Real-Time Traffic Video Analysis Using Intel Viewmont Coprocessor

Seon Ho Kim[1], Junyuan Shi[2], Abdullah Alfarrarjeh[3], Daru Xu[2], Yuwei Tan[3], and Cyrus Shahabi[1,2,3]

[1] Integrated Media Systems Center, University of Southern California, CA, USA
[2] Department of Electrical Engineering, University of Southern California, CA, USA
[3] Department of Computer Science, University of Southern California, CA, USA
{seonkim,junyuans,alfarrar,daruxu,yuweitan,shahabi}@usc.edu

Abstract. Vision-based traffic flow analysis is getting more attention due to its non-intrusive nature. However, real-time video processing techniques are CPU-intensive so accuracy of extracted traffic flow data from such techniques may be sacrificed in practice. Moreover, the traffic measurements extracted from cameras have hardly been validated with real dataset due to the limited availability of real world traffic data. This study provides a case study to demonstrate the performance enhancement of vision-based traffic flow data extraction algorithm using a hardware device, Intel Viewmont video analytics coprocessor, and also to evaluate the accuracy of the extracted data by comparing them to real data from traffic loop detector sensors in Los Angeles County. Our experimental results show that comparable traffic flow data to existing sensor data can be obtained in a cost effective way with Viewmont hardware.

Keywords: Video Analysis, Intel Viewmont, Traffic Flow Data Inference.

1 Introduction

As a robust traffic monitoring system becomes an urgent need to improve traffic control and management [1], many techniques have been proposed for traffic flow data extraction. Traffic flow data such as the count of passing vehicles and their speeds can be obtained from various devices like loop detector sensors, radars, and infrared detectors, to name a few [2]. The most widely used sensor type is loop detector installed on the surface of road to detect the movement of passing vehicles over it. One of the main shortcomings of these under-pavement traffic loop detectors is that they are expensive to install and maintain. Moreover, they cannot be replaced or fixed without disturbing traffic. Therefore, researchers have been studying computer vision-based techniques to extract traffic flow data from traffic monitoring cameras. Video sensor requires less installation and maintenance cost. Furthermore, video sensors can monitor a large area in multiple lanes and be also useful in analyzing vehicle classification and accident detection as well as extracting speed and vehicle counting. However, there are two challenges with this approach. First, since image-processing techniques are CPU-intensive, for real-time traffic flow data extraction from traffic

A. Madaan, S. Kikuchi, and S. Bhalla (Eds.): DNIS 2013, LNCS 7813, pp. 150–160, 2013.

videos, accuracy may be sacrificed. Second, the traffic measurements extracted from cameras have not been fully validated with real dataset.

This paper investigates the solutions of the above two challenges. First, we propose a vision-based traffic flow data extraction algorithm to support multi-channel live traffic video streams in real-time utilizing newly developed Viewmont video analytics prototype coprocessor by Intel, Corp. Second, to validate the effectiveness of the vision-based algorithm, we compare our results with the real data from loop detector sensors in Los Angeles County. The Integrated Media Systems Center at the University of Southern California is working with the Los Angeles Metropolitan Transportation Authority (LA-Metro) to develop the data management and analytics systems that will form the basis for building a large-scale transportation data warehouse for heterogeneous transportation data (e.g., traffic flows data recorded by loop detectors and videos from CCTV cameras, etc.). Through LA-Metro, we are acquiring real traffic flow data from thousands of loop detector sensors all around the Southern California.

Our experimental results demonstrate that our proposed algorithm with dedicated video coprocessor is capable of processing multiple video channels simultaneously in real-time so a cost effective implementation of vision-based traffic flow data extraction system would be feasible. The results also show that our vehicle counting and speed estimation are comparable with those from loop detectors in Los Angeles County.

The remainder of this paper is organized as follows. In Section 2, we present background and related work. Section 3 describes our real-time video analysis algorithm using Intel Viewmont. Experimental results and efficiency analysis are reported in Section 4. Finally, we conclude our work with future directions in Section 6.

