APPLIED VIDEO SEQUENCES ANALYSIS - LAB4

Sergio Romero Tapiador & Jan Sieradzki

I. Abstract

This laboratory is consisting from 6 sub-tasks, which are focused on implementing a single-object tracker based on histograms. In particular, we accomplished the following tasks:

1) Color-based tracking: code implementation
2) Color-based tracking: analysis over real data
3) Gradient-based tracking: code implementation
4) Gradient-based tracking: analysis over real data
5) Color&Gradient-based tracking: code implementation
6) Color&Gradient-based tracking: analysis over real data

These tracking algorithms were implemented with usage of OpenCV (C++).

This report is organised as follows: in section II-A the method of the Color-based tracking is described as well as our implementation of this algorithm. In section II-C there is explained the methodology of experiments, which we conducted and analysed in section II-D. In analogical order we presented our work regarding Gradient-based tracking (section III) and Color&Gradient-based tracking (section IV). Finally, in the sections V and VI the conclusions and the time log are presented, respectively.

A. User Manual

To run all of the tasks, please go to each of the `Lab4AVSA2020` folder and type "make" in order to get the executable. The name of the .exe file will be the same as the folder name. In order to use algorithm, you need to edit the path lines for the dataset and result folders beforehand or pass the path to a video as argument (after any intention of execution of program, there is printed exemplary form of path to video). Nevertheless, in purpose to change default parameters or set result folders, the main.cpp file edition is necessary.

II. Section I (for tasks 4.1 & 4.2)

A. Method & Implementation

In this section we briefly describe the Color-based tracking method we have developed. Afterwards, there are presented implementation details. We have previously studied how it works and how we can code it in a efficient way from lecture materials [1] and paper [2]. The description of the method and implementation will be more detailed for this method in compare to following sections, as the others algorithms share a lot in common with the one described in this section.

This method is based on comparing the potential object localisation's color histogram with previously modelled template-histogram. In our case, the template is defined by first frame ground truth and is not updated anymore, hence the dimensions of the predictions will be constant, what is not very realistic assumption, as the objects often change their shape and size. For each frame there is returned, as a object localisation prediction, bounding box with the highest template comparison score. During every frame, histogram-based tracking consists from following mains steps:

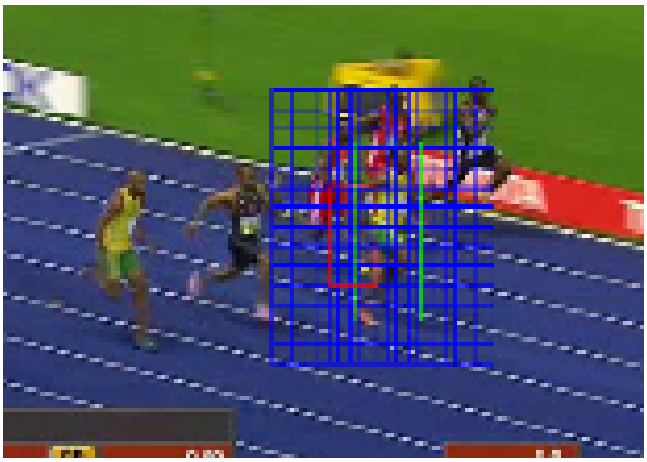

Fig. 1: Grid of candidates, size 10x10, stride 10px

1) Generate the candidates for the object position: The efficient way of candidates generating is open question in the research environment and may be conducted in multiple manners. We have decided for generating them in the grid (exemplary grid fig. 1), which is created around the previous frame object's location. In particular, the grid is in the form of square, which is built by shifting rectangles by the constant stride. It is important to include exact previous frame prediction to candidates set, since the tracked object may not change it's position - when the side is odd, this rectangle will be the center of the new grid - when the side is even, first we create grid $side - 1$ centered on previous prediction and afterwards we are adding one row to the bottom of the grid, and one column to it's right side. At the end of procedure, we are controlling if all generated candidates belongs to frame - if they exceeded image boundary, such rectangle is deleted from candidate list - otherwise the program would crash.

2) Create histogram for every considered candidate: After obtaining candidates for the object, we need to extract features from them. In this method we are calculating color histograms from previously selected image's color channel. In our implementation it is possible to choose one channel from following: gray-level from RGB, H channel (HSV), S

channel (HSV), R channel (RGB), G channel (RGB) and B channel (RGB). As we want to make tracker more roboust, for example for scale changes, we are normalizing obtained histograms to have values from range [0.01,1] (we are not including 0, because during distance calculation there would occur 'divide by 0' problem. Before starting tracking, user is passing parameter, which sets amount of bins in histogram, what directly influence tracking quality.

