Applied Video Sequence Analysis
Lab 1 “Foreground segmentation”

Hunor Laczkó and Anne-Claire Fouchier

I. INTRODUCTION

This laboratory report presents foreground segmentation algorithms to detect moving object on stationary background. The algorithms were written in C++ using the OpenCV library and Eclipse, and were evaluated with standard measures on the ChangeDetection.NET baseline, shadow and dynamic background datasets. The implemented algorithms include frame difference, selective running average and ghost suppression on both grayscale and RGB images, then shadow detection, and finally the unimodal and multimodal Gaussian models.

II. METHOD

During the lab, three different approaches were implemented. First, simple foreground segmentation algorithms were experimented with. Then a shadow suppression algorithm was implemented. Finally, an advanced background subtraction model was introduced, using the unimodal Gaussian model.

A. Simple foreground segmentation

1) Frame difference: A foreground binary mask is extracted by taking the difference between each frame and the background. Each pixel whose difference is higher than a chosen threshold τ is considered belonging to the foreground. The background is considered to be the first frame of the video.

$$| Image_t[x,y] - bkg[x,y] | > \tau \Rightarrow fgd_t[x,y] = 1 \quad (1)$$

$$| Image_t[x,y] - bkg[x,y] | \leq \tau \Rightarrow fgd_t[x,y] = 0 \quad (2)$$

2) Selective running average: In the previous method, the background is not updated. Here, the previous method is used but an update to the background is introduced based on an adaptation speed α.

$$bkg_{t+1}[x,y] = \alpha Image_t[x,y] + (1-\alpha) bkg_t[x,y] \Leftrightarrow fgd_t[x,y] = 0 \quad (3)$$

3) Ghost suppression: This method builds up on the previous methods and allows the removal of stationary objects that appear or are removed from the scene. To do so, each pixel is attributed a counter. The corresponding counter is incremented every time the pixel is detected as foreground, and reset every time the pixel is detected as background. When the counter reaches a threshold, the pixel becomes a background pixel and its counter is reset.

B. Shadow suppression

During simple detection, shadows also appear as foreground and this makes the foreground mask deformed and less precise. while increasing the complexity of subsequent operations. To avoid this, a chromacity based shadow detection is applied. This method detects changes in the intensity values of a given frame and the background frame. If a value changes but there is no or only a small difference in chromacity, then that corresponding pixel is classified as shadow in the shadow mask. The foreground mask is created by taking the intersection of the shadow mask and the background subtraction mask.

C. Advanced background substraction

1) Unimodal Gaussian: This method allows the elimination of Gaussian noise. Each pixel is attributed a Gaussian distribution representing the background. The initial values are estimated from the initial sequence of scene images with no moving objects. The mean and standard distribution are updated using selective update.

2) Gaussian Mixture Model: This method uses the same idea as the previous one but expands it by having multiple Gaussians represent a pixel this way it can remove background flickering (movement of leaves, water, blinking lights...).

III. IMPLEMENTATION

Different boolean flags are used to choose which part of the project to run, meaning which method is to be used. This way, everything remain in one project, so maintaining multiple projects were not required and switching between different modes is easy. The flags can be found *Lab1.0AVSA2020_2.cpp* from line 105 and *fgseg.hpp* from line 80. Only the ones that have to be set true will be mentioned in each subsection, the others are false. During the implementation, only matrix operations were used for effectiveness and to minimize copying and creating a large number of matrices.

Initialization is done in the *bgs::init_bkg()* function, where the necessary variables are initialized with zeros and the correct number of channels, while the background is initialized with the grayscale or the color version of the first frame.

A. Functions for frame difference

To run this method, *_simpledetection = true* has to be set. In the main file, the threshold value can be changed by modifying the *tau* value. It works both in grayscale and rgb. The method is implemented inside the *bkgSubtraction* function. The difference between the frame and the background, which is the first frame, is thresholded with this value to create our background subtraction mask (bgsmask).

