National Research University Higher School of Economics
Faculty of Business Informatics
School of Software Engineering
Software Management Department

AN APPLICATION FOR DYNAMIC OBJECT IDENTIFICATION BASED ON LUCAS-KANADE ALGORITHM

Student:	Kostenko Dmitry
Group:	472SE
Argument Consultant:	Prof. Ivan. M. Gostev, PhD
Style and Language Consultant:	Tatiana A. Stepantsova

Moscow
2013

Abstract

[1]

Contents

1 Introduction

Nowadays a significant growth of the vehicles amount can be observed in all Russian cities. According do the Russian highway patrol data, this growth is approximately a hundred thousand new cars per year. Taking into account the fact that almost no new highways appear, we can easily come to a conclusion that the traffic conjunctions problem is a direct result of this annual growth. Another problem that arises as a side effect of the jammed highways is an increase in fuel usage. A good solution of these problems can be installing an Intelligent Transportation System (ITS). These systems range from rather simple implementations which are able to control traffic lights to sophisticated solutions that are capable of registering the vehicle flow velocity and predicting violations. These are the tasks, which are usually assigned to these systems:

1. ensuring maximum traffic capacity;
2. reducing the amount of accidents on the highway and monitoring the human factor;
3. collecting the information about the traffic jams from the vehicle flow and informing its participants;
4. environment protection as an indirect result of real-time monitoring of situation on the roads.

ITS can contain various sensors from the ones that are capable of detecting heat to the ultrasonic ones. Manual processing of a significant amount of data which is received from these sensors is not applicable in the real-life situations. As a consequence, a vital necessity of automation of the process and decision making, based the information gathered during this process, arises. Automatic vehicle detection in this video monitoring is a complex objective of computer vision. On of the tasks of such a system is to count vehicles on the highway. In its turn, it is divided into subtasks of computer vision, such as foreground retrieving (vehicles) and tracking in the next frames. This paper offers an overview of an approach which is related to automatic tracking of moving vehicles. The only source of data used by this approach is a camera video recording. The first chapter of the present graduation paper offers an overview of existing solution in the field of video monitoring. Problem statement and requirements to the algorithm under development are covered in the second chapter. The third chapter deals with description of methods and algorithms. And finally, the brief exploration of expected results will be introduced.

2 Related work

In this section a short review of existing highway video surveillance solutions is presented. There is only one full ITS architecture, developed by the transportation department of the US. It focuses on creating a unified information environment, that connects automobiles, road equipment, dispatch centres, and data centres around the country. This system has been patented by the US government.

There are several analogues around the world that are based on the US system. In 2012 a Russian ITS was rolled out in Moscow. All of these systems are commercial and there are no ways of analysing there methods. However, there are also several open-source projects of smaller scale. As has been mentioned before, one of the main tasks of such systems is counting vehicles on the highway.

Most of the times the vehicle counting algorithm consists of the following steps:

1. create the background model;
2. subtract this model from the current image and obtain all moving objects of the foreground (vehicles);
3. count the amount of vehicles that has crossed the line of interest;
4. update the model of the background.

Examples of such an approach: ****

Our work uses a different approach. We suggest an alternative method for counting vehicles without the pre-modelled background. It is based on the differences sum of the neighbouring frames of a video sequence and highlighting the connected areas.

3 Problem statement

In this chapter we will state the set of requirements for the algorithm. As a formality we will state several definitions first. As any video is a sequence of frames, every moment in time we have one frame called the current frame. If the current frame is defined as i, the previous frame will be defined as (i-1). We will consider all the moving objects such as vehicles, trees, humans as the foreground. Whether the object is considered as a moving one depends on the camera properties and the environment. So we define moving objects as the ones that shift their position in the current frame, relatively to previous frame. When the camera moves, or the object in its focus, it leads to the image changes. The description of the movements which can be seen is called the optical flow. There are several methods for calculating this flow. Lucas and Kanade developed one of them, using differential approach [1]. It will be described later in this paper. Now let us move to the problem statement. One of the most important problems of video surveillance is identifying the object of interest and following its trajectory through the sequence of the following frames. This problem can be partitioned into several sub-problems. In the beginning it is necessary to find the objects of interest, which can also be called the foreground objects. Finding these objects involves separation of the foreground from the background of the image. As a result we get a binary map of the moving objects. After that we need to separate vehicles from other moving objects such as trees or humans. For doing so we limit the area in which we will track movement. A good choice will be a well-observed part of a highway. Then we need to track the changes in objects of interest coordinates in the following frames.