3) Score every histogram, calculating it's distance to the template: After calculating histogram for every candidate, for each one there is computed distance between template. The distance for color-histogram is obtained with Battacharyya distance (formula 1).

$$d(H_1, H_2) = \sqrt{1 - \frac{1}{\sqrt{\bar{H}_1 \bar{H}_2 N^2}} \sum_I \sqrt{H_1(I) \cdot H_2(I)}} \quad (1)$$

Finally, as the result for a particular frame tracking, there is returned rectangle-candidate, which has the smallest distance to template.

B. Implemetation details

Algorithm was developed with the usage of C++ library OpenCV2. The implementation developed in this laboratory assignment is structured in 4 modules (4 different C++ files):

1) **ShowManyImages**: this module is in charge of showing the different processed images of each frame in the video sequence. This module is not described in detail, as it was already developed by the AVSA teachers.
2) **Utils**: this module contains readGroundTruthFile and estimateTrackingPerformance functions. First one is responsible for reading ground truth files and creating list of ground truth bounding boxes from it. Second one compares two lists of bounding boxes to estimate their overlap using the criterion IOU (Intersection Over Union). This module is not described in detail, as it was already developed by the AVSA teachers.
3) **ColorBasedTracker**: this module corresponds with the color-based tracking described earlier and the functions implemented in this file provide the algorithms to compute object's position predictions for given frames. Moreover, this module has been created as a class in order to call and manage it from different external modules. Therefore, the procedure to create this class is the same as any C++ class, with an initialization and a destruction method. During initialization the ground truth from first video's frame is taken as a template, and later function execute_tracking_step can be used to propagate through next frames. Inside execute_tracking_step there are called other functions, which are realising steps described previously. This class has 5 parameters to tune:
   - BINS_NUMBER amount of bins that will be used in histograms
   - CANDIDATE_GRID_SIDE is side of the grid used to candidates generation
   - GRID_PIXEL_STRIDE the Pixel Dist. between candidates rectangles generated in grid
   - CHANNEL_TYPE is the id of channel of interest for tracker
   - NORMALIZATION tells, if histograms should be normalized (recommended to always use normalization)
4) **Lab4.1AVSA2020**: this module corresponds with the main function. After initializing and declaring some variables (related to paths, video processing, etc.) the tracker class is created. Then, for each frame processed, the tracker step is executed. With usage of ShowManyImages, during algorithm execution we can observe tracking; color histograms of candidate, ground truth and template; cropped object's rectangle-prediction from the image; candidate grid on the image. Afterwards, the performance (time and IOU) are printed.

C. Experimental methodology

Three different videos have been used as the experimental methodology. The first one is referred to the video utilized for the first 4.1 task, where the tracking object corresponds to a person (in this case, the runner Usain Bolt in a race). Also, this sequence will be employ to tune every different method in this programming task, where occlusions among runners are the main challenge. Other difficulties to track the object are related to the camera movement and changes in the trajectory due to the speed of the runner.

- **Car**: the second object to track corresponds with a car (video called car1) which drives over a road and the main challenge in this video is the camera movement: in some frames, the camera (which is recording from another car) experiences sudden movements which implies a slight blurring. Therefore, the target car appears in different parts of continuous frames. Also, this blurring causes noise in the frame sequence. Furthermore, lighting conditions are not properly: the sky is overcast and the light intensity is lower than optimal conditions and consequently, as some background appears similar to the car (similar car colors, pedestrians with similar color features), the tracker could be distracted with these kind of situations.
- **Sphere**: finally, in the third video, a transparent sphere has to be tracked (sphere video). This sphere is hold by a person who is continuously moving it. The main feature that identifies this object is its color: its transparency plays a challenge in this tracking task as the background of the sphere is always changing when it's being moved. Moreover, movements are not linear and in some frames, the camera zooms in to the object.

D. Results and Analysis

In this section, the results and analysis of the task 4.2 are detailed. A similar procedure has been followed for all the videos, including a first experiment with all the different color channel and taking into consideration the best one. Then, after choosing the best channel, a second experiment has been carried out: in this case, different parameters such

as number of bins or number of candidates (among others) have been tuning in order to obtain the best performance according to each video. For the task 4.2 videos, the initial parameters are the same that those tuned in task 4.1.