B. Function for selective running average

To run this method, *selective_update = true* has to be set along with an alpha value, which is the learning rate. It works both in grayscale and rgb. The bgsmask is calculated in the previous method and is used to create a logical mask that allows and makes easy updating only the pixels that belong to the foreground. The other pixels are kept with the inverse selection. The update slowly incorporates foreground pixels into the background.

C. Function for ghost suppression

To run this method, *_ghosting = true* has to be set, along with a threshold for the ghost detection. In grayscale, a counter matrix of the frame size is created and the same logical background mask as before is used to increase the counter of the pixels that are detected as foreground and zero them out if they are background. If the counter reaches the previously given threshold, the respective pixels have to be updated. In order to do this, another logical matrix is created and holds ones where the pixels have to be updated. With this, the pixels from the background that are going to be updated are deleted and then replaced with the corresponding pixels from the current frame. If a pixel was replaced, the counter is also reset.

The color version works similarly, but 3 channel matrices are used which require some changes to the code. The background mask has to be tripled, to have a three channel counter matrix so it can be directly applied to the 3 channel background.

D. Function for shadow suppression

To run this method, *_shadowdetection = true* has to be set along with *alpha, beta, tau_s, tau_h* parameters. The method is implemented in the *bgs::removeShadows()* function. First the background and the current frame are converted to the HSV color space, then split into the three channels so they can be accessed independently. The matrices are converted to floating numbers because decimal values will be calculated with them. The update condition consist of three parts that are calculated separately. The logical matrices are then converted back to unsigned char and divided by the maximum value, 255, to hold 0 and 1 values. The condition is checked by adding the three partial results together. If the value is 3 (meaning every partial result contained a 1), the condition is met. The shadow detection is only done in color mode, because the color information is needed for it.

E. Function for simple gaussian

To run this method, *_simplegaussian = true* has to be set along with an *alpha* learning rate and *std_coeff* which will determine how many times the variance value is taken for the threshold. This method only works in grayscale mode. The variance is initialized with the variance of the first frame, while the initial mean is the actual value of the pixels of the first frame. Again, floating point matrices have to be used because gaussian values and their differences are calculated. The differences will determine the bgsmask, and the pixels that were detected as foreground will be updated by updating their gaussian's mean and variance.

F. Function for gaussian mixture model

To run this method, *_multimodal = true* has to be set along with an *alpha* learning rate, *Wth* weight limit to determine the background gaussians and *K* which determines the number of gaussians used per pixel. This method only works in grayscale mode. Separate classes were created to make the implementation easier and cleaner. The *GMM* class represents the background model, it contains a matrix of *gmm* classes which represent each pixel with *K* number of *Gaussians*.

G. Running the code

The code consists of the seven source files contained in the src folder and the accompanying makefile. Using the makefile, the code can be easily compiled and then it only has to be run. For running on other systems, the paths given in line 45 and 55 of main.cpp have to be changed for the corresponding paths. Also for running different versions of the program (frame difference, selective update, etc), the corresponding boolean flags have to be changed as mentioned above, and the code needs to be compiled again. Further information about how the code works can be found as comments at each relevant location next to the code.

IV. DATA

To evaluate the foreground segmentation algorithms, three datasets from changedetection.net are used. The simple foreground segmentation algorithms are tested on the baseline dataset. This dataset includes four video sequences : highway, office, pedestrians and PETS2006. These include simple situations, with minimal complex attributes.

The shadow suppression method is evaluated on the shadow dataset, which includes the following videos sequences : backdoor, bungalows, busStation, cubicle, peopleInShade and copyMachine. These frames record indoor and outdoor scenes, with shadows of various intensities.

Finally, the Dynamic Background dataset is used to evaluate the advanced background substraction method. It includes the boats, canoe, fountain01, fountain02, overpass and fall video sequences. These frames include water and leaves,

which should remain in the background despite their periodic movement.

