Vehicles detection and tracking in videos for very crowded scenes

Sara Atito Aly
Nile University, Smart Village, Egypt
sara.atito@nileu.edu.eg

Ahmed Mamdouh
Nile University, Smart Village, Egypt
ahmed.mamdouh@nileu.edu.eg

Moataz M. Abdelwahab
School of Communications and Information Technology
Nile University, Smart Village, Egypt
mabdelwahab@nileuniversity.edu.eg

Abstract

Counting and tracking vehicles in very crowded scenes is a very challenging problem, where many products require discipline conditions to deal with. In this paper, a novel algorithm for automatically counting the number of moving vehicles and estimating their velocities and paths in regular and very crowded scenes, under different conditions, is presented. In this method, interest points are detected and trajectories are calculated independently where confusing trajectories are removed. Initial clustering of the interest points based on proposed mathematical relations is performed. The number of moving vehicles is estimated by grouping the initial clusters based on a new adaptive background construction method, maximum sub-rectangle sum algorithm and disjoint set data structure. Our algorithm has been applied to a collected dataset representing very crowded traffic scenes, where it showed an excellent high accuracy. In addition, it has low storage and computational requirements, which promotes it for real time applications.

1 Introduction

Intelligent transportation system (ITS) aims to ease traveling, enhance roads usage efficiency and decrease congestion and time consumption, which is extremely useful for public security and resources management. Traffic flow monitoring system can be divided into two categories: On-road detectors and vision based systems. On-road detectors, such as loop detectors, have high costs and some difficulties in their installation and maintenance. In addition, they are point detectors and provide limited amount of information. Vision based systems are more popular due to their low cost and easy maintenance. Most approaches for vehicles detection are using motion features to detect moving vehicles from video sequences. However, due to illumination and background variation, developing an accurate and robust vehicle detection system for very crowded scenes under different conditions is very challenging. Different approaches have been presented, as background subtraction which is extremely sensitive to light changes. Adaptive background techniques are used in [1], [2]. Vibha [1] used background subtraction to improve the adaptive background mixture model which makes the system learn faster and more accurate. However, the accuracy is limited in crowded scenes due to the presence of occlusions. Other methods [3], [4] are based on finding symmetry in image for vehicle detection. Lei Gao [3] used the representation of red colors to find rear-lights regions in vehicles and symmetry function to analysis the symmetry of the color distribution to figure out the accurate position of the symmetry axis. However, this algorithm constrains vehicles to move in a certain discipline pattern. Alberto [4] searched for areas with a high vertical symmetry in multi-resolution images. Symmetry is computed using different size boxes centered around interest areas, which is computationally expensive. These methods do not fit for crowded scenes due to occlusion. In this paper, a novel method for counting moving vehicles in regular and very crowded scenes is proposed. This method is based on obtaining corner points and tracking them independently to obtain their trajectories. Confusing trajectories are removed and the remaining ones are initially clustered. Newly adaptive background construction, sub-maximum sum algorithm and disjoint set data structure are used to obtain bounding boxes around moving vehicles and their numbers and trajectories. Experimental results, performed on a newly collected dataset for very crowded scenes, confirm the excellent properties of our algorithm.
This paper is organized as follows: Section 2 describes our approach. Experimental results and analysis are introduced in Section 3. Conclusion is presented in Section 4.

2 Our Approach

This section presents our method for detecting and tracking moving objects in crowded scenes as described in table 1. Section 2.1 introduces features extraction and tracking. Remove confusing trajectories are presented in section 2.2. Section 2.3 describes the initial clustering technique. Statistical method for adaptive background construction is introduced in section 2.4. Finally, section 2.5 shows extraction of vehicles bounding boxes.