Note: all the experiments have been executed with a slow computer. Therefore, the computational cost of some executions are really higher.

1) Bolt: The initial parameters values for this video are the following:

- Number of bins: 16
- Number of candidates: 10
- Pixel distance between candidates: 2
- Color channel: 0 (gray)

2) Car: As we mentioned before, in this video sequence prevails low intensity color, normally associated to the blue. The initial parameters for the first experiment are the same as in Bolt video. In general, the car is being tracked until the camera experiments different movement such as several shakes. In this moment, the tracker fails in several frames.

In Table I we can see the performance obtained for each channel, where both S channel (from HSV) and B channel (from RGB) are the best ones with a track performance around 49%. Therefore, both saturation and blue values show that they perform better due to the lighting conditions.

Track. perf. (IOU)	ms/frame	Channel
0.2380	3.8757	0 - Gray
0.0603	5.6643	1 - H
0.4852	**4.9726**	**2 - S**
0.0929	3.1739	3 - R
0.3313	3.2207	4 - G
0.4910	**3.6526**	**5 - B**

TABLE I: Experiment one - Car video. The average tracking performance is in terms of Intersection Over Union. In **bold**, the best performance obtained.

The second experiment will be only using these two channels. In Table II we can visualize the performance obtained with different parameters (the non-mentioned parameters are the same as previously): in this case, changing the number of candidates from 10 to 50 will be enough to increase the final performance to a percentage around 64%. Furthermore, higher number of candidates implies more computational cost.

Track. perf. (IOU)	ms/frame	Channel	Nº Cand.	Pixel Dist.
0.5251	51.1215	5 - B	50	2
0.6379	**53.7060**	**2 - S**	**50**	**2**
0.6460	**49.8284**	**2 - S**	**50**	**4**
0.5883	403.2550	2 - S	140	2

TABLE II: Experiment two - Car video. The average tracking performance is in terms of Intersection Over Union. Nº Cand = Number of Candidates. Pixel Dist. = Pixel Distance between candidates. In **bold**, the best performance obtained.

In Figure 2, two different scenarios are shown: at the top a good performance can be seen whereas at the bottom, a bad case due to the shake of the camera is shown.

Fig. 2: Examples of good (top) and bad (bottom) cases of car video.

3) Sphere: In this case, as the background of the sphere is continuously changing, we cannot rely on any color channel or component. However, although we achieved a good performance on gray colors, green channel shows the best one (See Table III) with around a 50%. The main problem here resides in the accuracy of the area: although the tracker is able to track the object, the area does not perfectly coincide with the ground truth area.

Track. perf. (IOU)	ms/frame	Channel
0.4640	**3.4548**	**0 - Gray**
0.1591	4.2778	1 - H
0.3674	4.1880	2 - S
0.4197	3.2748	3 - R
0.5038	**3.0288**	**4 - G**
0.4292	3.1838	5 - B

TABLE III: Experiment one - Sphere video. The average tracking performance is in terms of Intersection Over Union. In **bold**, the best performance obtained.

Consequently, we have taken into consideration this Green channel for the second experiment. In this case, we have changed three parameters: number of bins, number of candidates and pixel distance (the non-mentioned parameters are

the same as previously). The number of candidates is the key parameter for having a better performance, showing that this value as higher as it is, best performance is achieved (See Table IV). Nevertheless, as we have obtained similar results, we have to take into account the computational cost, therefore 50 number of candidates are fine.

Track. perf. (IOU)	ms/frame	Nº Bins	Nº Cand.	Pixel Dist.
0.5781	**43.5634**	**16**	**50**	**2**
0.5775	**158.7710**	**16**	**100**	**2**
0.5730	**44.8312**	**10**	**50**	**2**
0.5773	**38.2626**	**16**	**50**	**5**

TABLE IV: Experiment two - Sphere video. The average tracking performance is in terms of Intersection Over Union. Nº Cand = Number of Candidates. Pixel Dist. = Pixel Distance between candidates. In **bold**, the best performance obtained.

Finally, two different cases (good and bad) are shown in Figure 3.