2 Background and Related Work

2.1 Intel Viewmont Coprocessor

The Viewmont is a PCI-Express based video/image analytics coprocessor that provides simultaneous real-time video encode and video analytics capability. Through its software development kit (SDK) it supports various functionalities to perform video analytics with minimum CPU utilization while achieving a high performance. For instance, the convolve operation performance could be up to 240~400 MPixels/sec.

The coprocessor is equipped with a set of programming APIs for application developers to communicate with the processor via this programming interface. There is also a software-based coprocessor simulator which simulates the hardware coprocessor's analytics capabilities and results.

2.2 Related Work

A visual surveillance system is required to be fast in processing with low cost and high reliability [3]. Current video based traffic tracking system can be categorized into three classes: tripline system, closed-loop tracking and data association tracking [4]. Tripline systems allow the user to define a number of detection zones in the field of view of the video camera. When a vehicle crosses one of these zones, it is identified by noting changes in the pixels caused by the vehicle relative to the roadway in

the absence of a vehicle. Closed-loop tracking system is an extension of the tripline approach that permits vehicle detection along larger roadway sections. The closed-loop systems track vehicles continuously through the field of view of the camera. [5] is an implementation of closed-loop system, which is capable of tracking multiple targets. Data association tracking systems identify and track a particular vehicle or a group of vehicles by locating unique connected areas of pixels. [6] utilizes the idea with Kalman prediction. Our proposed approach depends on tripline system theme.

A significant amount of work has been investigated to establish mathematical or empirical traffic models. A 3D (flow, speed, and concentration) traffic model has mostly used in the area [7]. It is believed that by measuring the three parameters simultaneously, the current road condition can be obtained [2]. Currently, our solution follows the traffic model in two dimensions (flow rate and speed) but it can be extended to 3D as [2] proposes.

To meet the real-time constraint of vision-based traffic system, much work has been done to improve the efficiency of algorithms. Thus, many real-time approaches have been proposed in CPU level processing [1] [11] [12]. But in this work we adopt an extra hardware to move heavy computations to a powerful coprocessor to achieve real-time processing for multiple video channels.

3 Video Analysis Algorithm

3.1 Region of Interest Initialization

Tripline framework [8] is followed to process the traffic video. Our algorithm needs to define a region of interest (ROI), which is a portion of image where actual analysis happens in order to reduce the overhead of computing-intensive video analytics process. Lane separation is also defined in this region. ROI is a rectangular area in the video frame where one can see the traffic most clearly. Within the ROI, virtual lines are also defined, which are used to detect how a vehicle passes each lane. Defining ROI and virtual lines should be done manually by a human operator but it is a one-time job at the beginning since traffic monitoring cameras are fixed.

Fig. 1. ROI with virtual lines

Figure 1 shows an example of ROI and virtual line definition. The rectangular region is the ROI and for each lane, there are two virtual lines. One of them is used for counting passing vehicles when the cars are crossing. The algorithm also records the time whenvehicle hits each line and based on this information, the vehicle speed is calculated using the passing time between two virtual lines.

3.2 Analysis Techniques

The algorithm takes traffic monitoring video (stored file or stream) and ROI configuration as input, and outputs two critical traffic flow data: the number of passing vehicles and their speeds as illustrated in Figure 2.

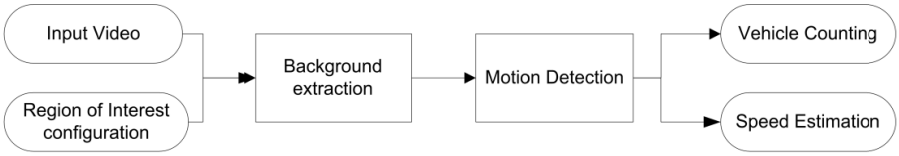

Fig. 2. Video analysis pipeline

After applying background extraction and foreground motion detection techniques to incoming video frames, moving vehicles are represented as moving blocks. Over each virtual line, moving blocks (i.e., vehicles) are detected by examining the variation of motion pixel values along time.