To calculate the optical flow we need to make several assumptions:

1. An image is a continuous function of two variables. This will let us use mathematical analysis, and also carry out mathematical operations with the image;

2. The brightness of an object remains constant in a short period of time. Without this assumption we will not be able to track the object;

3. The location of the object of interest will not change a lot relatively to the previous frame. This is due to the fact that in real life objects do not move with the speed of light.

Let us state the problem: counting the amount of vehicles which appear in a certain area of interest.

We can also formulate the following algorithm requirements: 1) working without any specific data about the highway, or the images that are going to be processed; 2) real-time processing of the data flow; 3) not requiring a significant amount of computational power. The minimal hardware requirements will be proposed in the specification. At the time of writing this paper these requirements are impossible to describe.

Now let us move to the descripton of the algorithm.

4 Algorithm

In the following chapter an approach to the task of vehicles on motor ways detection and counting will be revealed.

As a video is a frame sequence, at every moment only one frame is available. In the beginning, it is necessary to eliminate a noise of each frame. Gauss filter must be applied in order to perform this. The information about object color seems to be not in use in further. Therefore it will be appropriate to transform the image from RGB color space to grayscale. Also, for the sake of increasing the processing performance it is necessary to reduce the size of the image. But it is possible only if the camera allows to do so. For instance, if the camera shoots with a resolution of more than 640 by 480 pixels, every frame of the video stream can be reduced to this resolution to increase the processing performance of every frame. After all of the images pre-processing procedures, the foreground objects are detected.

4.1 Allocation of foreground objects

The primary goal of the method, described in this section, is retrieving the foreground objects location. The traditional algorithms of retrieving the foreground objects location are based on pixel-by-pixel frame difference method. But this method also has several disadvantages. This method's application might lead to the enormous amount of isolated areas (Fig. **??**).

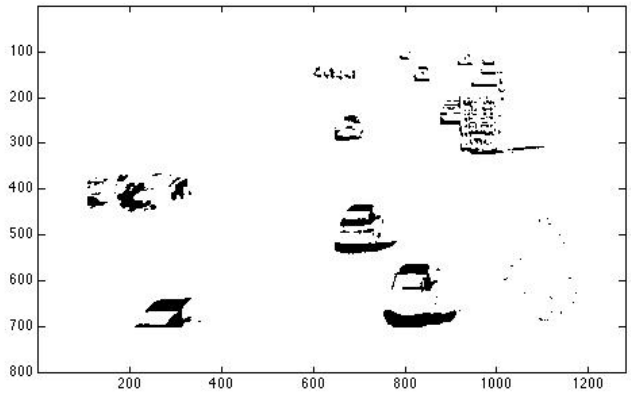


Fig. 1 *Pixel-by-pixel frame difference method.*

For this reason, this method must be modified to avoid this disadvantage and to make these areas connected with each other. At first, each frame has to be divided into blocks which do not overlap. The optimal size of a block is found empirically and depends on characteristics of the camera and the environment. That is why in the beginning the standard size will be assigned to each block - 3x3 pixels. In Fig. 2.