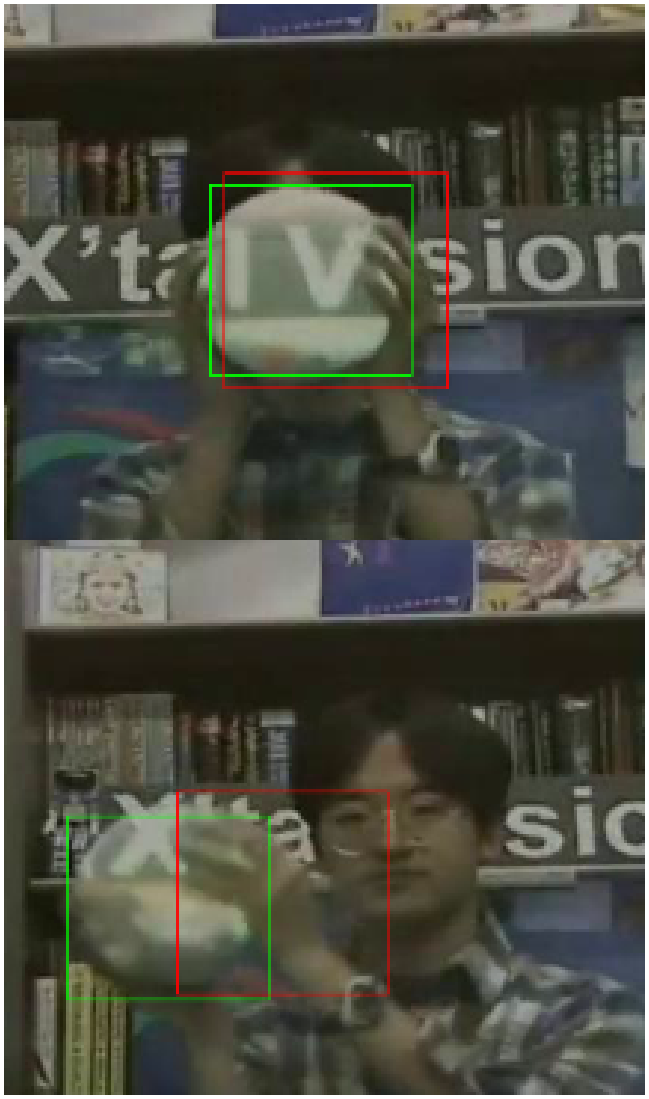

Fig. 3: Examples of good (top) and bad (bottom) cases of sphere video.

III. Section II (for tasks 4.3 & 4.4)

A. Method & Implementation

In section II we developed Gradient-based tracking. In general the whole concept of the tracker is similar to Color-based tracker, but here gradient features are in the usage instead of color histogram. In particular, we are using Histogram of Oriented Gradients (HOG) feature descriptor to estimate object positions through video. This technique counts occurrences of gradient orientation in localized portions of an image. To do that, algorithm first divides image into small connected regions called cells, for which a histogram of gradient directions is compiled - later these histograms are concatenated into descriptor. The size of descriptor depends from image size, and among others on bin amount, which is parameter in our implementation.

Another important difference in compare to color tracker, is the distance measure. In this case we are using L2 distance (formula 2), instead of Battacharyya distance.

$$d(H_1, H_2) = \sqrt{\sum_I (H_1(I) - H_2(I))^2} \qquad (2)$$

Moreover, we are operating only on gray scale channel, so we should not change this channel for any other in class parameters (although we have left such option in the code for experiment reason, nevertheless we want to mention that change of this parameter will compromise tracking and is not recommended).

The last difference between color tracker and tracker described in this section, is lack of normalization, since descriptor is not really histogram and such operation does not have sense here (although similarly, like with channel parameter, we have left possibility to turn normalization on for experiment reasons, though it is highly recommended to turn it off).

The rest of tracker methodology and implementation details are the same as in the section II-A.

B. Experimental methodology

The experimental methodology is similar to the Section I: 3 different videos have been chosen in order to perform the developed tracker. Also, the first video is the same as in Section I, where Usain Bolt runs a race.

- **Basketball**: secondly, the camera records a basketball match where a player from Boston Celtics has to be tracked. There are some challenges in here: first of all, there are another 5 players from the same team who are recorded and consequently, there are similar color features among them; players (and referees) with similar color skin also appears in the video sequence; several occlusions among players; the trajectory of the player is continuously changing ; there is also camera movement that records the match. Therefore, these challenges will hinder the tracker task.
- **Ball**: finally, the third video corresponds with a scene where a ball is hit towards a goal (ball1 video). As the camera is close to the shooter, the ball is going to be

farther and will be smaller than in the beginning. Apart from this first challenge, the projected trajectory of the object is not linear as the ball draws an effect ones. On the other hand, the goal has similar color features as the ball and all these mentioned difficulties are the main challenges to approach.

