Applied Video Sequence Analysis
Lab 4 “Histogram-based object tracking”

Hunor Laczkó and Anne-Claire Fouchier

I. INTRODUCTION

This laboratory report presents Histogram-based object tracking routines, written in C++ using the OpenCV library. First a Color-based model was implemented and tested, then a gradient-based model, and finally a combination of both. Histograms, average accuracy and average processing times were analysed during the fine tuning of the models.

II. METHOD AND IMPLEMENTATION

In this section the methods and their implementations will be presented. Smaller details can be found in the source files in the form of comments.

A. Code structure

This section provides a general overview of the implementation structure and class hierarchy.

The two main classes are the *Tracker* and *Model* classes. The tracker class provides the general methods for realising the tracking itself. This consists of the implementation of the grid search method for finding the best candidate. For this, a model is used, to which a reference is stored as a class member. This reference is to a *Model* instance. The *Model* class is an abstract class. This helps generalize the code and makes the *Tracker* class capable of working with several kinds of models by providing an interface through which the necessary model operations can be reached. This operations are: calculating the difference between the model and a given candidate, updating the model, calculating the corresponding feature (histogram or descriptors) and converting a given candidate to the format used by the given model. For easier use of the class and to make it possible to combine all classes under one structure, it needs to have a separate method to calculate difference for all its children (fusion model will reuse the other ones'). Then each child class will overwrite the corresponding method and leave the others with default implementation which has a dummy implementation. The separate models will be detailed in the next sections.

Besides these classes, only two other class/function group was used, one for showing multiple images in the same window, another for calculating and displaying histograms.

B. Tracker

As mentioned above, this class is responsible for the tracking itself, meaning it implements a search algorithm which, based on a given model, finds the best candidate in a given neighbourhood. This is done in the *GetEstimate()* method. It determines the size of the search area based on the desired number of candidates to be checked, calculates the differences with each of these, then finds the candidate with the minimum distance, returns this location as the estimate and updates the model with this new position2. The calculation of the difference score is done in the following way: in every case the color and gradient difference scores are calculated, normalized and accumulated. These also have a default implementation returning zero difference so that if they are not implemented, they do not affect the accumulated score. This way, each model can implement only the corresponding difference calculation function.

C. Model

Additionally to the difference calculation and model update functions mentioned above, the *Model* base class has two more pure virtual methods that the children classes have to implement. These are the *ConvertPatch()* and *CalculateFeature()* methods. The first one transforms the given candidate image patch to a desired form so that the latter one can calculate some features from it. In the next sections the specific implementations of these functions will be detailed.

D. Color Based Model

In case of the color model the used feature is a histogram. The calculated difference is the the Battacharyya difference between the model and candidate histograms. To calculate the histogram, the candidate image patch is first converted to the desired channel (R,G,B,H or S) then the histogram of this patch is calculated and normalized.

E. Gradient Based Model

For the gradient model, a HOGDescriptior (built-in class for HOG) is used and the features are represented by these descriptors. The difference is calculated by taking the L2 norm if the two descriptors. Converting the patch consists of resizing the patch to a fixed size for the HOGDescriptor, then using that class, the descriptors are acquired.

F. Fusion Model

This model fuses the previous two models together and takes the accumulated scores from both of them. It creates an instance for both classes and only forwards the requests to them. For this class both kind of difference is calculated which later will be accumulated by the Tracker. Here the conversion and feature calculation is not used since they are done inside the respective models.

G. Running the code

The code consists of the 15 source files contained in the src folder and the accompanying makefile. Using the makefile, the code can be easily compiled and then run. It can be run by executing *lab4*. The program expects no input parameters.

The tracker and all relevant parameters can be changed on lines 50-55 in lab4.cpp file.

III. DATA AND CHALLENGES

Tasks	Video sequences
Color Model	sphere.mp4 car1.mp4
Gradient Model	ball2.mp4 basketball.mp4
Fusion Model	bag.mp4 ball.mp4 road.mp4

The video for baseline testing *bolt1*. It consist of panned video of a sprint race where one of the racers is the object to be tracked. The challenges here are that after around the 150th frame the view angle changes because of the panning so the target is seen from a different angle making most of the models lose track at this point. Another challenge is the similarity among the racers, in their skin color and clothing.

The video *sphere* contains a glass ball-like object moved around the frame. The challenge is that the ball acts as being transparent, so the object changes a lot based on what is behind it, making it hard to track because of the variations.