After that the previous frame is subtracted from the current frame. Then every pixel block of the result is filtered in the following way: if the amount of foreground objects pixels

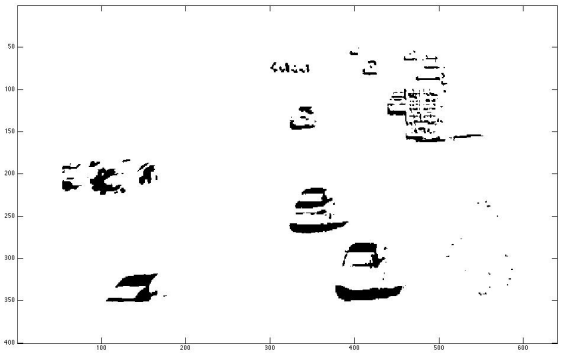


Fig. 2 *The result.*

in every block is more than some pre-defined threshold, this block is considered to be the foreground otherwise, it is considered to be the background

As a result we get the binary image of the foreground objects. But this image is not going to be perfect, it will definitely have noise. For that reason the image must be processed using morphological operations. These operations are the following: erosion and dilation.

It is necessary to consider that the method of subtracting the previous frame from the current frame gives good response near the moving object border. However, inside the objects borders the response is NOT SO GOOD. If the vehicle is relatively large, the probability of its recognition as an object of the foreground is low.

To get rid of this effect let us consider two definitions: sort-term model of the foreground and long-term model of the foreground.

The short-term model is the model which we get, after applying the steps stated above. The long-term model is the pixel-by-pixel sum of N previous short-term foreground models, where N is a positive integer and more than 1 (Fig.3). N is found empirically and depends on characteristics of the camera and the environment.

a) b) c) d) e)

Fig. 3 *(a) - first frame; (b) - secind frame; (c) - third frame; (d) - fourth frame; (e) - sum of all frames.*

As the objects that we want to track are moving, calculating the long-term model of the foreground we get the object regions, which is more connected. As can be seen from Fig.4 the object region can be separated into several regions. To reduce the amount of these regions we use the dilation operation with pre-defined window.

Fig. 4 *Object, separated into several regions.*

4.2 Selection of connected regions

After creating the binary map of the foreground objects, it is necessary to give every object a certain number. A straightforward segmentation algorithm is used. Suppose, we obtained the binary image. In our case white pixels stand for the background, and black pixels stand for the potential vehicles. The mapping from the original image to the segmented image is mono-semantic. We will call an array of pixels connected if and only if each pixel has at least one neighbouring pixel belonging to this array. There are two ways for pixels to be connected:

1. 4 connected (Fig. 5 (a)) - the neighbours of the pixel are only the following pixels: the one above, the one below, the one to the right and the one to the left of it;

2. 8 connected (Fig. 5 (b)) - the neighbours of the pixel are the pixels above, below, to the right, to the left, and the diagonal pixels.

	1	
2	*	3
	4	

a)

1	2	3
4	*	5
6	7	8

b)

Fig. 5 *(a) 4-connected, (b) 8-connected*

In our approach the 8 connected way is used, because it gives a significant raise in precision. There are generally known two methods of image marking: recursive and iterative. The shortcomings of the recursive approach are the following:

1. a great amount of memory is needed for it;

2. it is rather slow, compared to the iterative method.

The iterative method is an alternative to the recursive approach. It can be often met in literature by the name of: "Sequential scanning algorithm". Let us present it in detailed steps:

1. To be specific, we start processing the pixels of the image from the left-top corner, going from the top to the bottom and from the left to the right.

2. While cycling through the pixels of the image we ignore the background pixels, and mark the current pixel with the color of the top-left neighbouring pixel.

The idea behind this algorithm is to use a special mask, shaped like a corner, called the ABC mask. This mask can be seen in the Fig. 6. Going through the image with this mask is carried out in a top to bottom and left to right fashion. It is assumed that there are no objects behind the image border, so it is possible for a B or a C to end up behind this border. This requires an extra verification after the scanning. Figure 7 shows five possible positions for the ABC mask on the image.

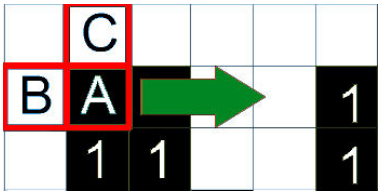


Fig. 6 *ABC-mask.*

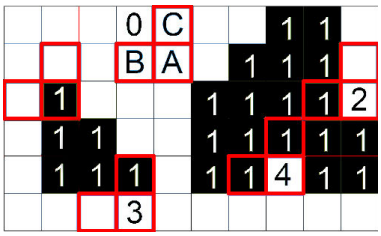


Fig. 7 *5 cases of ABC-mask.*

1. Position 0, where all three components of the mask are not marked — in this case we simply skip the pixel.

2. Position 1, where only the A element is marked — in this case we have found a new object which gets its own unique number.