C. Results and Analysis

In this section, the results and analysis of the task 4.4 are detailed. A similar procedure has been followed for all the videos, including a first experiment with all the different color channel and taking into consideration the best one. Then, after choosing the best channel, a second experiment has been carried out: in this case, different parameters such as number of bins or number of candidates (among others) have been tuning in order to obtain the best performance according to each video. For the task 4.4 videos, the initial parameters are the same that those tuned in task 4.3.

Note: all the experiments have been executed with a slow computer. Therefore, the computational cost of some executions are really higher.

1) Bolt: The initial parameters values for this video are the following:

- Number of bins: 9
- Number of candidates: 10
- Pixel distance between candidates: 2
- Color channel: 0 (gray)

2) Basketball: In this scenario, although we can think that the green channel could perform well, the results obtained are really different than expected. As we can see in Table V, green channel gets a bad perform (around 8%) whereas Saturation channel achieves the best one with an average of 61.78%. This result shows that the main factor in here is the relationship between the player and the background scenario (in this case the basketball pitch).

Track. perf. (IOU)	ms/frame	Channel
0.0903	31.4624	0 - Gray
0.1036	28.9202	1 - H
0.6178	**32.8496**	**2 - S**
0.0913	25.9526	3 - R
0.0812	31.9825	4 - G
0.4292	3.1838	5 - B

TABLE V: Experiment one - Basketball video. The average tracking performance is in terms of Intersection Over Union. In **bold**, the best performance obtained.

Following the procedure, some parameters have been changed in order to get different performances. As we can see in Table VI, although the number of candidates are not relevant, the number of bins for histograms plays a key role: reducing this value to 3 and 7, the performance are higher than previously (2 more points). This behaviour indicates that the method does not need huge changes in the object (in this case the basketball player) in order to have better performances.

The behaviour of the tracker performs well in almost all the frames (See top Figure 4). However, in the final frames, when there are a lot of players close to the target one, the tracker fails and identifies another player from Toronto team with similar skin color (See bottom Figure 4).

Track. perf. (IOU)	ms/frame	Nº Bins	Nº Cand.	Pixel Dist.
0.4298	669.0530	9	50	2
0.6281	**32.8867**	**7**	**10**	**2**
0.6422	**35.0154**	**3**	**10**	**2**
0.5446	30.7947	3	10	3

TABLE VI: Experiment two - Basketball video. The average tracking performance is in terms of Intersection Over Union. Nº Cand = Number of Candidates. Pixel Dist. = Pixel Distance between candidates. In **bold**, the best performance obtained.

3) Ball: In this video, as we mentioned before, the main challenges are the size of the ball (which decreases) and the similar color features when the ball enters the goal. Before that, the tracker achieves great performance. Consequently, there is not a predominant channel to focus on as the performance are really low (below 10%) in the first experiment. Following a similar procedure, we are going to take both H and S channels as they have performed better results (See Table VII).

Track. perf. (IOU)	ms/frame	Channel
0.0519	7.0793	0 - Gray
0.0609	**9.8478**	**1 - H**
0.0586	**9.2500**	**2 - S**
0.0526	6.6467	3 - R
0.0513	6.5532	4 - G
0.0513	6.5071	5 - B

TABLE VII: Experiment one - Ball video. The average tracking performance is in terms of Intersection Over Union. In **bold**, the best performance obtained.

After several trials, we have decided to tune the tracker changing the number of bins and candidates as we want to cover a larger area. As a result, although we can not achieve good results, we have doubled the performance from experiment one as both number of bins and candidates have increased. In Table VIII we can see the results obtained, with a 18.73% as the best model.

Track. perf. (IOU)	ms/frame	Nº Bins	Nº Cand.	Channel
0.1523	14.4502	9	15	1 - H
0.1532	15.2831	9	15	1 - H
0.1703	14.7684	9	15	2 - S
0.1873	**16.4917**	**7**	**17**	**2 - S**

TABLE VIII: Experiment two - Ball video. The average tracking performance is in terms of Intersection Over Union. Nº Cand = Number of Candidates. In **bold**, the best performance obtained.

