Real time vehicle detection and tracking on multiple lanes

Kristian Kovačić, Edouard Ivanjko, Hrvoje Gold
Department of Intelligent Transportation Systems
Faculty of Transport and Traffic Sciences, University of Zagreb
Vukelićeva 4, HR-10000 Zagreb, Croatia
kkovacic@fpz.hr, edouard.ivanjko@fpz.hr, hrvoje.gold@fpz.hr

ABSTRACT

Development of computing power and cheap video cameras enabled today's traffic management systems to include more cameras and computer vision applications for transportation system monitoring and control. Combined with image processing algorithms cameras are used as sensors to measure road traffic parameters like flow, origin-destination matrices, classify vehicles, etc. In this paper development of a system capable to measure traffic flow and estimate vehicle trajectories on multiple lanes using only one static camera is described. Vehicles are detected as moving objects using foreground and background image segmentation. Adjacent pixels in the moving objects image are grouped together and a weight factor based on cluster area, cluster overlapping area and distance between multiple clusters is computed to enable multiple moving object tracking. To ensure real time capabilities, image processing algorithm computation distribution between CPU and GPU is applied. Described system is tested using real traffic video footage obtained from Croatian highways.

Keywords

Multiple object detection, intelligent transportation system (ITS), vehicle detection, vehicle tracking, algorithm parallelization, trajectory estimation

1 INTRODUCTION

Video sensors or cameras combined with image processing algorithms are more and more becoming the approach to today's road traffic monitoring and control. They have become robust enough for continuous measurement of road traffic parameters [Con14]. From the obtained video footage high level traffic information can be extracted, i.e. incident detection, vehicle classification, origin-destination (OD) matrix estimation, etc. This information is crucial in advanced traffic management systems from the domain of intelligent transportation systems (ITS).

In order to provide high level traffic information using a computer vision system, vehicle detection has to be implemented first. Most often used current approaches are based on: (i) foreground / background (Fg/Bg) image segmentation methods where moving (foreground) objects are separated from static (background) objects as described in [Con12a] and [Con07]; (ii) optical flow computation of specific segments (moving clusters) in an image [Con00]; and (iii) vehicle detection algorithms based on the Hough method [Con12b] or on Haar-like features [Con06]. For real time systems most suitable approach is the Fg/Bg image segmentation method because of its low computational demands. Low computational demand is important in creating real time systems especially for multiple object detection and tracking systems.

After a vehicle in an image has been detected, its movement is tracked by storing its pose (position and orientation) and its pose change for each time segment. Using the saved vehicle poses and pose changes, its trajectory can be estimated. Trajectory estimation can be performed using various approaches (mostly a prediction/correction framework with odometry as the motion model) and its basic task is to determine a mathematical function which will best describe given set of points (vehicle poses) in 2D space. Vehicle trajectory estimation can be separated into following three parts: (i) finding a function f which will best describe given set of points in 2D space; (ii) transformation of parameters from 2D to 3D space model; and (iii) computing movement parameters such as vehicle direction, velocity and acceleration/deacceleration in 3D space model [Pon07].

Typical commercial computer vision based traffic monitoring systems use one camera per lane to ensure accurate and robust traffic parameters measurement. This presents a drawback since many cameras are needed for roads with multiple lanes which makes such systems expensive. This paper tackles the mentioned problem by modifying the image processing part to enable vehicle detection and tracking on multiple lanes in real time. Image processing is parallelized and its execution is distributed between the CPU and GPU. So, only one camera per road is needed making such systems simpler and cheaper.

This paper is organized as follows. Second section describes the approach used in this paper. Third section describes the vehicle tracking algorithm. Following fourth section presents obtained results. Paper ends with conclusion and future work description.

2 VEHICLE DETECTION

Basic work flow of the proposed system consists of four main parts: (i) image preprocessing; (ii) Fg/Bg image segmentation; (iii) pixel clusterization; and (iv) multiple object tracking. The task of the image preprocessing part is to enhance the image imported from a video stream using a blur filter. After preprocessing, the image is passed through Fg/Bg image segmentation to separate foreground (moving) from background (static) segments in the image. This method is based on creating a background model from a large number of preprocessed images and comparing it with the new preprocessed image. The Fb/Bg segmentation result is then passed through pixel clusterization which computes location of each object (vehicle) in a scene and tracks its trajectory through consecutive images (frames). Final part for multiple object tracking of the proposed system performs also vehicle counting using markers defined in the scene.

