Use of TensorFlow and OpenCV to detect vehicles

Angel Ciprian Cormoș
Telematics and Electronics for Transport
Politehnica University of Bucharest
Bucharest, Romania
angel.cormos@upb.ro

Răzvan Andrei Gheorghiu
Telematics and Electronics for Transport
Politehnica University of Bucharest
Bucharest, Romania
andrei.gheorghiu@upb.ro

Valentin Alexandru STAN
Department of Telematics and Electronics for Transports
POLITEHNICA University of Bucharest
Bucharest, Romania
valentin.stan@upb.ro

Ion Spirea Dănăilă
Telematics and Electronics for Transport
Politehnica University of Bucharest
Bucharest, Romania
danaila.ionspirea@gmail.com

***Abstract*—Streets are becoming overcrowded, especially in larger cities. Traffic monitoring and control is a solution to mitigate this problem, but is essential to have actuated information about vehicles. In this project the aim is to build a traffic monitoring system that detect the movement of vehicles, to observe and to count the different categories. The real-time processing (15-30 fps) of video streams works mainly in daylight. The system consists of three subsystems: image processing, motion detector, control and display. To maximize the detection speed, each subsystem can be processed on a different thread. Image processing is partially performed via OpenCV with a data set previously trained with TensorFlow. The monitoring area will be marked with a polygon. It is possible to filter sidewalks and the opposite traffic direction, so the system will avoid processing in those areas.**

Keywords—Computer vision, artificial intelligence, traffic monitoring, vehicle detection

I. Introduction

In order to optimize the actions that specifically target the surveillance area, it is necessary to extend the surveillance systems, both on public roads and on the entrances from the highway and the belt. Closed-circuit video systems play a very important role, providing information and up-to-date data to public departments on vehicle characteristics, toll payment, operating permits, etc. The safety of people and property is a growing concern for society. With the flexibility offered in relation to new technologies, video surveillance is one of the increasingly requested security solutions in recent years. Closed-circuit surveillance systems' implementation has grown in recent years, but these solutions are frequently the subject of debate. On one hand, there are many benefits, such as their use in criminal investigations, providing a sense of security to the public, or, in case of traffic systems, possibility to extract many features of the vehicles (including shape, color, type, license plate number, and many more). On the other hand, many believe that intrusiveness has far outweighed the benefits. The design and proper use of such systems is essential to ensure a surveillance system that meets the needs of the user, provides a tangible benefit and ensures the safety and security of the general public.

II. Computer vision and OpenCV

Computer vision is the transformation of data taken from a video camera or recording into a decision or a new representation. All these transformations are made in order to achieve a certain goal. A new representation could mean turning the colour image into a grayscale picture or eliminating camera movement from a sequence of images. In order to be able to extract the desired information from an image, multiple transformations may be required, to eliminate the noise and emphasize the desired components.

For example, from a vehicle image, the computer receives a matrix of numbers from the real-time video source, or from the locally stored video source. Initially, there is no built-in pattern recognition, no automatic focus control, no association with some kind of information based on years of experience (that is true for humans). The machine has to implement all the recognition stages, step by step. In Fig. 1 we can see the side mirror on the driver's side. What the computer "sees" is just a matrix of numbers.

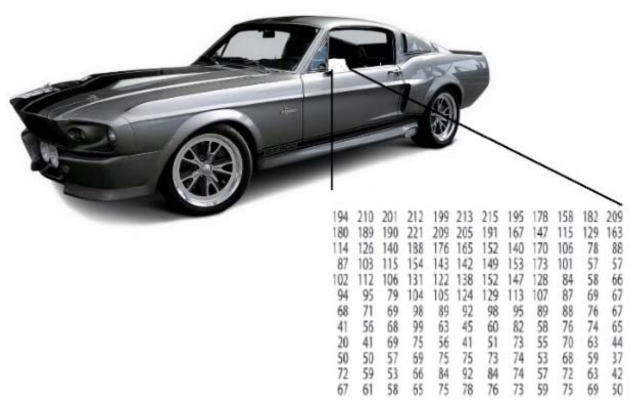

Fig. 1. For a computer, the side mirror represents only an array of numbers 0

Authorized licensed use limited to: Auckland University of Technology. Downloaded on October 25,2020 at 19:55:45 UTC from IEEE Xplore. Restrictions apply.