Similarly as previous videos, we can visualize in Figure 5 both scenarios: at the top, a good case where the ball is being tracked; at the bottom, a bad case where the tracker has failed. In this last case, the tracker can not distinguish between the ball and the white net as they have similar color features.

Fig. 4: Examples of good (top) and bad (bottom) cases of basketball video.

Fig. 5: Examples of good (top) and bad (bottom) cases of ball video.

IV. SECTION III (FOR TASKS 4.5 & 4.6)

A. Method & Implementation

The last method that is concerned in this report, is fusion between Color and Gradient trackers. That means, that for every generated candidate, there are computed both features - color histogram and HOG. The scoring of the candidates is analogical to the previous implementations of the both respective methods, the only part to precise is the way of creating combination of these two score into one value. In our implementation, after calculating both scores separately, we are standardizing them (scores will sum up to 1), by dividing every score by sum of scores for given feature (we can use that formula, since all distances are bigger or equal 0): $S_i = \frac{S_i}{\sum_I S_i}$. Afterwards, we are calculating new, joint score from standardized distances with formula 3, where w means weight parameter.

$$dist^i_{JOINT} = w * dist^i_{COLOR} + (1 - w) * dist^i_{HOG} \quad (3)$$

Therefore, by tuning $w \in [0, 1]$ parameter, which is given as the input for algorithm (alongside with other parameters for color - section II-A and HOG - section III-A), we can tune how much fusion tracker will depend on HOG/Color features, e.g. $w = 0$ is equivalent to using only HOG-based tracker from section II-A, $w = 1$ means using only color-based tracker from section III-A, wheres $w = 0.5$ means that both modules have equal influence on candidate choice.

The rest of the tracker's methodology, parameters and implementation details are the same as in the section II-A.

B. Experimental methodology

The experimental methodology is similar to Sections I and II: in this case, 4 different videos have been chosen in order to perform the developed tracker. Also, the first video is the same as in Sections I and II, where Usain Bolt runs a race.

- **Bag**: in the second video, a white bag has to be tracked (bag video). Although the complexity of this video sequence is low, there are some challenges to face: the shape of the bag is continuously changing as it has not a solid form and the wind also deforms it; on the other hand, both camera and bag are moving in almost all the video and therefore it implies an extra difficulty in the process of tracking.
- **Ball**: the third video consists on a scene where two people are passing between them a red ball with white hexagons (ball video). The scene is quite easy to track as the ball movement is slow and smooth and its size is quite big. Also, the background colors are really different from the ball color. Therefore, this video sequence seems to be easy to track.
- **Road**: finally, the last video records a motorbike which drives over an empty road from an helicopter (road video). One of the main challenges in this sequence is related with the size of the object, as it is quite small in comparison with the scene. In addition, there are occlusions (mainly trees and woods) which hide the motorbike over the whole sequence as these structures are close to the road. Other challenge to face corresponds with the color similarity of the road with the object: in this case, light colors are presented in both structures.

C. Results and Analysis

In this section, the results and analysis of the task 4.6 are detailed. A similar procedure has been followed for all the videos, including a first experiment with all the different color channel and taking into consideration the best one. Then, after choosing the best channel, a second experiment has been carried out: in this case, different parameters such as number of bins or number of candidates (among others) have been tuning in order to obtain the best performance according to each video. For the task 4.6 videos, the initial parameters are the same that those tuned in task 4.5.

Note: all the experiments have been executed with a slow computer. Therefore, the computational cost of some executions are really higher.

1) Bolt: The initial parameters values for this video are the following:

- Number of bins: 9
- Number of candidates: 10
- Pixel distance between candidates: 5
- Color channel: 1 (H from HSV)

2) Bag: As we mentioned before, the key challenges here are the continuously changes of the shape of the bag. After several trials, we realize that increasing the number of candidates to 200 or higher does not improve the overall performance. In this case we have selected the R channel from RGB thus it has achieved better results (around 17%). Then, after tuning some parameters like the number of bins and pixel distance, we have improved the performance until 23.40% (See Table IX). In this case, the performance is based on the fusion of both models, with a 0.5 (50%) of weight. In this case, the bag is tracked in the first frames. Then, when the bag has changed its shape, the tracker starts to fail and finally, when it lays on the ground, the model can not detect it with accuracy (it only tracks a small area).