In order to accurately detect, track and count vehicles obtained traffic video needs to satisfy following requirements: (i) camera perspective in the video footage must be constant over time (fixed mounted camera); (ii) all moving objects in the scene are vehicles (system does not perform classification between detected moving objects); and (iii) traffic video must not be previously preprocessed with image enhancing algorithms (for example auto-contrast function which can degrade the system accuracy).

Algorithms for image downsampling, image preprocessing and Fg/Bg segmentation are executed entirely on GPU. Today's GPUs enable high parallelization support for mentioned algorithms and can reduce execution time for basic image processing operations. Part of the system related to image preprocessing is performed with a single render call. In Fg/Bg segmentation, creation of background model is performed with 8 render calls to process one frame. Moving object detection (comparison of background model and current image) is performed with one single render call. Algorithms for pixel clusterization and vehicle tracking are not suitable to run on GPU because of their structure and therefore are run entirely on CPU. Pixel clusterization and vehicle tracking algorithms are made of many dynamic nestings and IF clauses, and too complex to run on GPU in parallel. The mentioned image processing workflow and distribution of computations between the CPU and GPU is given in Fig. 1.

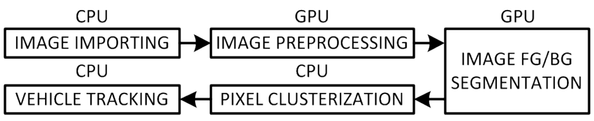

Figure 1: CPU/GPU computation distribution.

2.1 Image Preprocessing

Every image imported from the road traffic video footage contains a certain percentage of noise. Noise complicates the vehicle detection process and significantly reduces the accuracy of the described system so it needs to be minimized. For noise reduction, blur filters are commonly used. They reduce the number of details in the image including noise. In the proposed system a 4×4 matrix Gaussian blur filter is used for noise reduction. Work flow of image preprocessing is given in Fig. 2.

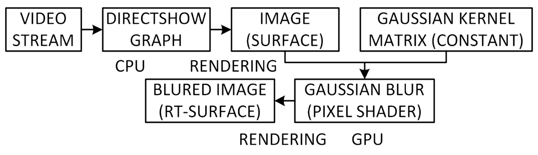
VIDEO STREAM
DIRECTSHOW GRAPH
IMAGE (SURFACE)
GAUSSIAN KERNEL MATRIX (CONSTANT)
CPU
RENDERING
BLURED IMAGE (RT-SURFACE)
GAUSSIAN BLUR (PIXEL SHADER)
RENDERING
GPU

Figure 2: Image preprocessing work flow.

2.2 Foreground / Background Image Segmentation

After the imported image has been successfully preprocessed, Fg/Bg image segmentation is performed as shown in Fig. 3. As mentioned, this process consists of creating a background model of the scene and comparing computed background model with the latest image imported from the video [Con13]. The background model is obtained using the following equation:

$$BG_t = BG_{t-1} + \left\lfloor \frac{\sum_{i=1}^{n} sign(I_i - BG_{t-1})}{n} \right\rfloor , \qquad (1)$$

where BG_t represents value of a specific pixel in the background model for current frame, BG_{t-1} is value of a specific pixel in the background model for the previous frame, I_i is a value of the certain pixel in i^{th} image, and n is the number of stored images.

By comparing mentioned pixels in imported images, every pixel in the currently processed image can be classified. If difference between current image pixel value and background model pixel value is larger than specified threshold constant, pixel is classified as a part of a foreground object. Otherwise it is considered as a part of the background model. Result of preprocessing and Fb/Bg image segmentation is given in Fig. 4.

2.3 Pixel Clusterization

After each pixel in the image is classified as part of a foreground object or as a segment of the background model, pixel clusterization needs to be performed. Used approach is based on marking all adjacent pixel that have the same pixel value as a part of a specific cluster. Afterward, all pixels within the same cluster are counted and their minimum and maximum values of x and y coordinates are found. With mentioned information clusters can be represented as rectangles. Rectangle center is used as the cluster center.

In the proposed system, pixel clusterization is performed only on foreground pixels. Additionally, all clusters that do not contain enough pixels related to them are discarded and excluded from further processing.

