Foreground Segmentation

Iuliia Alekseenko and Ricardo Luque

I. INTRODUCTION

The two main objectives of this laboratory session are developing a basic foreground segmentation algorithm to detect moving objects on a stationary background, as well as, improving the performance of a basic algorithm. All tasks are implemented using C++ and Open CV, while evaluating the performance of the algorithms is carried out using standard measures and a Matlab script code that compares the performance of the implementation in context with a ground truth. The evaluation of the algorithms showed that each segmentation method seems to be more robust in each specific task.

II. METHOD

There are five main methods implemented during this laboratory session. First, the foreground segmentation mask based on frame difference. Second, a progressive update of the background model on both, blind and selective, modes. Third, a suppression of stationary objects known as ghosts. Fourth, a shadow detection based on HSV color space information about a frame. Fifth and last, a more complex background segmentation model based o a single Gaussian distribution. The mentioned methods are described below.

A. Foreground Segmentation Mask

One of the simplest methods to distinguish foreground from background is performing a subtraction between the current frame and a background model. This can be achieved when all or most of the background has very little to no change, and bearing a static camera in use. To achieve this, a background mask is used, so that each frame is compared to it and obtain the respective foreground in context. Based on the assumption that the first frame of the video is static, we consider it the background.

This frame is used as to initialize the mask and later, when obtained the foreground through subtraction, it is put through a threshold to achieve a better result. The threshold value is set empirically to 50 for all sequences, and adapted to 20 for evaluation. It is important to mention there is a correlation between the threshold and improvement of each frame; thus, frames containing more rapid movements may require a higher threshold value.

B. Progressive Update of the Background Model (Blind and Selective Modes)

The intention of this step is updating the background model with the current image excluding the pixels which belong to the foreground mask. This process can be either blind, or selective. On the first one, only background pixels are updated; on the latter, this limitation does not exist. In general, this progressive update approach helps solving the problem of ghosts as well as false positives. The main parameter of the algorithm is an alpha value which regards to the adaptation speed.

C. Suppression of Stationary Objects

Due to a counter threshold; which is either incremented if a pixel is part of the foreground or reset if part of the background, the algorithm allows overcoming the problem of objects that appear or are removed and false detected as foreground. This is an ideal approach to deal with ghosts.

D. Shadow Detection based on HSV

The image is processed in a HSV color space since it allows explicitly separating chromaticity and luminosity. Not only so, but it is more efficient for shadow detection. For each pixel belonging to the objects, it checks if it is a shadow according to the the method presented in [1].

E. Single Gaussian and Mixture of Gaussians (MOG)

Single Gaussian implies a Gaussian distribution to each pixel of the frame. Then, the algorithm calculates the mean and standard deviation and updates these values based on the new (incoming) pixel values and foreground mask.

The MOG algorithm, on the other hand, applies a set of Gaussians for every frame pixel and uses an approximation to update the model. Later, the distributions of the adaptive mixture model are evaluated to determine which of them is most likely to result from a background process [2].

III. IMPLEMENTATION

A. New Functions and Variables for Progressive Update of the Background Model

1) double _alpha: is the parameter which specifies the adaptation speed in the update formula:

*_bkg =_alpha*_frame+(1-_alpha)*_bkg;*

2) bool _selective_bkg_update: is used to switch between a blind mode (false) and selective one (true).

3) progressiveUpdate(): - the function which contains two modes of upgrade. In this function, the two extra variables are created - *fgLogicalMask* which is the foreground logical mask and *bgLogicalMask* which is the background logical mask.

B. Suppression of Stationary Objects

1) cv::Mat counter: is a counter matrix which will be incremented or reset depending if a current pixel is background or foreground.

2) int _threshold_ghosts2: is a threshold used in the algorithm specifying when the frame update happens.

C. Shadow Detection based on HSV

1) cv::Mat _framehsv: is a matrix for the HSV version of a frame.

2) cv::Mat _bkghsv: is a matrix for the HSV version of background.

3) Mat hsv_img[3]: is matrix for splitting HSV frame into three channels.

4) Mat hsv_background[3]: is matrix for splitting HSV background into three channels. The following constants are needed for the algorithm implemented - *const double ShadowDetetionAlpha_*, *const double ShadowDetetionBeta_*, *const int ShadowDetection_S_*, *const int ShadowDetection_H_*.

D. Single Gaussian

1) INIT_SIGMA_SQUARED: is an initial sigma value of the algorithm which the user can change.

