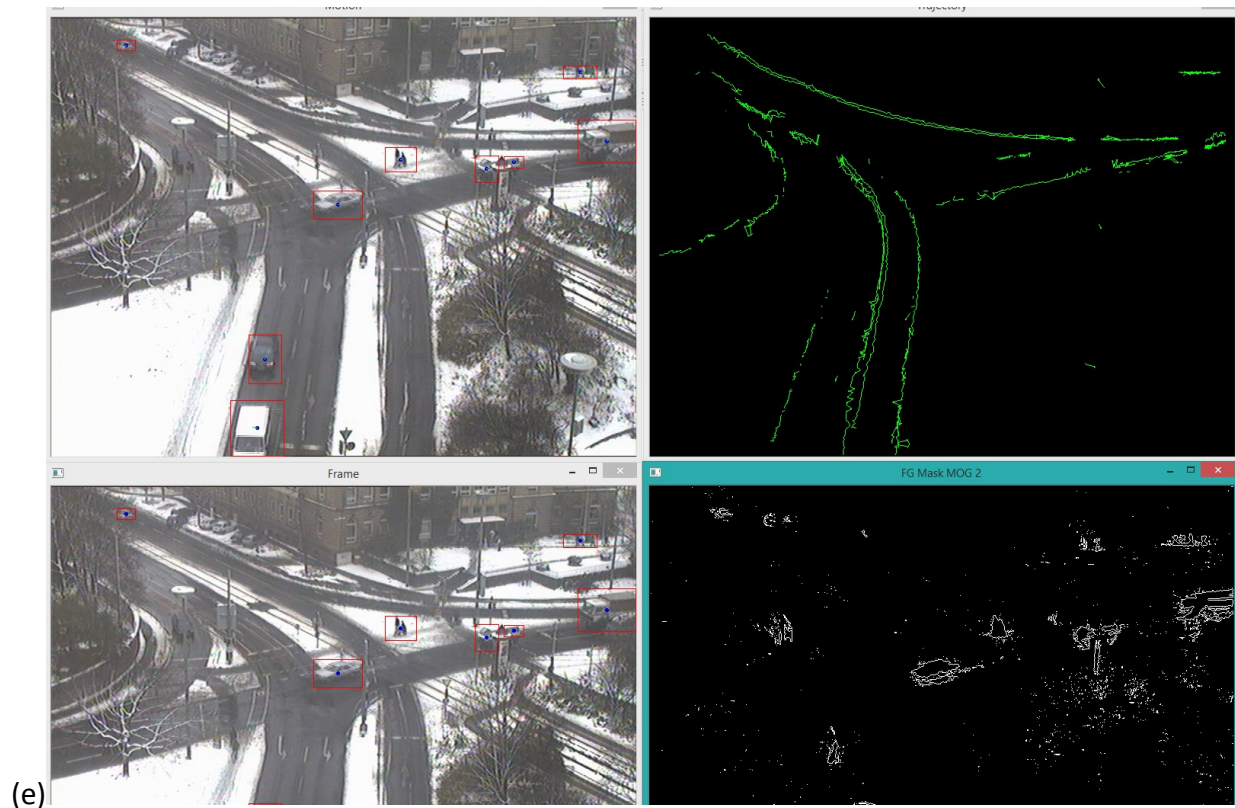
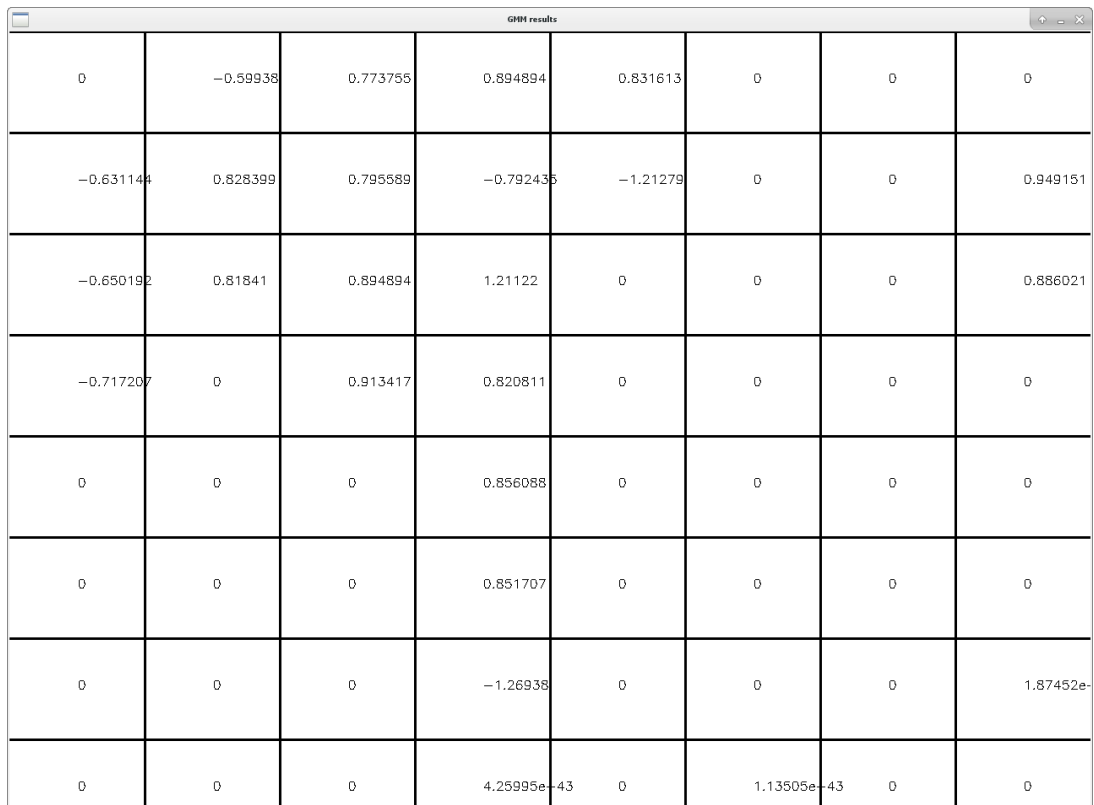