3 VEHICLE TRACKING

The clustering part returns a large number of clusters i.e. objects or possible vehicles. To sort out objects

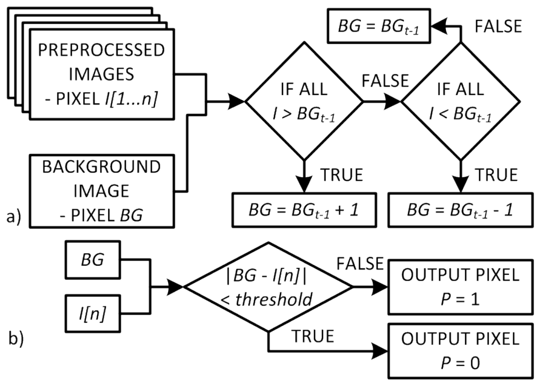

Figure 3: Fg/Bg image segmentation work flow: a) background model creation, and b) background model and current image comparison.

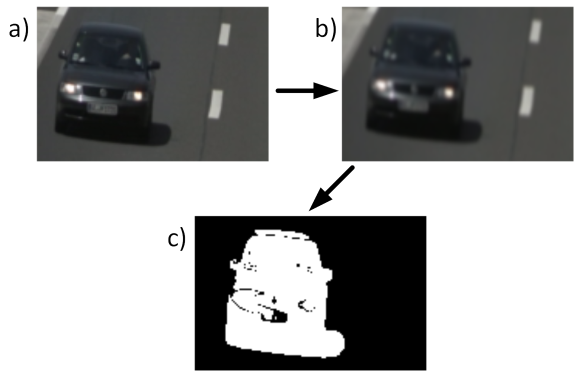

Figure 4: Original image (a) passed through preprocessing algorithm (b) and Fg/Bg segmentation (c).

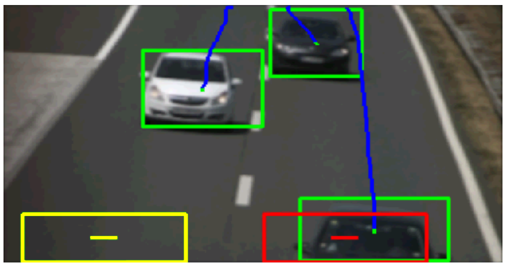

Figure 5: Vehicle tracking and counting on two lanes.

that are not vehicles, filtering is applied. In the proposed system, spatio-temporal tracking of objects in a scene is used for filtering. Every currently tracked object in the scene is compared with each cluster detected in the current image. Cluster that does not match with any of the previously detected objects is set as a new object. Cluster matching is performed by searching for the largest weight factor related to the cluster and specific object. Cluster will be assigned to the object with highest weight factor. Appropriate weight factor w is computed using the following equations:

$$w_{dist} = 1 - \frac{d - d_{min}}{d_{max} - d_{min}}, \tag{2}$$

$$w_{area} = 1 - \frac{a - a_{min}}{a_{max} - a_{min}}, \tag{3}$$

$$w_{cover} = \frac{a_{is}}{max(a_{obj}, a_{cl})}, \tag{4}$$

$$w = \frac{w_{dist} + w_{area} + w_{cover}}{3}, \tag{5}$$

where d is distance between location of the specific cluster and estimated object location, d_{min} and d_{max} are minimum and maximum distance between all clusters and processed object, a is difference between the cluster area (size) and estimated object area, a_{min} and a_{max} are minimum and maximum difference between all clusters area and estimated object area respectively, a_{is} is intersection area between cluster and object, a_{obj} is area of the object, and a_{cl} is the processed cluster area.

To compute the distance between location of the specific cluster and estimated object location their geometric centers are used. Cluster and object area are computed as their surrounding bounding box area.

4 EXPERIMENTAL RESULTS

The proposed system has been tested using real world traffic footage captured on a highway with two lanes near the city of Zagreb in Croatia. Camera was mounted above the highway and passing vehicles were recorded using a top view camera perspective as given in Fig. 5. Duration of the test video was 10 [min]. Obtained original video resolution is 1920×1080 pixels (RGB).

Approach		Vehicle count		
		Total	Lane	
			Left	Right
Overlap check	Hits	126	65	61
	FP / FN	0/6	0/5	0/1
	Accuracy	95.6%	92.9%	98.4%
Trajectory check	Hits	129	68	61
	FP / FN	1/4	0/3	1/1
	Accuracy	96.2%	95.8%	96.8%
True vehicle count		132	70	62

Table 1: Counting results of the proposed system.