2) int std: is a standard deviation value of the single Gaussian distribution.

3) double alpha_gaussian: is the temporal window size used to fit the probability density function.

4) void bgs::new_fgmask: is the function which implements the algorithm for the Gaussian distribution for each pixel.

E. RGB version

Instead of converting the frame to gray scale, it is processed as it is. The channels of the frame, background and difference between them are stored in specific arrays created for each channel. The algorithm is the same.

F. Running the code

The process of compiling and running the project is straightforward. First, it is important to choose the proper path for video files, as well as, the output to store results for further evaluation (it should be set in the main function).

Also, the user can choose other values of threshold (*tau, alpha*), threshold for ghost suppression (*threshold_ghost2*) in the main function of the program. It is also available to switch in this function between the type of background update (*bool selective_bkg_update*) and color spaces (*bool rgb*). When working with a single Gaussian model a tuning of *INIT_SIGMA_SQUARED* is required, as well as the standard deviation (*std*) and *alpha_gaussian*, which determines the size of the the window that is used to fit the probability density function (PDF).

For the mixture of Gaussians, on the other hand, we need a variable *K*, for the number of distributions used, *omega* for a weight associated to the ith Gaussian at a specific time. The threshold *T* is used for modelling the background.

IV. DATA

The dataset consists of four video - office room (*empty_office.avi*), hall with students (two available versions - *eps_hotstart.avi* and *eps_shadows.avi*), hall with office workers (*office.avi*), and table (*stationary_objectr.avi*). The frames from every sequence are presented in Figure below.

The first video is *empty_office.avi* which contains the frames of office room and after a certain amount of time, the light in the room is changing. The main problem when processing the video is connected with these daylight changes and variation of background. Thus, the alpha value should be set properly during progressive update of the background model. This is a typical example of a ghosts problem when false foreground pixels may be detected.

The second video shows a hall with walking students (*eps_hotstart.avi and eps_shadows.avi*. The difference between two versions is that the first frame of hot start contains a moving object (persons) which, in ideal case, can not be detected as background. The second version has first frames of background (no moving foreground objects). The main problems to deal with are ghosts (in case of a hot start video, the background model will contain moving foreground objects) and shadows/reflections which can be detected as foreground.

The third video is a office hall (*office.avi*) with two workers crossing it. Due to the clothes similar to background, the main problem when segmenting is camouflage, thus some parts of foreground can not be detected.

The last video in the dataset contains frames of a table (*stationary_objectr.avi*) and after some time a hand is placing new objects on the table and later changes their locations. The main point to take into account is a ghosts problem when background objects will be detected as foreground if they move.

V. RESULTS AND ANALYSIS

After obtaining the results it is important to make an analysis both, in order to comprehend the functioning of the algorithms under different circumstances. In this section the quantitative and qualitative analysis are presented.

A. Qualitative Analysis

This analysis is in charge of describing the quality in a non numerical way in order to analyse the algorithms in context.

1) Suppression: For **Exercise I** we can observe that a correct foreground and background mask were achieved, which can be observed in **Fig. 1** and **Fig. 2**. Due to the static nature of a written report, it is hard to show the ghost suppression; however, it can be observed in the respective *makefile*.

Fig. 1. Suppression I

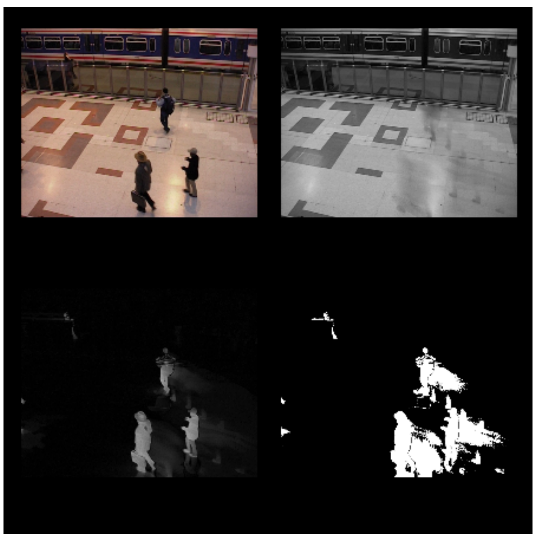

Fig. 2. Suppression II

2) Shadow mask and removal: For **Exercise 3**, we can observe that an appropriate shadow mask was detected, which can be observed in **Fig 3** and **Fig.4**. On the third image, from left to right, on both images, the shadow mask can be observed, which in fact matches the actual shadow mask that should be expected.