2.1 Extracting suitable corner points and their trajectories

Determining the motion parameters of moving points in a sequence of images can be computed by identifying pairs of points that correspond to each others in two images taken at time t and t + δt. A good property of corner points is that they can be precisely identified in an image rather than pixels in flat zones. A corner detector [5] can be described as a point with high spatial gradient and high curvature. For this reason, motion vectors associated with corner points are quite reliable. Rather than computing corner points

Table 1. Major Steps of the Proposed Algorithm

1− Extract suitable corner points and their trajectories.
2− Remove False Trajectories.
3− Initial clustering based on Angle, Displacement and Parallel Similarity.
4− Adaptive background construction.
5− Extracting bounding boxes of vehicles based on adaptive background construction method, submaximum sum algorithm and disjoint set data structure.

overall frame pixels, which waste time and efficiency, foreground pixels can be extracted from frame differencing, which highlights regions that changes in consecutive frames. Morphological operations are applied to remove noisy regions. The filtered foreground pixels are grouped into regions employing connected component labelling algorithm. corner points detector is applied on each region which assure that each corner is a moving pixel. This pre-processing step will also facilitate clustering. After calculating corner points, optical flow, e.g Pyramid lucas kanade optical flow method [6], is applied on these points in frame t to get the corresponding points in frame t+δt. Points are tracked over sequence of images using optical flow to calculate trajectories.

2.2 Remove confusing Trajectories

This step is for enhancing the clustering process. In our algorithm, detected trajectories can be classified into two categories; normal (smooth) and abnormal (non smooth) trajectories. A smoothness value of the trajectory can be defined by speed and direction. In our method, speed measurement is used. For each three consecutive points $P0$, $P1$, $P2$ the smallest distant between the first and the third point is calculated divided by the summation of the length of segment $P0P1$ and $P1P2$. Smoothness of speed is measured by the following equation

$$Smooth(T_j) = \frac{\sum_{i=1}^{N-2} \frac{dist(p_i,p_{i+2})}{dist(p_i,p_{i+1})+dist(p_{i+1},p_{i+2})}}{N-2} \quad (1)$$

Where P_i is the point number i in the trajectory j, N is the number of points in the trajectory and dist(P_1,P_2) is the Euclidean distance between these two points. Trajectory will be removed if the smoothness term is less than a specific threshold. The best smoothness occurs when Smooth(T_j) term is equal to one. Small Trajectories that may result from simple shaking movement in the camera caused by wind, which is common in outdoor surveillance, beside small movements in the scene background, e.g movement of tree branches, should be removed by calculating the length of the trajectory

$$Length(T_j) = \sum_{i=1}^{N-1} dist(p_i, p_{i+1}) \quad (2)$$

Where Length of T_j represents the number of pixels in the trajectory.

2.3 Initial Clustering

Clustering is an important step of our algorithm where trajectories that belongs to the same object will be obtained. The input of this step is the vector T_r

$$Tr = \{Tr_1, Tr_2, \ldots, Tr_L\}$$

Where Tr_j is the j^{th} trajectory and L is the number of trajectories. A single trajectory Tr_j is often represented as a path, which consists of sequence of points. It can be denoted as

$$Tr_j = \{P_j^1, P_j^2, P_j^3 \ldots, P_j^N\}$$

Where N is the number of points in the trajectory. Given such input data, the goal is to produce a set of clusters

$$Clust = \{C_1, C_2, \ldots, C_M\}$$

Where M is the number of moving objects. Clustering is based on three parameters by which we can decide if there is an overlap between any two trajectories or not. The first parameter is angle distance where the similarity is represented by the mean of angles between each two segments with respect to standard deviation.

$$AM = \frac{\sum_{i=1}^{N-1} A(p_a^i p_a^{i+1}, p_b^i p_b^{i+1})}{N-1} \quad (3)$$

$$AS = \sqrt{\frac{\sum_{i=1}^{N-1} (A(p_a^i p_a^{i+1}, p_b^i p_b^{i+1}) - AM)^2}{N-1}} \quad (4)$$

Where A and AM refer to Angle and Angle Mean, respectively. a and b represents the number of trajectory. AS refers to standard deviation of angle.