Method	Dataset	Video sequences
Simple foreground segmentation	baseline	highway office pedestrians PETS2006
Shadow suppression	shadow	backdoor bungalows busStation cubicle peopleInShade copyMachine
Advanced background substraction	Dynamic background	boats canoe fountain01 fountain02 overpass fall

V. RESULTS AND ANALYSIS

During the evaluation of each task various different values for all the possible parameters were tried and evaluated subjectively. This subjective analysis allowed the trial of a large number of value combinations and getting a visual result for them almost immediately. This way, multiple values that did not provide viable result could be easily and rapidly eliminated. In the end, for each variable, a small range was determined in which the results were generated. Later these result were evaluated with the help of the provided Matlab script. The results contained different measures which were compared and evaluated, the most important ones are mentioned below along with the observations made. The FMeasure was the base of the evaluation as it takes into account the recall and the precision, although all measures were evaluated.

1) Frame difference: This method has been tested with three different thresholds. The results are available in Table I in the Appendix. Increasing the threshold meant that more noise was canceled, but only a part of the foreground was detected, while decreasing it introduced too much noise and made the segmentation unusable. With a threshold of 50, the Fmeasure reaches 0.85, which was the most balanced result. The results can be seen on Figure 1. However, this implementation shows worse results than any results performed on the baseline dataset by other methods found on the changedetection.net website. This was to be expected as this method is very simple.

2) Selective running average: This method performs worse than the simpler method. This is surprising as we are updating the background. The results can be found in Table II in the Appendix. However, it is interesting to mention that this method performs better on the original color images than when converted to grayscale. We reach a maximum of 0.53 for the Fmeasure with a threshold of 50 and an alpha at 0.03. The results can be seen in Figure 2. This means that a slower learning rate proved more beneficial, otherwise the background would get updated too soon, and include parts of the foreground too.

Fig. 1. Frame difference with threshold of 40 (left), 50 (middle) and 60 (right)

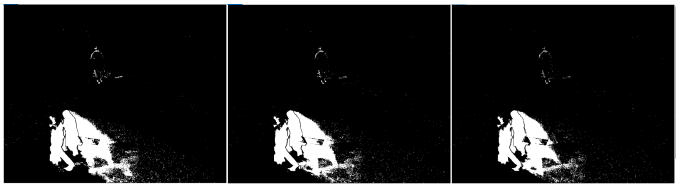

Fig. 2. Selective update with alpha of 0.03 (left), 0.05 (middle) and 0.07 (right). While the backpack in the middle gets incorporated to the background in all cases they perform slightly different on other regions. There is significant ghosting on the scene, this would be resolved in the next method

3) Ghost suppression: This method should help remove disappearing background objects or their traces from the background. It was especially visible on the video with moving objects around and on the desk. During the evaluation, it was shown that the bigger threshold (of 25) worked the best (based on the Fmeasure). It is probable that otherwise other objects or the actual foreground itself would also be identified as "ghost". The results also show that using the color frame improved the segmentation. The numerical results can be found in Table III in the Appendix. The image comparison can be seen in Figure 3.

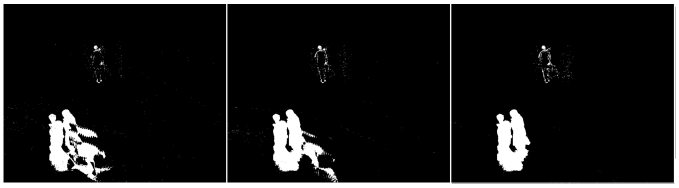

Fig. 3. Ghost suppression with a threshold of 10 (left), 15 (middle) and 25 (right). There is significant improvement with the ghost removal with the increase of the threshold

4) Shadow suppression: With fixed alpha and beta, with respective values 0.5 and 0.9, the best results were obtained when both hue and saturation thresholds were low, i.e respectively 40 and 30. The recall is the most sensitive to the change. With these values, a recall of 0.767 was achieved, and a Fmeasure of 0.66, which is also the best result. With (τ_s=80 ans τ_h=70), the worst recall was achieved, 0.666, and the corresponding F-measure is also the worst one, 0.618.