Figure 4: Output for 5 different videos of the software. Top left: The Motion detection and trajectory; Top right: Previous frame; Bottom left: Current frame; Bottom right: The MOG 2 background subtraction.

The algorithm finds great trajectories and records the valid points for most of the valid data. Same data is left off due to the filtering or generalization of the bound box. Even though, the data is still refined when calculating the quadratic regression as it needs at least 3 points to be calculated.

The results obtained from GMM are nClusters--we implemented GMM with 3 clusters based on the research of [2]-- of expected values for the parameters (a,b,c). In addition each expected value also comes with a weight that determines its significance in the model. Taking the parameter with the largest weight we then display the b parameter on our 8X8 grid. This parameter signifies the motion pattern a particular block. As shown in Figure 6, this is the GMM result we get for the scene in 4. You can that it does a fairly good job at representing the scene. Since the b parameter

Figure 5. Results from the GMM algorithm for the scene depicted in 4a



The image shows a screenshot of a window titled "GMM results" containing an 8x8 matrix of numerical values. The values are as follows:

0	-0.59938	0.773755	0.894894	0.831613	0	0	0
-0.631144	0.828399	0.795589	-0.792435	-1.21279	0	0	0.949151
-0.650192	0.81841	0.894894	1.21122	0	0	0	0.886021
-0.717207	0	0.913417	0.820811	0	0	0	0
0	0	0	0.856088	0	0	0	0
0	0	0	0.851707	0	0	0	0
0	0	0	-1.26938	0	0	0	1.87452e-
0	0	0	4.25995e-43	0	1.13505e-43	0	0

```
Terminal - root@abemillan:~/Developer/openCVPractice/DataMiningTrafficProject/src
File Edit View Terminal Tabs Help
Car #3 has 28 trajectories
Car #3 has a:0.000986 b: -1.518283 c: 238.128098
Car #24 has 22 trajectories
Car #24 has a:-0.020675 b: 4.683914 c: -27.801119
Car #40 has 37 trajectories
Car #40 has a:0.023808 b: -14.829285 c: 2103.330322
Car #43 has 25 trajectories
Car #43 has a:-0.000280 b: -0.844850 c: 132.970901
Car #45 has 55 trajectories
Car #45 has a:0.011118 b: -5.263264 c: 612.656860
Car #49 has 27 trajectories
Car #49 has a:-0.000404 b: -0.883342 c: 141.342804
Car #52 has 31 trajectories
Car #52 has a:-0.000360 b: -1.197390 c: 218.138000
Car #58 has 34 trajectories
Car #58 has a:0.002298 b: -1.361568 c: 161.437393
Car #71 has 34 trajectories
Car #71 has a:0.004634 b: -1.578869 c: 165.679245
Car #75 has 28 trajectories
Car #75 has a:0.027268 b: -15.313869 c: 2052.115723
Car #76 has 21 trajectories
Car #76 has a:0.002250 b: -1.796227 c: 255.147369
Car #87 has 38 trajectories
Car #87 has a:0.003915 b: -6.050499 c: 1135.957764
Car #98 has 44 trajectories
Car #98 has a:0.003154 b: -3.069101 c: 480.494568
Car #99 has 28 trajectories
Car #99 has a:0.000130 b: -0.972635 c: 142.439453
Car #107 has 27 trajectories
Car #107 has a:0.000678 b: -1.449972 c: 232.113632
Car #110 has 28 trajectories
Car #110 has a:0.006514 b: -6.808009 c: 1172.779663
Car #129 has 21 trajectories
Car #129 has a:-0.000637 b: -1.163161 c: 220.226135
Car #3 has 28 trajectories
Car #3 has a:0.000986 b: -1.518283 c: 238.128098
Car #24 has 22 trajectories
Car #24 has a:-0.020675 b: 4.683914 c: -27.801119
Car #40 has 37 trajectories
Car #40 has a:0.023808 b: -14.829285 c: 2103.330322
Car #43 has 25 trajectories
Car #43 has a:-0.000280 b: -0.844850 c: 132.970901
Car #45 has 55 trajectories
Car #45 has a:0.011118 b: -5.263264 c: 612.656860
Car #49 has 27 trajectories
Car #49 has a:-0.000404 b: -0.883342 c: 141.342804
Car #52 has 31 trajectories
Car #52 has a:-0.000360 b: -1.197390 c: 218.138000