The video *car1* is an on-board footage from a car in which the car in front is tracked. The challenge is in the movement of the camera because it has sudden large deviation from the center.

The video *ball2* is about a football in midair. The problem here is the speed of the ball as it moves fast and the tracker has to be able to keep up with it. Also the ball is small so background can easily affect the candidates.

The video *basketball* is from a basketball game where one of the players is tracked. The main problems here are the occlusions and the fact that the players on the same team are similar to each other. Specially when there is occlusion with a player from the same team the tracker might follow the wrong person afterwards.

The video *bag* is about a plastic bag being blown away by the wind. The main challenge here is that the bag is easily deformed making its shape highly varied.

The video *ball* is about a patterned ball being kicked back and forth. The challenges here are: the pattern changes as the ball rotates, the ball's speed varies a lot, stops then start suddenly, the size of the ball also changes as it gets closer to the camera.

The video *road* follows a motorbike on the road. The problem here is occlusion, as the vegetation often occludes the bike, and the speed of the bike, and the moving camera, which also makes it harder to track.

IV. RESULTS AND ANALYSIS

A. Color Based Model

The Color based model was evaluated on the sphere and car1 video sequences, with 16 bins with varying number of candidates and channels : grayscale (K), Hue (H), Saturation (S), Red (R), Green (G) and Blue (B). The number of candidates were incremented until convergence of the best performances.

1) Sphere: The sphere video sequence was evaluated first. The camera is static. This sequence has several challenges; the image is dark with a majority of blue tones, and the tracked object blends with the static background (but not with the the person holding the ball). It is probably part of a chroma key experiment with a green ball.

First of all, the experiment revealed that the Red, Green and Blue models have a similar average processing time. Also, the Hue and Saturation models have a similar average processing time. The fastest models are the Red, Green and Blue models, followed by the Grayscale model and then the Hue and Saturation models. These processing times can be observed in Table II.

The performance of each channel with several number of candidates are in in Table I. This shows that the best performance was obtained with the green channel, followed by the blue and grayscale channels. The best average performance goes up to 59.87 percent, with 3600 candidates.However, the average performance converges with the increasing number of candidates, but the average processing time keep increasing. Therefore, the best performance results in a tradeoff between accuracy and time. With 3600 candidates, the average performance of the Green model is 59.87 percent for an average processing time of 45.85 ms/frame. With 1660 candidates, the processing time is reduced by 2 and the average performance is still very close to the previous one, 59.24 percent. Even with 1225 candidates, the time is reduced by 3 and the average performance is 58.98 percent. If time is an important constraint, it is reasonable to chose the color model with the green channel, with 1225 candidates.

It is also interesting to note the Hue and Saturation based models performed really poorly and slowly.

2) Car1: The car1 video sequence was then evaluated. This video sequence present different and more challenges than the previous one. In this video sequence, the camera is attached to a car. Not only it follows to car general movement, it is also very unstable and moves in random directions due to the vibrations and the sudden movements of the car. The frames are dark with a majority of blue tones, and the tracked object's color, a white car, is really similar to the road. The object also has similar features as other objects on the image, i.e. read lights. There is also some obstruction towards the end of the sequences with some raindrops and more importantly the movement of the windshield wiper.

Here again, the Red, Green and Blue models have a similar average processing time. Also, the Hue and

Nb. cand	K	H	S
64	0.385724	0.234884	0.230319
91	0.472581	0.232273	0.230296
100	0.460239	0.252066	0.233094
784			
1225			
1600	0.56249	0.120135	0.289381
2500	0.561212	0.116155	0.297713
3600			
Nb. cand	R	G	B
64	0.378739	0.354418	0.293882
91	0.411935	0.400214	0.308495
100	0.434001	0.42078	0.440273
784	0.498248	0.582242	0.562588
1225	0.497466	0.589797	0.563777
1600	0.501037	0.592366	0.56758
2500	0.505557	0.597247	0.569488
3600	0.508193	0.598679	0.569475

TABLE I

AVERAGE TRACKING PERFORMANCE IN % WITH SPHERE

Nb. cand	K	H	S
64	1.03206	2.79067	2.40398
91	1.27418	3.18978	3.19024
100	1.67469	3.9779	3.95677
784			
1225			
1600	23.6886	56.535	59.4147
2500	36.7121	82.7872	95.9793
3600			
Nb. cand	R	G	B
64	0.808137	0.816326	0.798777
91	1.06207	1.04689	1.00426
100	1.32293	1.38672	1.39729
784	10.1963	10.1252	10.1602
1225	15.1821	15.2419	15.3511
1600	20.3006	20.1526	20.1347
2500	33.1505	33.1468	34.3786
3600	47.2891	45.8497	46.4926

TABLE II

AVERAGE PROCESSING TIME IN MS/FRAME WITH SPHERE

Saturation models have a similar average processing time. The fastest models are the Red, Green and Blue models, followed by the Grayscale model and then the Hue and Saturation models. These processing times can be observed in Table IV.