$$A(p_a^i p_a^{i+1}, p_b^i p_b^{i+1}) = cos^{-1} \frac{\overrightarrow{p_a^i p_a^{i+1}} \ \overrightarrow{p_b^i p_b^{i+1}}}{\|\overrightarrow{p_a^i p_a^{i+1}}\| \ \|\overrightarrow{p_b^i p_b^{i+1}}\|} * \frac{180}{\pi} \quad (5)$$

The maximum similarity occurs when the result is equal to zero. The angle will be normalized by determining a specific threshold. If the result of AM is greater than the specific threshold those two trajectories are not similar. The second parameter is the difference between displacements of each two lines in two consecutive frames.

$$DM = \frac{\sum_{i=1}^{N-1} dist(p_a^i, p_a^{i+1}) - dist(p_b^i, p_b^{i+1})}{N-1} \quad (6)$$

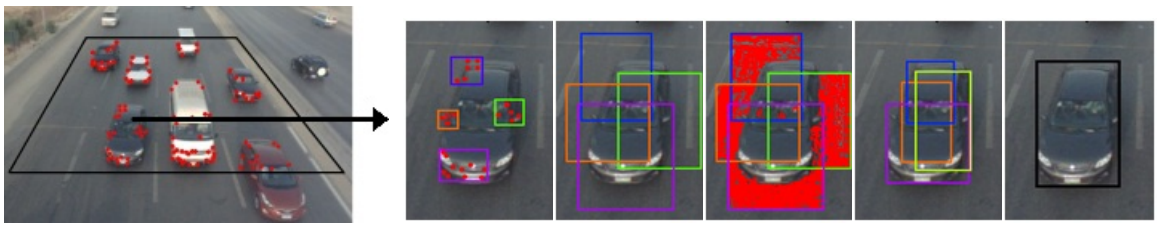
$$DS = \sqrt{\frac{\sum_{i=0}^{N-1} ((dist(p_a^i, p_a^{i+1}) - dist(p_b^i, p_b^{i+1})) - DM)^2}{N-1}} \quad (7)$$

Where DM and DS refer to Displacement Mean and Standard deviation, respectively. The last parameter is the parallel similarity. Distances between each two points in the same frame between each two trajectories are calculated.

$$PM = \frac{\sum_{i=1}^{N} dist(P_a^i, P_b^i)}{N} \quad (8)$$

$$PS = \sqrt{\frac{\sum_{i=1}^{N} ((dist(P_a^i, P_b^i)) - AM)^2}{N}} \quad (9)$$

Where PM and PS refer to Parallel Mean and standard deviation, respectively. The average distance



(a) (b) (c) (d) (e) (f)

Figure 1. (a) Extracting suitable corner points, (b) Clusters obtained from the initial clustering step, (c) Rectangles S_i around the initial clusters C_i, (d) Red points represent the negative score, (e) The sub-rectangles that have maximum sum and (f) Bounding box that merges all intersecting rectangles

between each of the values will be determined. The result of PS will be normalized by determining a specific threshold. if the result is greater than this threshold, it is considered to be non similar. Finally, the similarity between each two trajectories is calculated by averaging the three parameters discussed above.
To enhance the computational speed, rather than comparing all trajectories to each others, Histogram based on the primary angle from the horizontal axis of all trajectories was first obtained [7]. Trajectories that have the same angle or close to it will only be compared.

2.4 Adaptive Background Construction

Background construction is an important step for detecting moving objects. Since background changes with time due to illumination and background variation, in this section an adaptive background construction method is applied based on frame differencing. Our claim is that the occurrence of background pixels values in non-moving parts is repeated more than any other value. A global 2D-array **BK** with the same size of the input video frame size representing the background, employing the HSV color system, is presented. Each element in **BK** represent a pair of values H(the Hue of the pixel color) and C(counter representing the confidence of H). H and C will be updated as follows

- The initial values of H and C are, $C = 0$ and $H = 0$
- For each pixel of the non-moving parts.
  1. If $C = 0$, then $H = hue$ of the current pixel color value, and increase C by 1.
  2. If $C > 0$ and $H \neq hue$ of the current pixel color value, decrease C by 1.
  3. If $0 < C < C_{max}$ and $H = hue$ of the current pixel color value, increase C by 1 and set H = Hue of current frame, where C_{max} is a constant representing the maximum confidence.

