Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 168 (2016) 47 – 50

Procedia Engineering

www.elsevier.com/locate/procedia

30th Eurosensors Conference, EUROSENSORS 2016

Visual Traffic Load Sensor for Emission Estimation

Kristóf Csorba[a,**], Lilla Barancsuk[a], László Blázovics[a]

[a]*Budapest University of Technology and Economics, Department of Automation and Applied Informatics, Magyar Tudósok krt. 2/Q, 1117 Budapest, Hungary*

Abstract

Traffic enumeration is a frequently used approach in smart city applications for estimating the traffic load of the road network. Many solutions use a camera and state-of-the-art image processing technologies. As image processing is relative resource consuming, either very simple methods are used, or the processing is performed off-site, on a central computer. The approach presented in this paper employs a powerful embedded system to process the video images in real-time, so that the communication traffic is limited to the transmission of retrieved statistics. In order to fit into the limited resources and still allow vehicle size and speed measurements, special data representations and processing pipeline has to be used. This paper presents the operation principles of the sensor, followed by the corresponding evaluation results.

Peer-review under responsibility of the organizing committee of the 30th Eurosensors Conference

Keywords: smart city, traffic counting, embedded image processing

Environment monitoring is a necessary tool for the preservation of healthy urban environments. Many smart city solutions provide air quality, humidity and temperature measurements in distributed sensor networks. This paper presents a traffic monitoring solution which is integrated into a similar network to monitor one of the most significant emission sources of urban environments. Beside conventional reporting services, it also allows for further smart services built over them like route planning for bikes avoiding congestions or areas with low air quality due to high traffic. Furthermore, real-time traffic monitoring allows adaptive traffic light and street lighting control reducing expected travel time and energy consumption.

* Corresponding author. Tel.: +36-1-463-3702; fax: +36-1-463-2871.
E-mail address: kristof.csorba@aut.bme.hu

Peer-review under responsibility of the organizing committee of the 30th Eurosensors Conference
doi:10.1016/j.proeng.2016.11.139

Our solution is part of a system designed for city-wide operation and with several monitoring capabilities. The purpose of the traffic sensor is to estimate the number and type of vehicles using a simple video camera, and report the counting results periodically. Due to communication traffic limitations, video images cannot be transferred to servers in real-time, so they have to be processed on-site, inside the sensor.

Vehicle counting has many common approaches like induction loops in the road, LIDAR or RADAR fences, and video camera based solutions. In cases when the counting has to be extended with vehicle size and speed measurement, video images have significant advantages compared to the other types of sensors. A disadvantage is the processing resource need, as real-time object tracking in video images is hard to be done in low-cost embedded systems and requires significant compromises between speed, accuracy, robustness and allowed features.

Vehicle motion tracking in video images [1,2,3] has several main approaches: background subtraction, feature point or region tracking, and tripwire based solutions. Background subtraction [4] aims for separating the moving image parts from the static background. Gaussian Mixture Models are frequently used to build a slowly adapting color model of the background to detect movements. These approaches work even in the presence of shadows. Difficulties arise in the presence of occlusions due to heavy traffic, and with stopping vehicles which may be lost by motion detection. Feature point and region tracking [5] methods do not have this drawback, but matching the features extracted from different video frames may add significant noise and ambiguities into the process. Tripwire based approaches limit the observation to a line and count every object passing this line. This requires less image data to be processed, but needs further extensions to be able to measure the size and speed of the vehicles, as short and slow vehicles may be hard to distinguish from fast and long ones, as they may spend the same amount of time under the tripwire. The solution presented in this paper uses a hybrid method: the basis of the system is a tripwire combined with background subtraction, extended with point tracking for reliable speed measurement.

1. Software architecture

The traffic sensor has to handle several complex situations like occlusions, lighting changes, and stopping vehicles as mentioned before. Different situations influenced by topography, traffic characteristics like the presence of traffic lights, and camera placement may require different processing. Furthermore, the development itself may need a flexible software environment with interchangeable components. For these reasons, the architecture of the sensor is a plug-in system using a blackboard data model: several processing steps (called processors) follow each other in a configurable sequence allowing for interchangeability and providing the flexibility for experimentations. They work on a common set of images and image sequences called the media storage. On one hand, the media storage contains separate image sequences like the original video images and one showing only moving objects. For example, the background subtraction processor reads the original images and writes binary images for every video frame marking only the moving areas. On the other hand, the media storage has another type of images called timeline images described next.

1.1. The timeline image concept

Beside the conventional video frames and other images derived from them, our solution takes advantage of timeline images (Fig. 1). These are images containing the pixels of the tripwire as columns, for every video frame after each other. The rows of the timeline image correspond to locations on the tripwire, while columns of the timeline image correspond to the frames of the video. The images are called timeline images as the horizontal axis corresponds to the time.

Beside the option to create a timeline image from any frame sequence, like the one of the original video in the case shown in Fig. 1, one also has the option to transform the timeline images directly into other timeline images. This leads to an architecture utilizing several processor plug-ins reading arbitrary frame sequences and timeline images and creating new ones. Three timeline image examples are shown in Fig. 2.