The performance of each channel with several number of candidates are in in Table III. Less different number of candidates were selected. Here, the best performances were obtained with the Red channel based model followed by the Saturation based model. The Red channel based model yields results up 63.76 percent, with 1225 candidates. The average processing time is 16.1 ms/frame. More candidates is detrimental not only to the processing time, but also to the performance. It is interesting to mention that the Saturation based model also yields good results, but never as good as the Red channel based model, and for double the processing time. The Hue based model yields extremely low results, the Blue channel based model also yields a low performance, and the grayscale model yields unstable results.

Nb. cand	K	H	S	R	G	B
64	0.384515	0.132112	0.431039	0.422868	0.396905	0.190377
100	0.428831	0.0477936	0.455079	0.467044	0.458033	0.223756
225	0.522291	0.036575	0.528132	0.528475	0.439143	0.250564
784	0.272537		0.598005	0.622466		
1225	0.586113		0.614992	0.637645		
1600				0.633521		

TABLE III

AVERAGE TRACKING PERFORMANCE IN % WITH CAR1

Nb. cand	K	H	S	R	G	B
64	1.21729	2.66522	2.52651	0.860072	0.87989	0.843544
100	1.56466	4.21295	3.98333	1.30885	1.28821	1.35656
225	3.66882	9.80414	9.60069	3.0901	3.13903	3.13924
784	13.0421		34.2178	11.7838		
1225	20.9965		52.2226	16.0968		
1600				20.4995		

TABLE IV

AVERAGE PROCESSING TIME IN MS/FRAME WITH CAR1

B. Gradient Based Model

This section presents the results acquired with the Gradient model.

1) Ball2: The main challenges of this video sequence are: small size of the tracked object, speed of the object and the fusion with the background at certain stages (Figure **??**). The results for this video can be seen in Table V.

Fig. 1. Main challenge in ball2.mp4: interaction with background

Because the ball flied relatively fast, using low number of candidates, meaning a smaller search area, lost track of the object almost immediately, since it traveled more than the neighbourhood between two frames. To avoid this the number of candidates had to be increased above 100.

Since the object is small, even a slight variation in its background caused the tracker to lose the object. That is why using too detailed or too sparse descriptors (number of bins) resulted in a slightly worse performance. It can be seen that using 12 bins per histogram has better performance in all but the last case, with 16 bins. While 16 bins with 625 candidates gave the overall best result, using the same number of bins with less candidates had under-performed all other cases with a smaller number of bins. For this it was not

Nr. Bins	Nr. Candidates	Accuracy (%)	Time (ms/frame)
6	81	5.8	27.24
	100	6.5	30.55
	225	12.6	58.77
	625	**27.1**	153.29
9	81	5.7	29.02
	100	6.6	29.17
	225	23.3	67.39
	625	**28.1**	161.93
12	81	5.7	27.20
	100	6.5	23.94
	225	**19.6**	28.07
	625	13.0	172.77
16	81	5.7	23.69
	100	6.5	26.24
	225	6.1	69.90
	625	**29.6**	188.30

TABLE V
RESULTS ON VIDEO BALL2

considered as the best parameter for number of bins. With 9 bins almost the same results were achieved with similarly 625 candidates and the performance time was slightly better too.

As for the computation performance, it was mostly determined by the number of candidates; the more candidates were used, the more time the calculations took, while the number of bins only slightly affected these times.

So in conclusion, a low number of bins worked better for generalizing the object disregarding some of the background and a high number of tested candidates was necessary to be able to follow the ball with high speed. But even with these parameters the best result was 29.6% because once the ball reached the net of the gate, the tracker lost the object and never regained it.

2) Basketball: In this video, a basketball player was tracked during a game. The main problem are the occlusion of the person, the fact that his stance changes (wider area, lower height, legs far while running), his orientation also changes and when somebody gets close the tracker might start tracking that other person. These can be seen in Figure 2.