These steps will be applied on all frame sequences to detect any shadow changes (day/night) and variations in the background of the scene.

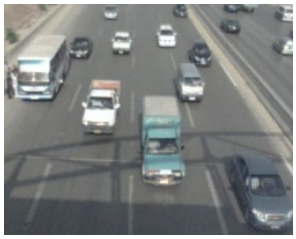
2.5 Extracting bounding boxes around vehicles

For each cluster C_i obtained from section 2.3 as shown in figure 1(b) an initial rectangle I with size $h*w$ will be generated where the center of I is the same as the center of C_i as shown in figure 1(c). Each element of I will take a negative score if the corresponding pixels is background as indicated by **BK** and positive score otherwise, as shown in figure 1(d). We aim to obtain the sub-rectangle S_{max} of I that covers the largest part of the moving object as shown in figure 1(e). It is worth to note that there are $h^2 * w^2$ possibilities to obtain sub-rectangles inside I. Consider a matrix S, having h rows and w columns whose elements $s(i, j)$ retain the sum of the elements of the sub-rectangle $(1,\ 1,\ i,\ j)$ where $i = 1$ to h and $j = 1$ to w. Using the matrix S we are able to compute the sum of the elements of the current sub-rectangle $(i,\ j,\ k,\ l)$ in order of one, $O(1)$. The Algorithm that calculates the matrix with the maximum sum is based on the Maximum Interval problem which is the one-dimensional array version of the maximum sub-rectangle problem. Using dynamic programming we can solve the Maximum Interval problem in order of n where n is the number of elements in the interval, so, the same algorithm is applied for solving the maximum sub-rectangle problem in order of $h^2 * w$ time complexity, $O(h^2 * w)$. Finally, the bounding box that bounds the vehicle shown in figure 1(f) is obtained by joining all sub-rectangles S_i which are representing all clusters contributing in this moving object. It is supposed that the rectangles that share the same vehicle have a common intersection area. Those rectangles are joined using a disjoint set data structure [8] to obtain exactly one rectangle covering the vehicle. The complexity of joining those rectangles is $O(n)$ where n is the number of rectangles.

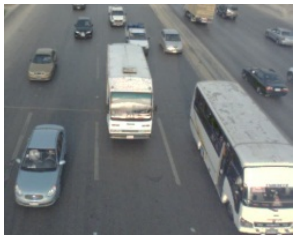
3 Experimental results and analysis

Three experiments were conducted on our newly collected dataset [9] representing vehicles in very crowded scenes taken in different places and at different times. This dataset has three videos, $NUvideoI$, $NUvideoII$ and $NUvideoIII$, where vehicles are not following any lanes or fixed pattern. $NUvideoI$ was recorded on a highway road during day time as shown in figure 2(a). Camera was installed on a six meters high. Number of vehicles passed through a predefined virtual zone was 3337 in 45 minutes. $NUvideoII$ and $NUvideoIII$ were recorded on ring road at two different times with different shadow conditions as shown in figure 2(b) and 2(c). Camera was installed on an eight meters high. Number of vehicles in ten minutes was 1141 and 1168, respectively. The first experiment was applied on $NUvideoI$ where camera was attached with panda board 1M RAM and CPU Dual-core ARM $Cortex^tm$ A9 $MPCore^tm$ with Symmetric Multiprocessing (SMP) at upto 1.2 GHz as shown in figure 3. The algorithm counted 3366 vehicles and recorded the

(a)



(b)



(c)

Figure 2. (a) *NUvideoI*, (b) *NUvideoII* and (c) *NUvideoIII* dataset samples

time stamp of the entrance and exit of each vehicle which passed through the virtual zone and achieved an accuracy of 99.2% with a real time computation.