Background extraction technique follows the idea of frame average method [9]. Background is initially defined as the first frame of the video. Afterwards, it is compared to every new frame and updated according to the pixel value difference. Specifically, the background pixel value is moved towards the current frame by a certain amount Δ if there is pixel value difference at the location. In this way, a stable background is obtained by continuously updating after a certain number of frames. This number may vary according to the selection of Δ. A typical background updating process is demonstrated in Figure 3.

(a) (b) (c)

Fig. 3. Background extraction: (a) original background frame, (b) intermediate background, (c) final background

With background information, Difference Image (DI), is created as the absolute difference between the current frame and background image. DI is then thresholded

with a predefined value T to construct the binary Motion Image (MI). T should be chosen wisely to identify the moving object while suppressing background noise. Figure 4 shows an example of this process.

(a) (b) (c)

Fig. 4. Intermediate steps of motion detection: (a) original video frame, (b) Difference Image (DI), (c) Motion Image (MI)

Morphological operations, such as open and close, are sequentially applied to the ROI of the obtained motion image in order to suppress the noise and render the moving vehicles as simple white blocks to make the detection of moving blocks easier. The intermediate steps of this process are shown in Figure 5.

(a) (b) (c)

Fig. 5. Intermediate steps of motion detection: (a) Original video frame, (b) Motion Image, (c) Motion Image after morphological operation

3.3 Video Analysis Output

The percentage of motion points (white pixels in the binary images in Figure 5.c) over each virtual line is examined to determine whether a vehicle is passing by or not. For instance, in the case shown in Figure 5, there is a car passing by, indicated by the high percentage of motion points over the corresponding virtual line. Each lane will have one flow rate status indicator to monitor whether there is a car over the virtual line at a specific moment. To count vehicles, the algorithm tracks the percentage value p, (i.e., how much portion of virtual line meets motion points) and uses its temporal variation over a series of consecutive frames to determine the exact time when a vehicle hits a certain line. Two predefined percentages $p1$ and $p2$ ($p1 < p2$) are also used to control the flow status over the virtual lines. For each virtual line, initially p

equals to zero. When p goes greater than $p2$, the flow status is set to 1, which means there is a vehicle passing by. When p decreases to less than $p1$, the flow status is then set to 0, meaning that the vehicle has left the virtual line and the vehicle count is increased by one. In our algorithm and experiments, $p1$ and $p2$ are set to 20% and 50%, respectively.

In order to estimate the speed of moving vehicles, the algorithm adopts a two-line approach. For each lane on the road, two parallel virtual lines are manually positioned so that the distance between them is known. This distance is inferred according to the Manual on Uniform Traffic Control Devices [10] published by US Highway Administration. For each line, vehicle detection is performed independently, and the frame index in video, when a vehicle hits the virtual line, is recorded. This process results in two independent records for both parallel virtual lines. Then, the difference between frame indices of two virtual lines leads to the time difference with the knowledge of video frame rate, which is usually fixed (e.g., 30 frames per second). Provided with both the actual distance and the time difference, the speed for moving vehicle is calculated. The overall speed is obtained by averaging all detected speeds within a statistic window over all the lanes. Figure 6 illustrates the two-line approach.