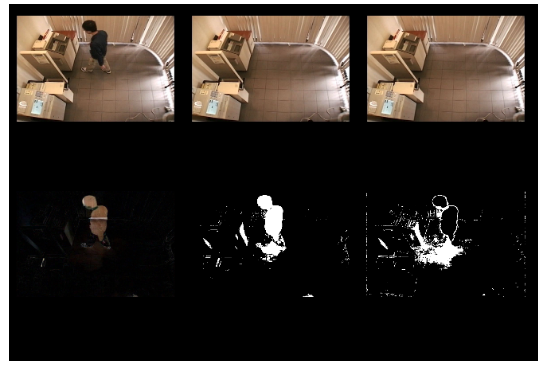

Fig. 3. Shadow Mask I

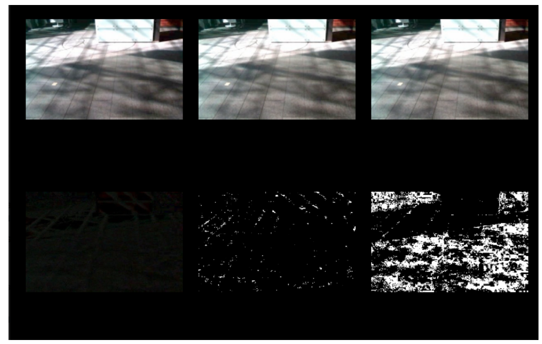

Fig. 4. Shadow Mask II

3) Gaussian: Dynamic For **Exercise 4**, we can observe that an appropriate mask was a dynamic background is detected, which can be observed in **Fig 5** and **Fig.6**. It is important to mention that this model works as an adaptive threshold in which the values change according to a distribution, so that each pixel changes statistically rather than with a fixed threshold value.

B. Quantitative Analysis

We can observe the results for **Exercise 1**, which was in charge of doing three things mainly, in **Table I** and **Table II**. First, the foreground detection based on absolute difference, second the selective update based on the alpha value and finally the ghost suppression.

We can observe that for the first tuning with the values shown in **Table I** vs the second one, shown in **Table**

Fig. 5. Mask for Dynamic background I

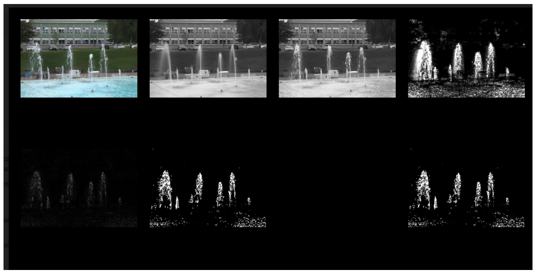

Fig. 6. Mask for Dynamic background II

II, the recall, and overall performance of the algorithm improved drastically. Meaning that in fact, the detection of true positives has increased. However; the specificity, decreased slightly, which means a couple of true negatives were detected as false, meaning some pixels were not detected correctly as the background. Finally, the precision also went down, meaning that there is a decrease in true positives, meaning that less correctly identified foreground pixels were detected.

TABLE I
EXERCISE 1 RESULTS: TUNING 1

Exercise #	**Recall** $\tau = 90$	**Specificity** $\alpha = 0.04$	**Precision** $Select = yes$	**Overall** $GhThr = 25$
1.1	0.4104	0.9980	0.9309	0.8648
1.2	0.6838	0.9956	0.8797	0.7754
1.3	0.3366	0.9978	0.8688	0.9107

TABLE II
EXERCISE 1 RESULTS: TUNING 2

Exercise #	**Recall** $\tau = 20$	**Specificity** $\alpha = 0.04$	**Precision** $Select = no$	**Overall** $GhThr = 10$
1.1	0.8965	0.9326	0.4589	1.3899
1.2	0.6652	0.9742	0.4405	1.0045
1.3	0.6652	0.9742	0.4405	1.0045

Moreover, we can observe the results for **Exercise 3**, which is in charge of removing the shadows by identifying a shadow mask and subtracting it from the initial background separation. In here two different tuning approaches were used with the values shown in **Table III**.