However, after analysing some pictures from the best and the worst estimations, it turns out that even though the worst estimation removed the shadow better, it also removed a significant part of the foreground object. This model was able to beat only a few of the results on the challenge site.

5) Unimodal Gaussian: This model implementation was not able to beat the models tested on the dynamic background category. One obtained recall is better than the Color Histogram Backprojection (0.75 instead of 0.63) but the F-measure is significantly lower. These results were achieved with a standard deviation coefficient of 0.1 and an alpha of 0.01. With these parameters, there was less foreground included in the background but more background included in the foreground. All in all, the best results were obtained with a coefficient of 1 and an alpha less or equal to 0.01. When the coefficient is too large, equal to 2 in this experiment, the FNR got really high. Positives were not classified correctly.

6) Multimodal Gaussian: The implementation was first tested subjectively on the baseline dataset to see how it handles the easy scenarios. After adjusting the learning rate to *alpha = 0.01* and *Wth = 0.5* it managed to deal with those videos with minimal noise 4. It could also handle the stationary objects, although the reaction to them was significantly slower then previous methods. Using higher learning rate would introduce a lot of noise which after a short time would self generate itself making the whole mask just foreground. For that reason these values were used for further evaluation. The number of gaussians, K, was 3. Trying higher number resulted in a slightly better segmentation but it was decided that it was not worth the extra computational complexity so it was not increased.

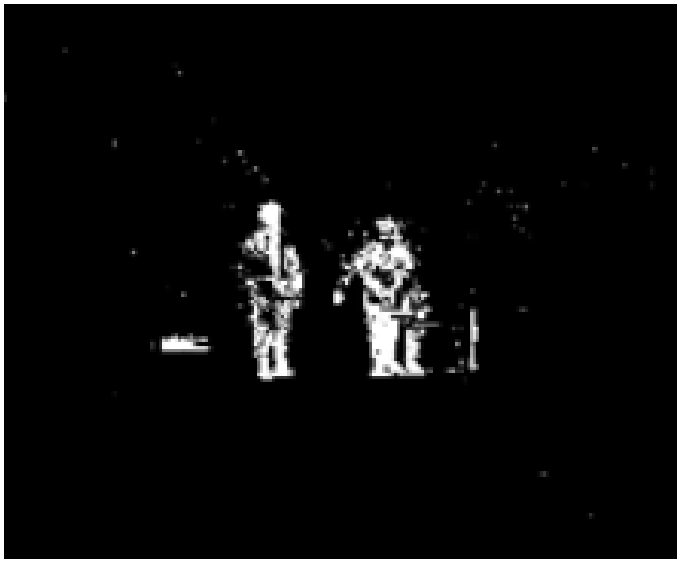

Fig. 4. Hall video with GMM

The implementation was also tested on the *dynamicBackground* dataset. The performance was better than that of the unimodal's but after a given time the stationary background part started to get noisy, which as before self generated itself and ended up making that part of the image all foreground mistakenly. As it can be seen on Figure 5 the moving background (river) is almost perfectly filtered out, the foreground object (canoe) is recognized, although the people on it not perfectly. The problem is in the upper part which is a mostly stationary background and for some reason it gets that noisy ultimately affecting the result in a negative way.