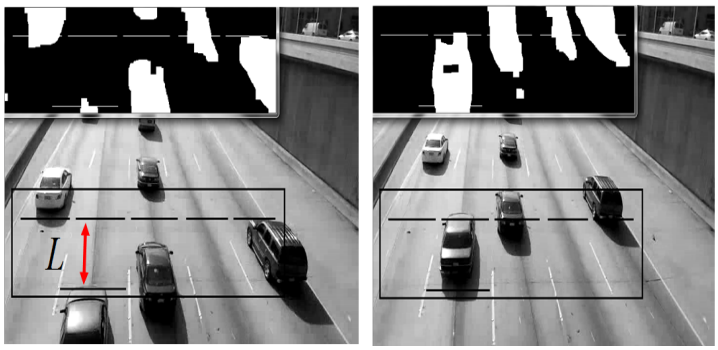

Fig. 6. Speed estimation using two-line approach

3.4 Using Viewmont Coprocessor

Viewmont facilitates implementation of various vision algorithms by providing hardwired functions as SDK so it can speed up computation of vision analysis significantly, especially when there are lots of matrix operations. In our approach, the captured frames are stored in 2D matrix structures at coprocessor hardware level. The Viewmont hardware processes these matrices very efficiently when performing arithmetic operations (i.e., summation and multiplication), convolution, and thresholding. In addition, the hardware fulfills time-consuming morphological operations such as open and close. Such computation-intensive operations are transferred from CPU into Viewmont coprocessor so that multiple video streams can be handled concurrently in real-time.

CPU captures frames from video streams and converts each frame to a predefined format before transmitting it to Viewmont coprocessor in a manner of reduced memory

pipeline. When handling multiple channels, a thread is created for each input video channel in order to utilize parallel CPU execution. Currently, Viewmont has four physical connections for up to four concurrent input video streams.

4 Experimental Results

In the experiments, we extract traffic flow data from real-time traffic monitoring videos captured along California freeways. For the evaluation of our approach, the extracted results are compared with those from the actual loop detector sensors installed on the freeways, provided by LA- Metro. The experiments include 16 test cases of different locations where loop detectors are positioned close to traffic monitoring cameras. One difficulty in testing was the fact that current traffic monitoring cameras were not installed for video analysis, but for human monitoring purpose. Thus, most cameras do not have appropriate angles to be applied for video analysis. Moreover, the locations of cameras and loop detectors are quite different so it is not easy to directly compare traffic flows on different locations. We carefully selected 16 locations where the locations of camera and loop detector are close to make the comparison meaningful. Furthermore, these cases include various different situations and environments (i.e., cloudy, rainy, or sunny weather, sparse and heavy traffic) to evaluate our algorithm. The test videos were recorded during daytime. One example of test cases is illustrated in Figure 7.

(a) (b)

Fig. 7. Example of a good test case: (a) camera (blue dot) and loop detector sensor (red dot) are located very close, (b) good camera angle and low traffic condition

4.1 Accuracy Analysis

The algorithm produces two outputs: count of passing vehicles and their average speed for a certain amount of time. To verify the precision of counting vehicles, the output of our algorithm is compared with that of loop detector, and also with the ground truth (i.e., counting by human being). The error can be evaluated based on the following formula:

$$error(\%) = \frac{count - ground\ truth\ value}{ground\ truth\ value} \times 100 \quad (1)$$

The positive value of error rate means the count value is greater than the ground truth value. For the average speed output, there is no ground truth value. Thus, the difference between the output of this approach and that from loop detector sensor was reported. The smaller difference indicates that our approach can produce comparable output to sensors.

Currently the vision algorithm in our approach works fine when the camera angle is good and there is no shadow. Under these two conditions and when traffic is sparse, the averages of counting errors of our approach and loop detector sensor are 3.04%, 6.44%, respectively, as shown in Figure 8. But when traffic is heavy, the averages of counting errors are 14.69% and 8.06%, respectively. This shows that the results of this approach are good when traffic is sparse, and get worse when traffic is congested. Overlapping vehicles and their connected shadows due to back-to-back traffic can cause bigger errors in vision-based approaches. The speed results are reasonable because they are very similar to the sensor results. Under sparse traffic condition, the speed difference is 5.59 MPH while it is 4.82 MPH when traffic is heavy as shown in Figure 9.