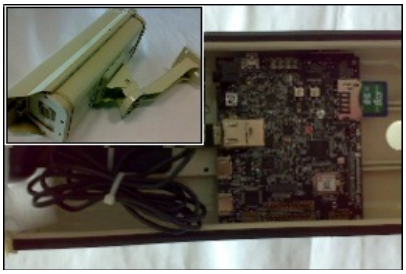


Figure 3. Camera attached to panda board

The second and third experiments were applied on *NUvideoII* and *NUvideoIII*. These experiments were run on an Intel core(TM)2 Duo T6500 2.1GHz PC with 4GB RAM. The algorithm counted 1151 and 1196 vehicles which achieve an accuracy of 99.13% and 97.60%, respectively. The average process time in our experiment was equal to 0.1 seconds per 5 frames which promotes our algorithm for real time applications. It is concluded that the algorithm count and track vehicles in real time with limited computational resources. Practical deployments shown in table 2 confirm these excellent properties. An additional experiment was performed on a video representing an aerial view of vehicles in Paris, as shown in figure 4. Consistent results were obtained, were an accuracy of 100% was achieved.

Table 2. Results of our method applied on our collected videos where T is the true number of vehicles and E is the estimated number calculated by our algorithm

Video name	T	E	Accuracy
NUvideoI	3337	3366	99.20%
NUvideoII	1141	1151	99.13%
NUvideoIII	1168	1196	97.60%

4 Conclusions

In this paper, we presented a novel algorithm for detecting and tracking moving vehicles in regular and very crowded scenes. Our algorithm is illumination and scale invariant where vehicles are not following any discipline or pattern. The proposed method was applied to our collected dataset representing very crowded scenes. An accurate and consistent results were obtained. In addition, this method has an inexpensive computational and storage cost which promotes it for real time applications.

Figure 4. Trajectories for an aerial view for vehicles in Paris

References

[1] Vibha L, Venkatesha M, Prasanth G Rao, Suhas N, P Deepa Shenoy, Venugopal K R, L M Patnaik: "Moving Vehicle Identification using Background Registration Technique for Traffic Surveillance". IMECS, 2008 Vol. I. March 19-21, 2008, Hong Kong.

[2] Chengcui Zhang, Shu-Ching Chen, Mei-Ling Shyu, Srinivas Peeta:"Adaptive Background Learning for Vehicle Detection and SpatioTemporal Tracking". ICICS, 2003 Vol. 2, pp. 797-801, December 15-18, 2003, Singapore.

[3] Lei Gao, Chao Li, Ting Fang, Zhang Xiong: "Vehicle Detection Based on Color and Edge Information". ICIAR, 2008 Vol. 5112, pp. 142-150, June 25-27, Portugal.

[4] Alberto Broggi, Pietro Cerri, Pier Claudio Antonello: "Multi-Resolution Vehicle Detection using Artificial Vision". IEEE Intelligent Vehicles Symposium - 2004. pp. 310314, June 14-17, Parma.

[5] C. Harris and M. Stephens: "A combined corner and edge detector", Proceedings of The Fourth Alvey Vision Conference (Manchester, UK), pp. 147-151, 1988.

[6] Bruce D. and Lucas Takeo Kanade: "An Iterative Image Registration Technique with an Application to Stereo Vision, Imaging Understanding Workshop". IJCAI, 1981 Vol. 2, pp. 674-679, April.

[7] Rizwan Chaudhry and Avinash Ravichandran and Gregory D. Hager and René Vidal: "Histograms of Oriented Optical Flow and Binet-Cauchy Kernels on Nonlinear Dynamical Systems for the Recognition of Human Actions". CVPR, 2009, pp. 1932-1939, June 20-25, Miami, Fl.

[8] T. H. Cormen and C. E. Leiserson and R. L. Rivest and C. Stein: "Introduction to Algorithms - 3rd ed.", MIT Press and McGraw-Hill, 2009, ISBN 978-0-262-03384-8.

[9] `http://www.cis.nileu.edu.eg/cv/papers/dataset-1/`