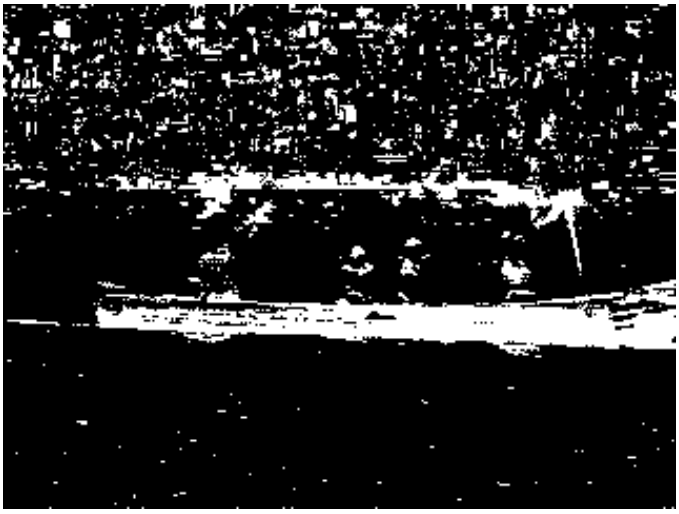

Fig. 5. Canoe video with GMM

Even with the above mentioned drawback, the method performed generally well according to the objective metrics and while it was not better in each metric than the unimodal method overall it got better results: recall = 0.721, specificity = 0.972, FPR = 0.027, FNR = 0.278, precision = 0.192, FMeasure = 0.281.

VI. CONCLUSIONS

After evaluating all the methods, it was found that the simple frame difference method yielded the overall best results. More complex methods, like selective update or ghost suppression, seemed to improve on certain aspects of the detection and adapt to the changing background. However, a lot of performance was lost while detecting the actual objects themselves. This explains why the overall results turned out worse for these methods. However, like all the implemented methods, the simple one was compared to the methods published on the challenge's website and its performance was one of the worst ones.

With the shadow and dynamicBackground categories, the same problems appeared. Effectively filtering the shadows or background also often meant removing big parts of the objects themselves too. But even though these methods were not performing well, they still managed to beat a few of the methods from the website. In the end, this lab project did not perform perfectly but it did implement the studied methods correctly.

VII. TIME LOG

Below the amount of time spent on each aspect of the project is detailed. The times are considered gross times, meaning they also include setting up the environment for the respective task (like starting the base code or the matlab script) and unforeseen challenges (virtual machine crashing, native linux not working properly, eclipse projects mixing together). Most of the work was done together, so the times are considered per student.

1) Frame difference: : 2 labs, meaning 4 hours: getting the base code to run 1 hour, understanding the given code 2 hours, writing the function 1 hour

2) Selective running average: : 3 hours: 2 hours writing the code and 1 hour debugging the matrix operations in the equation

3) Ghost suppression: : 4 hours: 2 hours writing the code, 2 hours converting it to color mode

4) Shadow suppression: 3 hours: 2 hours writing the code, 1 hour debugging

5) Unimodal Gaussian: 4 hours: 2 hours writing the code, 2 hours debugging

6) Gaussian Mixture Model: : 5 hours : 2 hours understanding the model, 2 hours implementing it and 1 hour debugging

7) Evaluation: : 4 hours : running the code with all the different values then running the evaluation

8) Report: 7 hours: compiling all the result together and analyzing them

VIII. APPENDIX

TABLE I
RESULTS WITH THE FRAME DIFFERENCE METHOD

color	yes	yes	yes
threshold	40	50	60
recall	0.8783468878	0.8294314475	0.7687777232
specificity	0.9937823373	0.9962790351	0.9972040783
FPR	0.0062176627	0.0037209649	0.0027959217
FNR	0.1216531122	0.1705685525	0.2312222768
Precision	0.8083156040	0.8818772730	0.9188573295
F-Measure	0.8344390294	0.8523339819	0.8349360589

TABLE II
RESULTS WITH THE SELECTIVE UPDATE METHOD

color	yes	yes	yes
threshold	50	50	50
alpha	0.03	0.05	0.07
recall	0.5224952650	0.4858288702	0.4589423337
specificity	0.9897671924	0.9905373476	0.9915716717
FPR	0.0102328076	0.0094626524	0.0084283283
FNR	0.4775047350	0.5141711298	0.5410576663
Precision	0.6028820161	0.5831265949	0.5888113119
F-Measure	0.5347953564	0.4946454259	0.4749926284

color	no	no	no
threshold	50	50	50
alpha	0.03	0.05	0.07
recall	0.4518173366	0.4184717854	0.3935666966
specificity	0.9919390409	0.9923586952	0.9931250469
FPR	0.0080609591	0.0076413048	0.0068749531
FNR	0.5481826634	0.5815282146	0.6064333034
Precision	0.6248972315	0.5986154651	0.6012346702
F-Measure	0.4998988694	0.4590906542	0.4380456590