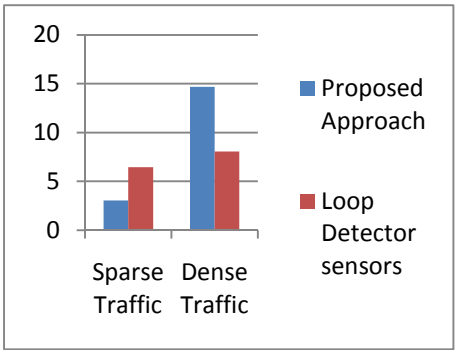

Fig. 8. Averages of vehicle counting error

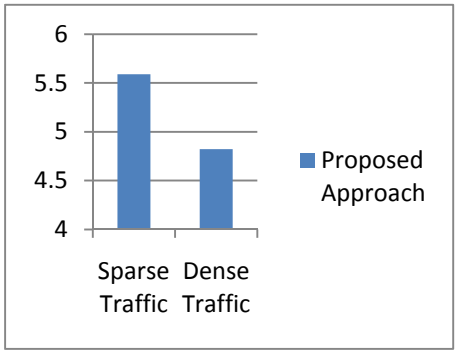

Fig. 9. Averages of speed difference between our results and sensor outputs

4.2 Processing Efficiency Analysis

Our algorithm supports three types of video inputs: stored video files, analog live video signal via BNC inputs, and streaming videos from the Internet. To evaluate the efficiency, we measured processing time (total elapsed time) of our program execution in two ways: 1) with Viewmont coprocessor and 2) with CPU alone. Intel provides a software simulator of Viewmont so that the same program can run with Viewmont or without it. In the experiments, the stored video files were three minutes long with the frame rate of 30 fps. The resolution of frames was 550*370. For analog signals, NTSC standard definition TV signals were used. For the video streams from the Internet, two different resolutions were used: 640*480 and 320*240.

In the case of stored video file, on average, the proposed approach using Viewmont took 40.6 seconds to extract traffic flow data of 180 seconds video while it took 231.97 seconds using the software simulator. Thus, it obviously shows that our algorithm running on Viewmont supports real-time video analysis while it is not when only utilizing CPU.

In the case of analog live video signal, our algorithm with Viewmont can handle four concurrent video inputs without any delay at frame level and produced traffic flow data in real-time, which means that real-time processing is guaranteed.

In the case of streaming videos from the Internet, for 640*480 video our algorithm with Viewmont took 0.39 seconds to process 50 frames while it took 2.28 seconds using the software simulator. For 320*240 resolution video streams the algorithm with Viewmont took 0.15 seconds to process 50 frames while taking 0.94 seconds using software simulator. These results confirm that the approach with Viewmont hardware is capable of real-time analysis for video streaming from the Internet. In summary, the average time costs (in milliseconds) to process one frame from different video sources are reported in Table 1.

Table 1. Time cost to process one frame

video source		**stored video file**	**live stream from Internet**	
frame resolution		**550*370**	**640*480**	**320*240**
time cost (msec)	with Viewmont	7.5	7.8	3.1
	CPU alone	43	45.6	18.8

Table 2. Time cost to process a frame at each channel using Viewmont

video source		**stored video file**	**live stream from Internet**	
frame resolution		**550*370**	**640*480**	**320*240**
time cost (msec)	1 channel	7.52	7.8	3
	2 channels	15	16.4	5.6
	3 channels	22.85	25	10.16
	4 channels	30.4	37.94	27.56

Besides, Viewmont can support multi-channel traffic video analysis concurrently utilizing the parallel execution at CPU and Viewmont coprocessor. Table 2 and 3 show the time cost of processing a single frame at each channel when multiple channels are concurrently processed, using Viewmont and the software simulator, respectively. When the frame rate of 30 fps was used, each frame should be processed within 1/30 sec for real-time traffic flow data extraction. Results in Table 2 show that our algorithm with Viewmont can support up to four concurrent videos while Table 3 clearly shows that CPU alone cannot. Also, the tables show the time cost increases almost linearly as the number of channels increases.