Car #58 has 34 trajectories
Car #58 has a:0.002298 b: -1.361568 c: 161.437393
Car #71 has 34 trajectories
Car #71 has a:0.004634 b: -1.578869 c: 165.679245
Car #75 has 28 trajectories
Car #75 has a:0.027268 b: -15.313869 c: 2052.115723
Car #76 has 21 trajectories
Car #76 has a:0.002250 b: -1.796227 c: 255.147369
Car #87 has 38 trajectories
Car #87 has a:0.003915 b: -6.050499 c: 1135.957764
Car #98 has 44 trajectories
Car #98 has a:0.003154 b: -3.069101 c: 480.494568
Car #99 has 28 trajectories
Car #99 has a:0.000130 b: -0.972635 c: 142.439453
Car #107 has 27 trajectories
Car #107 has a:0.000678 b: -1.449972 c: 232.113632
Car #110 has 28 trajectories
Car #110 has a:0.006514 b: -6.808009 c: 1172.779663
Car #129 has 21 trajectories
Car #129 has a:-0.000637 b: -1.163161 c: 220.226135
^C
[root@abemillan src]#
```

Figure 6. Here is a list of Cars that our algorithm has finished tracking and has designated a trajectory parameter (a, b, c)

Is the linear parameter it makes sense to see positive number here since the cars in this scene are traveling in the positive slope direction. What is interesting is the lack of parameters towards the middle. This could be due to objects that don't have sufficient trajectories since we cut off the GMM algorithm to only use cars with more than 20 trajectories.

7. Conclusion

The detection and classification of vehicles were rapidly obtained and stored. After diverse testing was proved that the algorithm works in a satisfactory manner for either far, mid or close range videos. The algorithm also includes a great "cleaning" mechanism, eliminating bad data from the moment it starts, that leads to better performance and lower memory required.

Compared with the other studies the presented algorithm considers a larger variety of sources as well as providing a better performance for the same result, making this a better solution.

In order for GMM to work optimally it needs to be fed objects that hold more centroids in their list. If they don't have enough then they could become misrepresented. Besides this GMM works fast and in our examples has small training data to work with which could be the reason why we see some peculiar results.

8. Future Work

Some improvements can still be further studied that would benefit our work. One improvement would be on the detection of the vehicle using histogram to make sure that the closest box on the next frame belongs to the same identity. Another way to improve it would be to allow the trajectory to "skip" points in frames. Even if the vehicle was not correctly detected on the next frame, if its not close enough to the edge of the image it could still be on the next frames.

Additionally we would love to mess with the parameters of GMM

9. Bibliography

[1] J. Leskovec, A. Rajaraman, J. Ullman, "**Mining of Massive Datasets**", Cambridge University Press, December 30, 2011.

[2] T. Zhang, S. Liu, C. Xu and H. Lu, “**Mining semantic context information for intelligent video surveillance of traffic scenes**” IEEE Trans. Ind. Informat., vol. 9, no. 1, pp. 149-160, Feb., 2013.

[3] Z.Chen, Y. Yan, T. Ellis, “**Lane detection by trajectory clustering in urban environments**”, Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference, Qingdao, pages 3076 - 3081 , Oct 2014.

[4] D. Buzan, S. Sclaroff, and G. Kollios, “**Extraction and clustering of motion trajectories in video,**” in Proc. of International Conference on Pattern Recognition, 2004.

[5] Nagel, H. Institut für Algorithmen und Kognitive Systeme. **Image Sequence Server.**
Dataset: http://i21www.ira.uka.de/image_sequences/