Track. perf. (IOU)	ms/frame	Nº Bins	Pixel Dist.
0.1906	57.3411	7	5
0.2270	54.4505	5	5
0.2340	**124.6660**	**5**	**3**

TABLE IX: Experiment one - Bag video. The average tracking performance is in terms of Intersection Over Union. Pixel Dist. = Pixel Distance between candidates. In **bold**, the best performance obtained.

Taking the best performance, we have executed a second experiment with different weight values. In Table X, we can see different weights values and performances, where values close to 1 means that the fusion tracker uses fully color model and vice versa; values close to 0 means that the fusion tracker uses fully HOG model. As a result, the tracker works better with fully colors model, achieving a final performance around 24.77%.

Finally, in Figure 6 we can visualize both good and bad cases track.

Track. perf. (IOU)	ms/frame	Weigth
0.0754	96.0580	0
0.2340	124.6660	0.5
0.2477	**4.6725**	**1**

TABLE X: Experiment two - Bag video. The average tracking performance is in terms of Intersection Over Union. In **bold**, the best performance obtained.

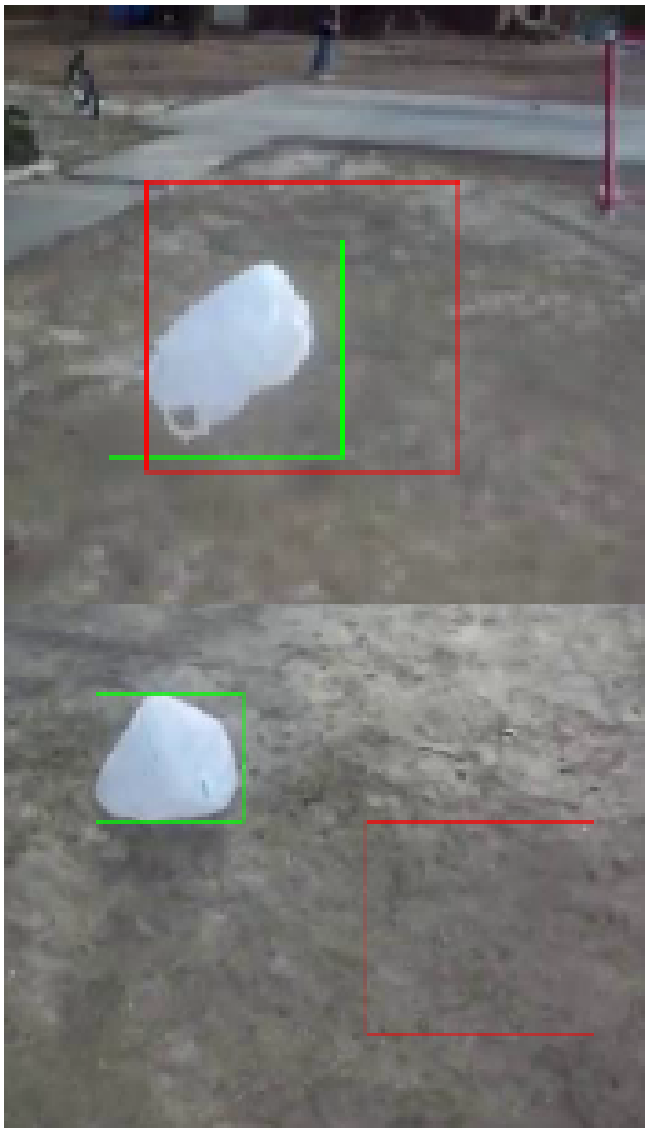

Fig. 6: Examples of good (top) and bad (bottom) cases of bag video.

3) Ball: As we mentioned before, this is one of the easiest videos in this dataset. However, as other videos, the covered area does not match in all the cases and therefore the performance decreases. For the rest, in almost all the frames the tracker can detect and follow the ball with any problem. Taking into account that the ball is red, we have used this channel for the fusion model. Consequently, the results obtained for different weights can be seen in Table XI. Similarly as previously, the color model works better than the HOG ones, with an overall performance around 63%. The initial parameters have been kept.