A. Different perspectives of a video taken from several cameras

This problem is really a difficult one: given the 2D image of a 3D world, a 3D signal must be built. The same image could represent any of the infinite combinations of 3D scenes, even if the data provided would be perfect (see Figure 2). however, the data is corrupted by noise and distortion. Such events may come from different sources:

- variations in the environment (weather, reflections, movements),
- lens imperfections
- mechanical configuration,
- final integration time on the sensor,
- electrical noise in the sensor or other electronics,
- compression after image capture.

In designing a practical system, additional contextual knowledge can often be used to address the limitations imposed by visual sensors.

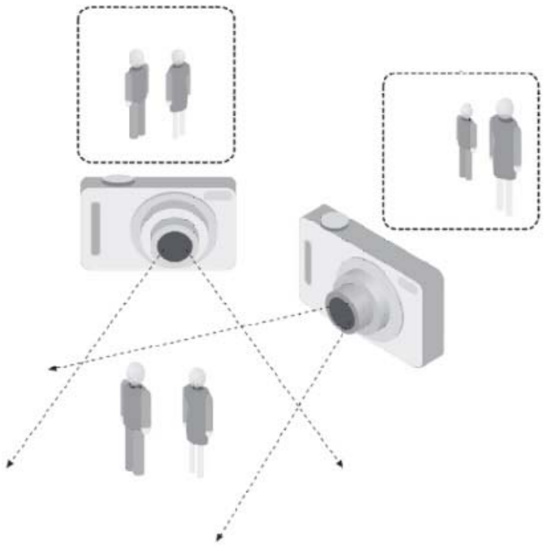
Fig. 2. Different pespectives of cameras 0

Consider the example of a mobile robot that has the task to find and pick up all the pencils in an office building. The robot could use the fact that a desk is an object found inside buildings and that pencils are mostly found on desks. It provides a default size reference that may be used for all the future considerations: pencils must be able to fit on desks. It also helps to incorrectly eliminate their "recognition" in impossible places (for example, on the ceiling or a window). The robot can certainly ignore a 50-meter advertising pole, that looks very similar to a pencil, because it does not meet the requirements, ie it does not have the required background, that is the desk.

B. Detection, control, and monitorig

The system should be capable real-time operations, while also process locally stored data. The traffic analysis software uses images as an input source, analyze them and displays different results on the respective frames: objects that have been detected, or statistical data (e.g. number of frames per second). The system can be divided into several components, as follows:

• Control and display

• Image

• Motion detector.

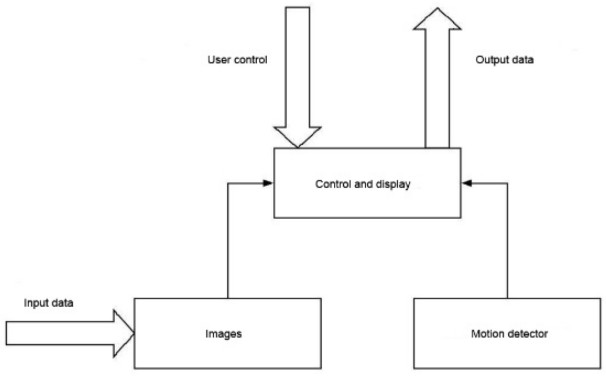

Fig. 3. Subsystems of cameras 0

Control and display

The user is able to interact with the system through the Control and Display subsystem. Here, he/she can open the desired video file, or view the statistical data as well as the results. This subsystem controls the functionality of the entire architecture and interacts the other subsystems.