In this table, it was observed that both, the *Recall* and *Overall* performance increased, meaning that the detection of true positives has increased. In other words, more pixels were correctly segmented as foreground and the performance of the shadows elimination was carried out better. For the specificity, on the other hand, it faced a slight decrease after applying a threshold of 20. this means that some pixels corresponding to background were not detected correctly. Finally, the precision faced a nosedive, which means some foreground pixels were not properly detected.

Finally, for **Exercise 4**, which is in charge of using a simple Gaussian model for correct foreground and background segmentation, we can observe the following results shown in **Table IV**.

TABLE III
EXERCISE 3 RESULTS

Tuning #	**Recall** $\tau = 90, 20$	**Specificity** $\alpha = 0.04$	**Precision** $Thrs = 100$	**Overall** $GhThr = 25, 10$
1	0.4390	0.9939	0.7373	0.8828
2	0.8326	0.8984	0.3292	1.8955

Here we can observe that both, the *Recall* and *Overall* performance increased. Which was the same case scenario in the last two exercises. More pixels were correctly segmented as foreground and the performance of the shadows elimination improved. We can also observe most of the variables compared have a similar behavior since the specificity and precision decreased as well. This can be seen in **Table IV**.

TABLE IV
EXERCISE 4: RESULTS ON GRAY IMAGE

Tuning #	**Recall** $\tau = 90, 20$ $GhThr = 25, 10$	**Specificity** $\alpha = 0.04$ $\sigma = 100$	**Precision** $\alpha_g = 0.05$	**Overall** $Thrs = 100$
1	0.1893	0.9950	0.4887	0.5834
2	0.6621	0.8864	0.0740	1.9742

VI. CONCLUSIONS

Several methods for background segmentation were implemented throughout this laboratory session which were of help in understanding segmentation in an object oriented environment.

Each segmentation method seems to be more robust in each specific task. For instance on **Exercise 3**, shadow detection, a correct shadow mask was identified and it helped improve the previous foreground mask. Although this was achieved using the mask, the ghost suppression had a very similar result, since it identified the shadows as ghosts and corrected them rapidly. This means, that even though, the shadow detection algorithm had a principal aim, it could have been solved by using the ghost suppression itself.

Furthermore, the Gaussian model worked according to expectations, since it was able to correctly identify backgrounds that were not necessarily static such as water

We can conclude the simplest version, implemented in **Exercise 1** had the best performance, since it had a better balance between all the measures. Also, the overall performance seen in **Table II** shows it.

VII. TIME LOG

A. Foreground Segmentation Mask

The time spent on this part of the laboratory session is 2 hours and 30 minutes for the gray space implementation: 30 minutes to understand the code and the structure of the program, 1 hour and 30 minutes to code the algorithm and test it, 30 minutes to tune the threshold value. The implementation of the RGB version required 30 minutes only.

B. Progressive Update of the Background Model (Blind and Selective Modes)

The time spent on this part is 2 hours 50 minutes in total for the gray space version: 1 hour to implement a blind mode, 1 hour and 30 minutes to code the selective mode and fix the errors, 20 minutes to tune the alpha parameter. It took 2 hours to modify the code for the RGB version.

C. Suppression of Stationary Objects

It required 4 hours to implement the suppression algorithm and test it, and fix some bugs.

D. Shadow Detection based on HSV

To implement the algorithm for detecting shadows, it took 7 hours in total: 2 hours to read the article [1] and understand the idea, 3 hours to code the algorithm, 1 hour to fix errors and debug them, and 1 hour to tune the parameters of the algorithm.

E. Single Gaussian

The time spent on this part of the laboratory session is 8 hours in total: 2 hours to read the article [3], 3 hours to code the algorithm and 2 hours to debug it, and 1 hour to tune the parameters.

F. Testing

It took around 5 hours to understand the Matlab test scrip, run *baseline*, *shadow*, and *Dynamic Background* categories and save the results.

G. Report

Around 7 hours: writing, analyzing the results, taking screenshots of the frames, and editing.

References

[1] R. Cucchiara, C. Grana, M. Piccardi, A. Prati, Detecting moving objects, ghosts, and shadows in video streams, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (10) (2003) 1337–1342

[2] C. Stauffer, W.E.L. Grimson, "Adaptive background mixture models for real-time tracking", in Proc. of IEEE Computer Vision and Pattern Recognition, June 1999, vol.2, pp. 246-252

[3] C.R. Wren et al., "Pfinder Real-Time Tracking of the Human Body", IEEE Trans. on Pattern Analysis and Machine Intelligence, 19(7):708-785, July 1997