Table 3. Time cost to process a frame at each channel using CPU alone

video source		**stored video file**	**live stream from Internet**	
frame resolution		**550*370**	**640*480**	**320*240**
time cost (msec)	1 channel	42.78	45.6	18.8
	2 channels	82.34	95.04	39.6
	3 channels	122.79	141.97	59.4
	4 channels	164.45	191.97	79

5 Conclusion and Future Work

In this paper, we studied the performance enhancement of vision-based traffic flow data extraction using additional hardware, specifically Intel Viewmont coprocessor. In addition, extracted data were compared and evaluated with real data from loop detector sensors in Los Angeles County. The results showed that our approach produced comparable traffic data to loop detector sensors (even better under some conditions). Moreover, the performance of our approach using coprocessor was significantly better as compared with the use of CPU alone. The system was able to process in real-time multiple input video sources, including stored videos, live videos from analog inputs and video streams from the Internet.

Overall, results are encouraging but several improvements need to be included in the future. At the moment, we are working on the nighttime vehicle detection algorithm which is different from the daytime algorithm. Different methods for shadow removal are also being studied for better accuracy.

Acknowledgement. This research has been funded in part by Intel Corporation under an Intel Sponsored Research agreement, NSF grant IIS-1115153, a contract with Los Angeles Metropolitan Transportation Authority (LA Metro), the USC Integrated Media Systems Center (IMSC), and unrestricted cash gifts from Microsoft. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of any of the sponsors such as the National Science Foundation, Intel or LA Metro.

References

1. Cheung, S.-C., Kamath, C.: Robust Techniques for Back-ground Subtraction in Urban Traffic Video. In: Proc. SPIE, vol. 5308, pp. 881–892 (2004)
2. Harvey, B.A., Champion, G.H., Deaver, R.: Accuracy of traffic monitoring equipment field tests. In: IEEE Vehicle Navigation and Information Systems Conference (1993)
3. Fishbain, B., Ideses, I., Mahalel, D., Yaroslavsky, L.: Real-Time Vision-Based Traffic Flow Measurements and Incident Detection. In: Real-Time Image and Video Processing (2009)
4. Federal Highway Administration, Traffic Detector Handbook, 3rd edn., vol. I (2006)
5. Sanders-Reed, J.N.: Multi-target, Multi-Sensor Closed Loop Tracking. In: Proc. SPIE, vol. 5430 (2004)
6. Hsu, Y.C., Jenq, N.H.: Multiple-Target Tracking for Crossroad Traffic Utilizing Modified Probabilistic Data Association. In: Acoustics, Speech and Signal Processing (2007)
7. Gartner, N.H., Rathi, A.J., Messer, C.J. (eds.): Revised Monograph on Traffic Flow Theory: A State-of-the-Art Report. Special Report by the Transportation Research Board of the National Research Council (2005)
8. Vandervalk-Ostrander, A.: AASHTO Guidelines for Traffic Data Programs, 2nd edn. (2009)
9. Wang, G., Xiao, D., Gu, J.: A Robust Traffic State Parameters Extract Approach Based on Video for Traffic Surveillance. In: IEEE International Conference on Automation and Logistics (2008)
10. Highway Administration, U.S.: Manual on Uniform Traffic Control Devices, http://mutcd.fhwa.dot.gov/htm/2003r1r2/part3/part3a.htm#section3A05
11. Lin, C.-P., Tai, J.-C., Song, K.-T.: Traffic Monitoring Based on Real-time Image Tracking. In: IEEE International Conference on Robotics and Automation (2003)
12. Batista, J., Peixoto, P., Fernandes, C., Ribeiro, M.: A Dual-Stage Robust Vehicle Detection and Tracking for Real-time Traffic Monitoring. In: IEEE International Conference on Intelligent Transportation Systems Conference (2006)