For experimental results, two approaches for vehicle counting were tested. Both are based on markers (virtual vehicle detectors). Markers are placed in bottom part of the scene on each lane as shown in Fig. 5 with yellow and red rectangles. Yellow color denotes an inactive marker and red color an activated marker. Edges of markers are perpendicular to the image x and y axis. When a vehicle passes through marker and a hit is detected, counter for that marker is incremented. First approach checks if an object is passing through marker with its trajectory and second approach performs check if an intersection between marker and object exists. Both approaches discard all objects whose trajectory direction is outside of a specific interval. In performed test, all moving objects need to have their direction between $90-270$ [°] in order not to be discarded. Objects also need to be in the scene for more than 30 frames. Value of the threshold constant used in Fg/Bg segmentation method is 10 and number of consecutive images used when creating background model (n) is 105. Blue lines in Fig. 5 represent computed vehicle trajectory. Experimental results are given in Tab. 1. FP represents false positive and FN represents false negative hits. True vehicle count is acquired by manually counting all passed vehicles.

In Fig. 6, execution time is given for various resolutions tested on Windows 7 (64bit) computer with CPU Intel Core i7 - 2,4 GHz, GPU NVIDIA Quadro K1000M video card and 8 GB RAM. In the experimental testing, both approaches (overlap and trajectory check) for vehicle counting had the same execution time. Tested application was compiled without optimization and with GPU support. Video importing was performed by Microsoft Direct Show framework.

From the acquired results it can be concluded that real time vehicle detection can be performed on SVGA and lower resolutions using a standard PC computer. On SVGA resolution, 37 [ms] is required to process a single frame. This enables maximum frame rate of 27 [fps]. At QVGA resolution, 52 [fps] can be achieved with 19 [ms] required to process a single frame. It can also be concluded that approach with trajectory check gives better results (accuracy) than approach with overlap check. In second testing application was fully optimized by compiler and FFMPEG framework was used for video importing. Achieved results in the second testing show that application with GPU support and capability of executing on 8 threads by CPU can achieve much faster execution. At SVGA resolution, execution time for each frame was 17 [ms] which enables processing of 58 [fps]. At QVGA resolution, execution time was 7 [ms] which gives capability of processing 142 [fps]. The highest resolution in which application can still perform real time image processing is HD720 resolution with frame rate of 35 [fps]. In Fig. 8, ratio

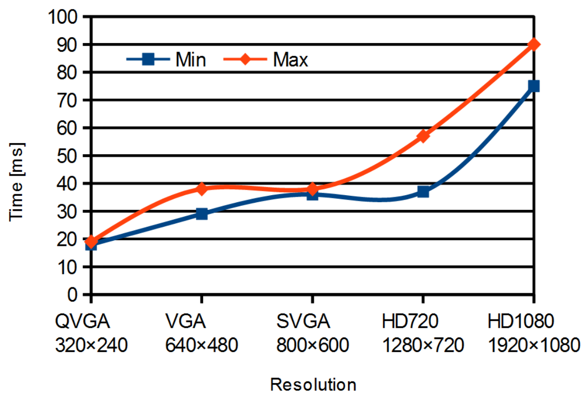

Figure 6: Comparison of execution time of the proposed system.

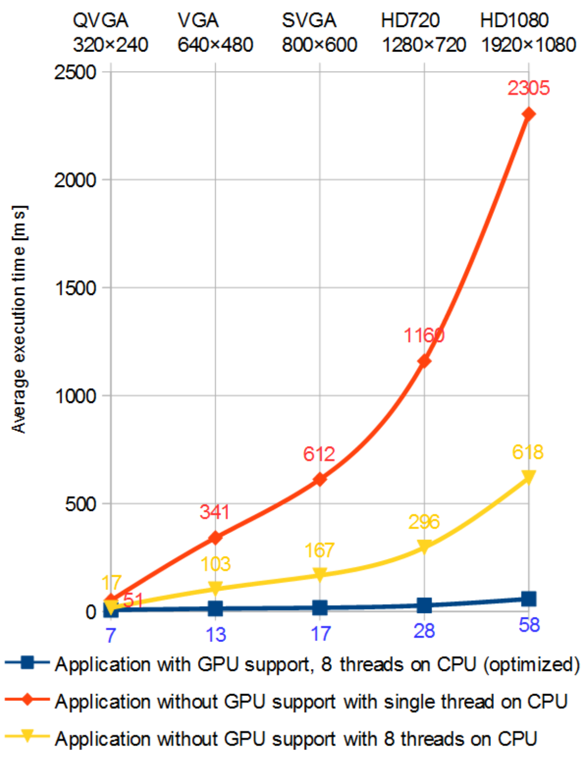

Figure 7: Comparison of application execution time with and without GPU and multi-thread capabilities.