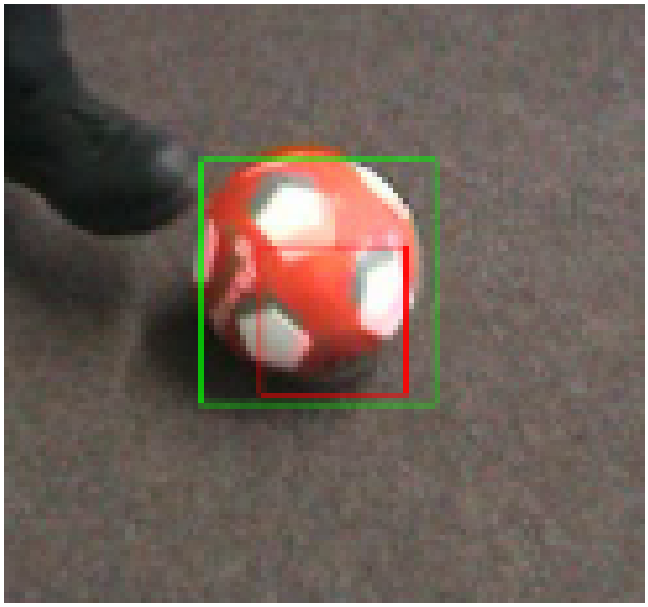

Fig. 7: Examples of a good case of ball video.

Track. perf. (IOU)	ms/frame	Nº Cand.	Weigth
0.1818	3 355.4400	225	0
0.6142	**3 395.6100**	**225**	**0.5**
0.6267	**3 477.7800**	**225**	**1**

TABLE XI: Experiment one - Ball video. The average tracking performance is in terms of Intersection Over Union. In **bold**, the best performance obtained.

In this case, as the tracker performs well, we only have added a good case scenario for the ball (See Figure 7).

4) Road: In this final scenario, the tracker achieves to detect the motorbike despite the far distance from the object. However, when it is occluded, the tracker starts to fail and in some cases, this situation repeats constantly. Taking into consideration the initial scenario, we have just changed the color channel to 3 (R from RGB) as it worked better than others. The main reason resides in the similarity of red colors from the motorbike. In a similar way, the results obtained for the fusion trackers shows that the best performance is when the fully color model is activated, with an overall performance of 37.41% (See Table XII).

Track. perf. (IOU)	ms/frame	Weigth
0.1489	4 483.9300	0
0.3659	4 571.9800	0.5
0.3741	**4 173.2300**	**1**

TABLE XII: Experiment one - Road video. The average tracking performance is in terms of Intersection Over Union. In **bold**, the best performance obtained.

In this case, we can see both good and bad cases when the tracker success (See top Figure 8) and when the tracker fails (See bottom Figure 8).

Fig. 8: Examples of good (top) and bad (bottom) cases of road video.

V. Conclusions

In this last programming task, we have developed two different trackers: one based on colors and the other based on gradients. Finally, a fusion from both has been achieved.

In general, each tracker can detect and follow the object until there is some problem due to the scenario: occlusions, changes in the behaviour or other features. Furthermore, each scenario needs to be tuned it independently because of its conditions.

To sum up, this techniques help us to understand and to improve in the future the state of art approaches which belongs to the Applied Video Sequence Analysis field.

References

[1] Materials from AVSA lecture. (Unit III Visual Tracking: Object modeling, Juan C. SanMiguel).

[2] Elliptical Head Tracking Using Intensity Gradients and Color Histograms, 1998, Stan Birchfield, Computer Science Department, Stanford University, Stanford, CA 94305

[3] Color-based tracking of heads and other mobile objects at video frame rates, P. Fieguth and D. Terzopoulos, In Proc. of the IEEE CVPR, pages 21–27, 1997.

VI. TIME LOG + TASK DIVIDE

In this section we describe the time spent for each student on each task and we show the task division for both members.

A. Time log

1) Jan:

- study for task 0+1+5 - 4H
- reviewing laboratory code - 6H
- conducting experiments + reporting [rest of the tasks] - 12 H
- SUM - 22H

2) Sergio:

- study for task 0+3+5 - 5H
- reviewing laboratory code - 3H
- conducting experiments + reporting [rest of the tasks] - 16 H
- SUM - 24H

B. Task divide

1) TASK 0 - **Sergio**, **Jan**
2) TASK 1 - **Jan**
3) TASK 2 - **Jan**
4) TASK 3 - **Sergio**
5) TASK 4 - **Sergio**
6) TASK 5 - **Sergio**, **Jan**
7) TASK 6 - **Sergio**, **Jan**