3. Position 2, where only the B element is marked — in this case we mark the current A pixel with a number from B.

4. Position 3, where only the C element is marked — in this case we mark the current A pixel with a number from C.

5. Position 4, here the B and C marks are connected and equivalent and the A pixel can be marked with the number either from C or B. In some implementations the equivalence graph of such marks is created, with some explanations. We do not consider it necessary. We are going to do the following - in the case, where B is equal to C, we go back and mark all processed pixels marked as C with B. This process will be described in detail later in this paper.

After processing the image, we need to go through the array of pixels again and do the marking procedure with the consideration of areas equality, As a result, an image is obtained, where each connected area is marked with its own number.

========== ==========

Now we cut out the areas which can not be classified as vehicles. To do so, we need to count the amount of pixels in each area. The areas in which the amount of pixels exceeds a certain threshold are eliminated. This threshold is found empirically and will depend on the camera characteristics and the environment conditions. At the time of writing we can not tell exactly what the threshold is going to be. That is why, for now we will just ignore it and consider all of the connected areas to be the vehicles that we are looking for.

4.3 Vehicle tracking and counting

After obtaining the set of connected areas in the current frame (which are vehicles that we are looking for), we need to determine which of these objects are new, and which of them has already existed in the previous frames. We will consider the position of the connected area as the coordinates of its center. These coordinates are calculated as the average value of all the X and Y coordinates of the pixels, that this area contains. Thus, the problem can be stated in the following way: we need to match the set of connected areas that we have on the current frame with the sets of areas we had on the previous frames, and initialize new areas as they appear. Earlier we made several key assumptions. One of them was that the vehicle in the current frame has only slightly moved in respect to itself in the previous frame. So, if we store the positions of the connected areas centres we can find the closest center of an area in the current frame to a center on the previous frame, using the method suggested by Lucas and Kanade [1]. The main idea of this method is to find the closest point to the given point by giving every point its weight. A more detailed description of this algorithm can be found here: [1].

Let us sum up our approach in an overall algorithm scheme:

1. Translate the image from RGB to gray scale;
2. Process the gray scale image with the Gaussian filter (this is done to eliminate noise);
3. Subtract the previous image from the current one;
4. Highlight the connected areas which are most likely to be the vehicles;
5. Compare the current connected areas with the same areas in the previous frames with the use of Lucas Kanade approach [1], initializing the new areas as they appear;
6. If such an area intersects an area that we are surveilling, add 1 to the counter.

5 Conclusion

In this paper we proposed another method for counting the amount of vehicles on a highway. The key propositions we made are the following:

1. the background model does not necessarily need to be stored at all times;
2. the Lucas - Kanade method [1] suits the task of vehicle counting really well.

Apart from numerous positive aspects, our method has its downsides. First, we assume that the camera is stable and does not move. Although, in real-life situations this will hardly be possible. The camera might be moved by the wind for instance. Another problem can be the rain. Raindrops might be recognized as the objects of the foreground. But the solutions of these problems is out of the scope of our research, they should be solved by the hardware, not the software.