TABLE III
RESULTS WITH THE GHOST SUPPRESSION METHOD

color	yes	yes	yes
threshold	50	50	50
alpha	0.05	0.05	0.05
th_gh	10	15	25
recall	0.5576749851	0.5951585198	0.6273573590
specificity	0.9925628925	0.9935528371	0.9947685500
FPR	0.0074371075	0.0064471629	0.0052314500
FNR	0.4423250149	0.4048414802	0.3726426410
Precision	0.6928428408	0.7353652751	0.7945190955
F-Measure	0.5786297320	0.6280916361	0.6815215521

color	no	no	no
threshold	50	50	50
alpha	0.05	0.05	0.05
th_gh	10	15	25
recall	0.5028470302	0.5351139242	0.5613203704
specificity	0.9936582978	0.9945841687	0.9955625809
FPR	0.0063417022	0.0054158313	0.0044374191
FNR	0.4971529698	0.4648860758	0.4386796296
Precision	0.7044277304	0.7514114531	0.8141595442
F-Measure	0.5519922911	0.5983727924	0.6467790272

TABLE IV
RESULTS WITH THE SHADOW SUPPRESSION METHOD

a	0.5	0.5	0.5
b	0.9	0.9	0.9
tau_s	80	60	40
tau_h	70	70	70
recall	0.7432910019	0.7483988054	0.7559488380
specificity	0.9799566872	0.9798883495	0.9795218615
FPR	0.0200433128	0.0201116505	0.0204781385
FNR	0.2567089981	0.2516011946	0.2440511620
Precision	0.6148153944	0.6150769653	0.6119882524
F-Measure	0.6505818794	0.6531557593	0.6551587296

a	0.5	0.5	0.5
b	0.9	0.9	0.9
tau_s	80	80	80
tau_h	30	50	90
recall	0.7594852404	0.7534683405	0.7339806073
specificity	0.9797657459	0.9798494137	0.9800548307
FPR	0.0202342541	0.0201505863	0.0199451693
FNR	0.2405147596	0.2465316595	0.2660193927
Precision	0.6178287119	0.6168979174	0.6131976617
F-Measure	0.6585821273	0.6555993541	0.6461572880

a	0.3	0.5	0.5	0.5
b	0.7	0.9	0.9	0.9
tau_s	80	40	100	120
tau_h	70	30	90	110
recall	0.6656795923	0.7670166899	0.7300021045	0.7205187480
specificity	0.9826636136	0.9793390614	0.9800745806	0.9801195275
FPR	0.0173363864	0.0206609386	0.0199254194	0.0198804725
FNR	0.3343204077	0.2329833101	0.2699978955	0.2794812520
Precision	0.6222296759	0.6134444656	0.6124031308	0.6101750639
F-Measure	0.6179969467	0.6601024119	0.6438488721	0.6384287127

TABLE V

RESULTS WITH THE UNIMODAL GAUSSIAN METHOD

coef	2	1.5	1
rate	0,01	0,01	0.05
recall	0.0589513262	0.1446144912	0.1858097491
specificity	0.9998291055	0.9993482076	0.9966333073
FPR	0.0001708945	0.0006517924	0.0033666927
FNR	0.9410486738	0.8553855088	0.8141902509
Precision	0.6425653285	0.5965488629	0.3616806144
F-Measure	0.1037050493	0.2175246930	0.2190455284

coef	1	1	0.5
rate	0,01	0,005	0.01
recall	0.3370169651	0.4027824591	0.7500661308
specificity	0.9941781026	0.9917753354	0.8103820533
FPR	0.0058218974	0.0082246646	0.1896179467
FNR	0.6629830349	0.5972175409	0.2499338692
Precision	0.3839676411	0.3601179487	0.0511250072
F-Measure	0.3213064726	0.3401786735	0.0924826213