Image

When starting the Image subsystem, one must first set the video stream source (camera – for real time processing - or video file) and then start playback. Two images will be considered: the original one, and a reduced size that can be used to detect motion.

Motion detector

In this component, the user can first mark the area of interest on the input video stream using matrix vectors (figures 4, and 5). This subsystem is also responsible for detecting motion and objects.

During this stage of detection, there are usually many motion vectors that are not important for the result, so there is no need to further process them, as this will slow down the software. These motion vectors can be detected in several areas of the image: movements in the opposite lanes, sidewalks, rivers, trams, etc. To avoid excessive image

Authorized licensed use limited to: Auckland University of Technology. Downloaded on October 25,2020 at 19:55:45 UTC from IEEE Xplore. Restrictions apply.

processing and overload of the algorithm, relevant areas of the image can be marked using delimitations. Therefore, the program will process only the area chosen. Setting the matrix can be done in a simple and easy to use way: just mark the vertices of the area and the algorithm will create a polygon that matches these vertices.

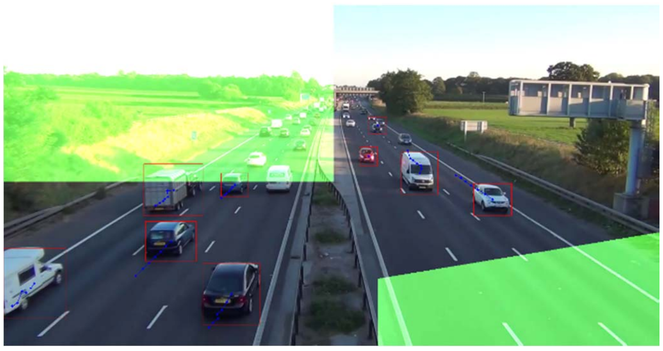
Fig. 4. Detection area for vehicle counting 0

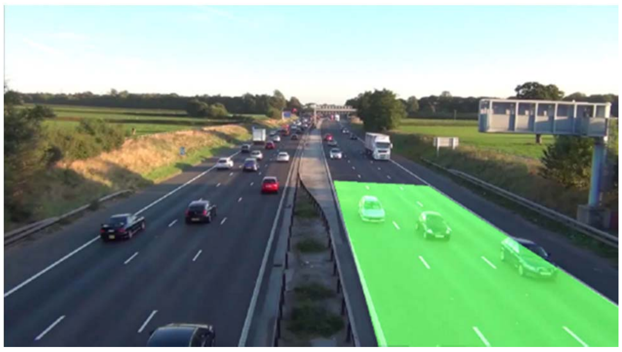
Fig. 5. Detection area used to determine occupancy 0

Background identification is an essential step in preprocessing in many computer vision-based applications. For example, consider scenarios, such as visitor counter for people entering or leaving a room, a traffic device that extracts vehicle information, and so on. In these cases, we must first extract people or vehicles from the noise that is unnecessary information. From a technical point of view, we need to extract the moving background from the static background. If we only have the background image, such as the image of an empty room, or the image of an empty, this task would have been simpler. But in most cases, such an image is not available, and is necessary to identify and extract the background from the images that are available. It becomes more complicated when the objects that we are trying to identify (people or vehicles) have shades. Since the shadow also moves, it will be counted. For this, a more complex algorithm is required [4]. In this project, MOG (Mixtures of Gaussians) was considered [5]. This evaluates each background pixel by using a K Gaussian distribution (K = 3..5).

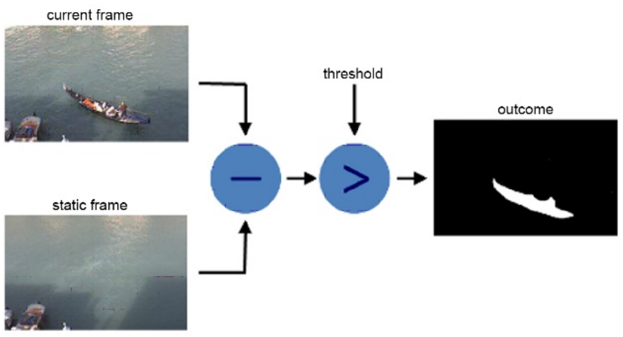

Fig. 6. Solution to exclude static frames 0

In some cases, we cannot set the static frame because the light can change, or the objects can be moved by someone, or the image is in a continuous motion. In such cases we must realize the pixels that are static in the largest part of the time and that becomes the background layer.