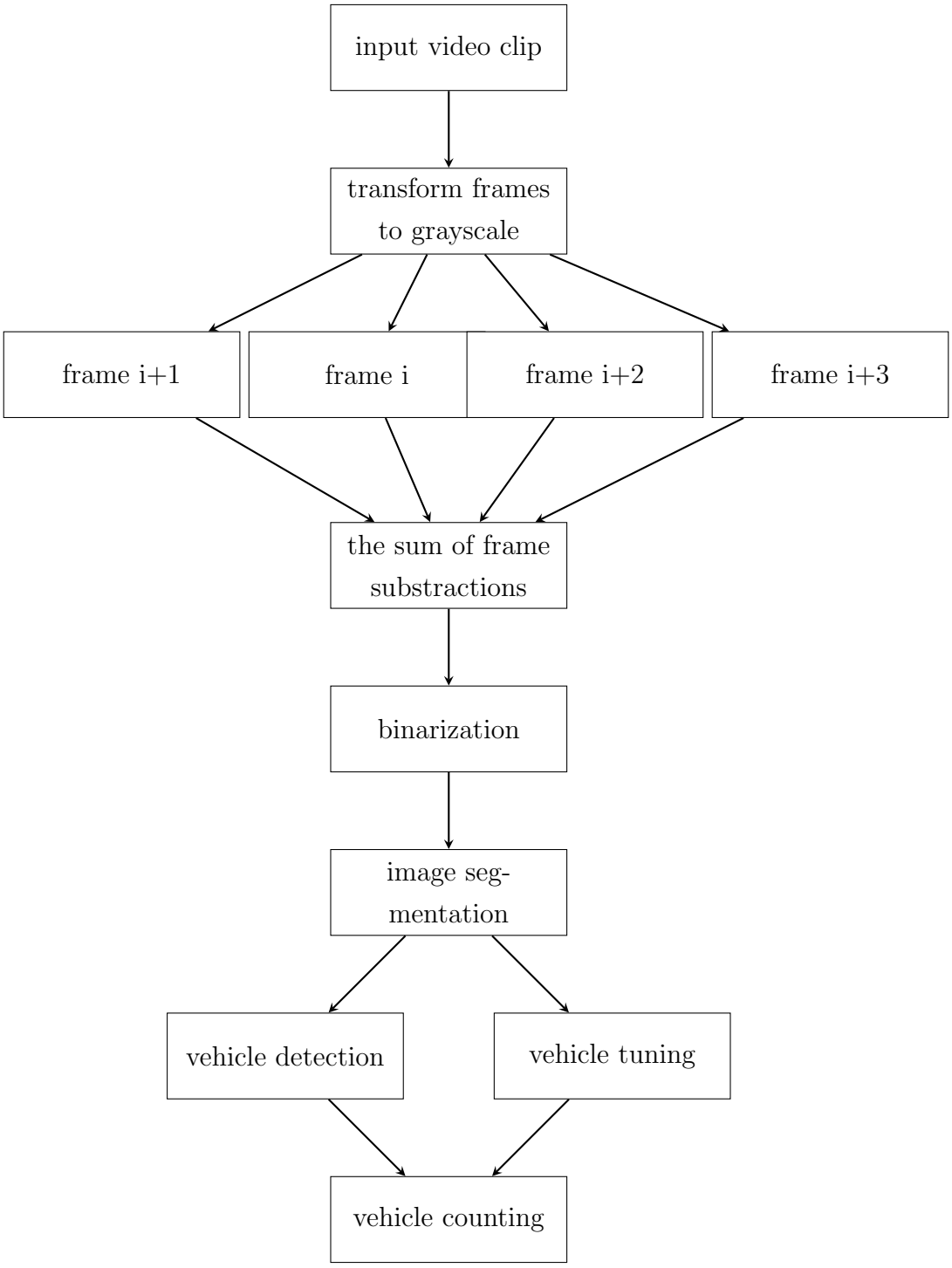
6 Bibliography

References

[1] An Iterative Image Registration Technique with an Application to Stereo Vision, Bruce D. Lucas Takeo Kanade, Computer Science Department Carnegie-Mellon University Pittsburgh, Pennsylvania 15213

[2] Real Time Vehicle Detection and Counting Method for Unsupervised Traffic Video on Highways, Mrs. P.M.Daigavane † and Dr. P.R.Bajaj ††, S. D. College of Engineering, M.S., INDIA

7 Appendix


input video clip
transform frames to grayscale
frame i+1
frame i
frame i+2
frame i+3
the sum of frame substractions
binarization
image segmentation
vehicle detection
vehicle tuning
vehicle counting