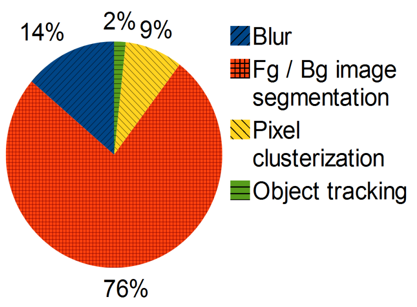

Figure 8: Execution time distribution between applied image processing tasks.

between execution time of a specific image processing task and overall execution time of the implemented application is given.

For the purpose of determining the efficiency of the proposed system with GPU and CPU multi-thread support, implemented application was modified to run without GPU support (only on CPU). At SVGA resolution application execution time for each frame was 167 [ms] (5 [fps]) with multi-thread capability (8 working threads) and 612 [ms] (1 [fps]) without multi-thread capability (single thread). In Fig. 7, comparison of previously mentioned version of implementations is given. All developed version of applications provide same results regarding vehicle detection accuracy (number of hits, FP and FN detections).

5 CONCLUSION AND FUTURE WORK

In this paper a system for vehicle detection and tracking on multiple lanes based on computer vision is proposed. Developed system uses only one camera to detect and track vehicles on a road with multiple lanes. First testing results are promising with vehicle detection accuracy of over 95%. Methods used in the proposed system are easy to implement and to parallelize. They are also suitable for executing in GPU and CPU multi-thread environments enabling real-time capabilities of the proposed system. Implemented vehicle trajectory tracking currently does not use any vehicle dynamics and predicts the tracked vehicle pose for one succeeding frame only.

From the execution time distribution analysis results can be concluded that algorithm for Fg/Bg image segmentation is the most slowest part of the application and it consumes 76% of the total application execution time. Further optimization of this algorithm would lower system requirements of the application and increase maximum resolution at which real time image processing can be achieved. It would also allow other complex algorithms (vehicle classification, license plate recognition, etc.) to be implemented into the system and executed in real time.

Future work consists of developing a multiple object tracking system which would estimate vehicle trajectory based on a vehicle model with dynamics included. Additionally, it is planned to develop a system which will perform vehicle classification and therefore separate vehicles by their type. This will enable detection and tracking of vehicles that are coming to standstill on crossroads.

6 ACKNOWLEDGMENTS

This work has been supported by the IPA2007/HR/16IPO/001-040514 project “VISTA - Computer Vision Innovations for Safe Traffic” which is co-financed by the European Union from the European Regional and Development Fund and by the EU COST action TU1102 - “Towards Autonomic Road Transport Support Systems”. Authors wish to thank Marko Ševrović and Marko Šoštarić for obtaining the road traffic video, and to Sanja Palajsa for preparing the test video.

7 REFERENCES

[Con14] Pirc, J., Gostiša, B. Comparison between license plate matching and bluetooth signature reidentification systems for travel time estimation on highways, in Conf. proc. ISEP, 2014.

[Con06] Bai, H., and Wu, J., and Liu, C. Motion and haar-like features based vehicle detection, in Conf. proc. MMM, 2006.

[Con12a] Braut, V., and Čuljak, M., and Vukotić, V., and Šegvić, S., and Ševrović, M., and Gold, H. Estimating OD matrices at intersections in airborne video - a pilot study, in Conf. proc. MIPRO, pp.977-982, 2012.

[Con13] Kovačić, K., and Ivanjko, E., and Gold, H. Computer vision systems in road vehicles: a review, in Conf. proc. CCVW, pp. 25-30, 2013.

[Pon07] Ponsa, D., and López, A. Vehicle trajectory estimation based on monocular vision, Pattern recognition and image analysis, Vol. 4477, pp. 587-594, 2007.

[Con00] Stanisavljević, V., and Kalafatić, Z., and Ribarić, S. Optical flow estimation over extended image sequence, in Conf. proc. MELECON, Vol. 2, pp. 546-549, 2000.

[Con07] Tanaka, T., and Shimada, A., and Arita, D., and Taniguchi, R.I. A fast algorithm for adaptive background model construction using parzen density estimation, in Conf. proc. AVSS, pp. 528-533, 2007.

[Con12b] Xiaoli, H., and Dianfu, H., and Xing, Y. Vehicle window detection based on line features, in Conf. proc. IMSNA, Vol. 1, pp. 261-263, 2012.