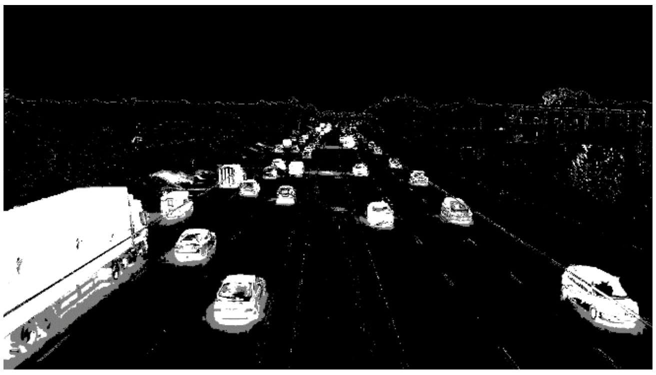
Fig. 7. Frame without static background 0

C. *Connection between OpenCV and TensorFlow*

Overall, OpenCV and TensorFlow are two libraries dealing with different aspects. OpenCV is used for computer vision that includes image classifiers, while TensorFlow is used to build models used to drive networks. While OpenCV is a collection of algorithms (for example, through a line of code, many actions can be performed), TensorFlow is a framework composed of small blocks, including some examples (eg classifiers). In many projects aimed at image recognition and object detection, both solutions can be used. For example, OpenCV can detect vehicles in images or videos and TensorFlow will determine the license plate or identify a specific type of vehicle.

Authorized licensed use limited to: Auckland University of Technology. Downloaded on October 25,2020 at 19:55:45 UTC from IEEE Xplore. Restrictions apply.

TABLE I. OPENCV VS. TENSORFLOW

OpenCV		TensorFlow	
Advantages	***Disadvantages***	***Advantages***	***Disadvantages***
Image prcessing	Memory management	Flexible and customizable	Model training requires a lot of time
30+ frames per second	Rather difficult to use	Very high accuracy	Not very well optimized
High accuracy	Does not have its own editor	Efficiency	Requires many resources
Supports many languages		Scalability	Hard to interpret
Well optimized		Debugging support	Accessible only for Nvidia hardware
Runs on all possible units		Chart generation	Does not provide support for Windows platform

III. TESTS PERFORMED

The tests in this project were performed on a Lenovo B50-80 computer with 2 cores, 4 threads, 16 GB RAM, Ubuntu operating system, an OpenCV 4.0.0. As processing algorithms, were used:

- Caffee 1.0.0 with TensorFlow 1.12.0
- Keras 2.2.4 with TensorFlow 1.12.0
- PyTorch 1.0.0.

For image classification models have been trained with YOLO network [6]. A comparison between different technologies is presented in Fig. 8.

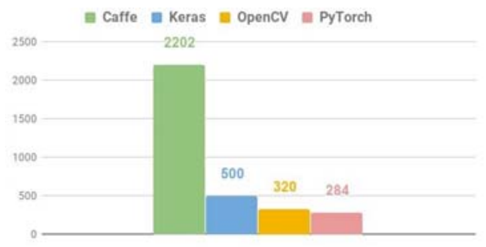

Fig. 8. Image clasification – execution time comparison [9]

Obstacle detection was performed using YOLO v3 on Darknet [7]. The comparison between OpenCV and DarkNet is presented in Fig. 9.

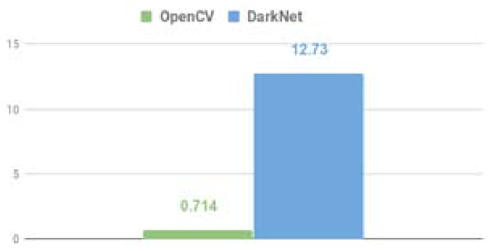

Fig. 9. Obstacle detection – execution time comparison [9]

For object following GOTURN (Generic Object Tracking Using Regression Networks) algorithm was used [8][9]. In this case, OpenCV was compared to Caffe architecture that was used by GOTURN developers, and the results are presented in Fig. 10.

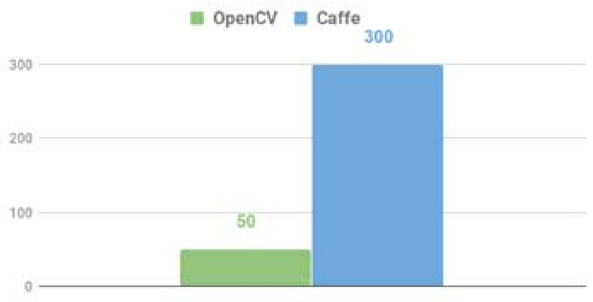

Fig. 10. Object following – execution time comparison [9]

From all the tests performed, it resulted that OpenCV is the optimal solution for image analysis.

IV. CONCLUSION

Lately, learning algorithms as well as artificial intelligence have grown. More and more applications and automation processes are using these progressive learning algorithms, the performance increasing exponentially. The method presented in this paper is designed to handle two situations: static background and moving background. For the first scenario, we can use a detector that will identify minor variations in the real scene by updating the model. In the second case, the background movement analysis, image recording is used to estimate the movement of the camera, thus allowing the detection of a vehicle in a frame of reference.

REFERENCES

[1] Gary Bradski and Adrian Kaehler,„Learning OpenCV", O'Reilly Media, Inc, 2008

[2] Attila József Kun, Zoltán Vámossy „Traffic Monitoring with Computer Vision", 2015

[3] Andrey Nikishaev, Making Road Traffic Counting App based on Computer Vision and OpenCV, 2017

[4] https://docs.opencv.org/3.3.0/db/d5c/tutorial_py_bg_subtraction.html

[5] P. KadewTraKuPong and R. Bowden, An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection, In Proc. 2nd European Workshop on Advanced Video Based Surveillance Systems, AVBS01. Sept 2001. VIDEO BASED SURVEILLANCE SYSTEMS: Computer Vision and Distributed Processing, Kluwer Academic Publishers

[6] Sania Bhatti, Liaquat A. Thebo, Mir Muhammad B. Talpur and Mohsin A. Memon, *A Video based Vehicle Detection, Counting and Classification System*, I.J. Image, Graphics and Signal Processing, 2018, 9, 34-41, Published Online September 2018 in MECS (http://www.mecs-press.org/) DOI: 10.5815/ijigsp.2018.09.05

[7] Hirokatsu Kataoka, Teppei Suzuki, Shoko Oikawa s.a., *Drive Video Analysis for the Detection of Traffic Near-Miss Incidents*, 2018 IEEE International Conference on Robotics and Automation (ICRA), 21-25 May 2018, Brisbane, QLD, Australia

[8] Wei Yang; Ji Zhang; Hongyuan Wang; Zhongbao Zhang, *A vehicle real-time detection algorithm based on YOLOv2 framework*, Proceedings Volume 10670, Real-Time Image and Video Processing 2018; 106700N (2018) https://doi.org/10.1117/12.2309844 Event: SPIE Commercial + Scientific Sensing and Imaging, 2018, Orlando, Florida, United States

[9] Satya Mallick, „CPU Performance Comparison of OpenCV and other Deep Learning frameworks", 2019

Authorized licensed use limited to: Auckland University of Technology. Downloaded on October 25,2020 at 19:55:45 UTC from IEEE Xplore. Restrictions apply.