There are several challenges in this video. Right at the beginning, there are several occlusions with other players. Around frame 80, the player starts running, meaning that he moves faster. His leg movements also make the object wider, change its appearance and differ more from the model. Results below 10% could not handle this case. At frame 280, because of the orientation change, the similarity was too small and at one point the background got higher similarity, losing the object and never reaching it back. Results around 40% percent accuracy could not handle this case. At the very end, he comes close to another player and the tracker starts following the other person. None of the models could handle this case.

When it comes to the performance, increasing the number of candidates greatly affected the computation time and after a point, it also decreased the accuracy. Increasing the number of bins above 9 also negatively impacted the performance,

Fig. 2. Challenges in video basketball.mp4: Occlusion (top-left), Orientation (top-right), Stance (bottom-left) and Similarity (bottom-right)

going above 12 the results were not acceptable at all. Using 100 or 225 candidates had less then one percent gain in accuracy but more than twice the time in computation so the lesser number is recommended.

The best results were achieved with a low number of bins (6 or 9) where using 81-225 bins had only slight affect on the accuracy, but nevertheless, using 225 candidates had the best results. Using more bins meant using more complicated models which had bad results since the object changed a lot so a more general model was needed.

Using 16 bins the results were not consistent. Overall, the results are worse than for the other parameter combinations, but for two cases (81, 225 candidates) it was a lot worse. This leads to believe that the other two cases (100, 625 candidates) were only good because of coincidence.

Since the tracker uses a fixed size model and in this video the size of the person changed greatly, even though the tracker was mostly following him, the accuracy scores were not that good. The similarity scores can be seen in Figure 3 which show that the tracking was mostly stable except at the beginning where it was a bit behind the object and at the end where it lost it.

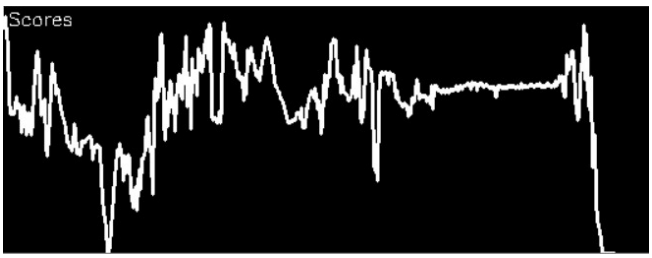

Fig. 3. Similarity scores on basketball.mp4 with 9 bins and 225 candidates

In conclusion, the original parameters (9 bins, 100 candidates) were the best if we take into account the performance too, otherwise slightly increasing the number of candidates

Nr. Bins	Nr. Candidates	Accuracy (%)	Time (ms/frame)
6	81	57.5	19.53
	100	59.1	23.03
	225	**59.7**	51.18
	625	44.3	145.55
9	81	59.6	19.33
	100	60.0	23.53
	225	**60.5**	52.34
	625	44.2	144.48
12	81	25.4	19.78
	100	**56.34**	23.77
	225	37.6	53.98
	625	47.0	154.28
16	81	9.0	19.90
	100	26.28	24.71
	225	8.5	53.71
	625	**42.9**	155.5

TABLE VI
RESULTS ON VIDEO BASKETBALL

can gain half a percent but the price is twice the computation time.

C. Fusion Model

In this section the fusion model's performance is analyzed. Since the previous sections did not result in an overall best model a few combination of the best ones was tested. The number of bins was fixed, 16 for the color model and 9 for the gradient model.

Note: Fusion model has an overall computational performance loss of around 10%. This is most likely due to the implementation, since the methods are not reimplemented they are rather forwarded to the two stored models. These extra steps, and the fact that some data is stored unnecessarily adds the overhead to the method. But this does not affect the overall conclusion of this section.

1) Bag: This video showed a plastic bag being blown in the air by the wind. There were two main challenges with this sequence. First the bag gets deformed by a great amount. Secondly, its distance to the camera changes, for a large portion of the video it is only a quarter of its original size.

The results of different models and their combinations can be seen in Table VII. As it can be seen in the table, the gradient model performed poorly, but the color and fusion model performed well, with only slight differences so these will not be discussed in detail.

The gradient model's poor performance was probably due to the shape and size difference of the object for which it was not robust to. The color model on the other hand performed a lot better. Both channels (G and R) performed similarly around 38%. Even though, it was able to follow the object most of the time, the size difference between the estimation and ground truth decreased this accuracy number. The good performance is mostly due to the fact that the object has a distinguishable color which set it aside from the background. The similarity scores can be seen in Figure 5. It shows that even though the object was tracked most of the time, the similarities were low because of the size and shape difference and only got high again at the end when it resembles mostly the original shape and size.

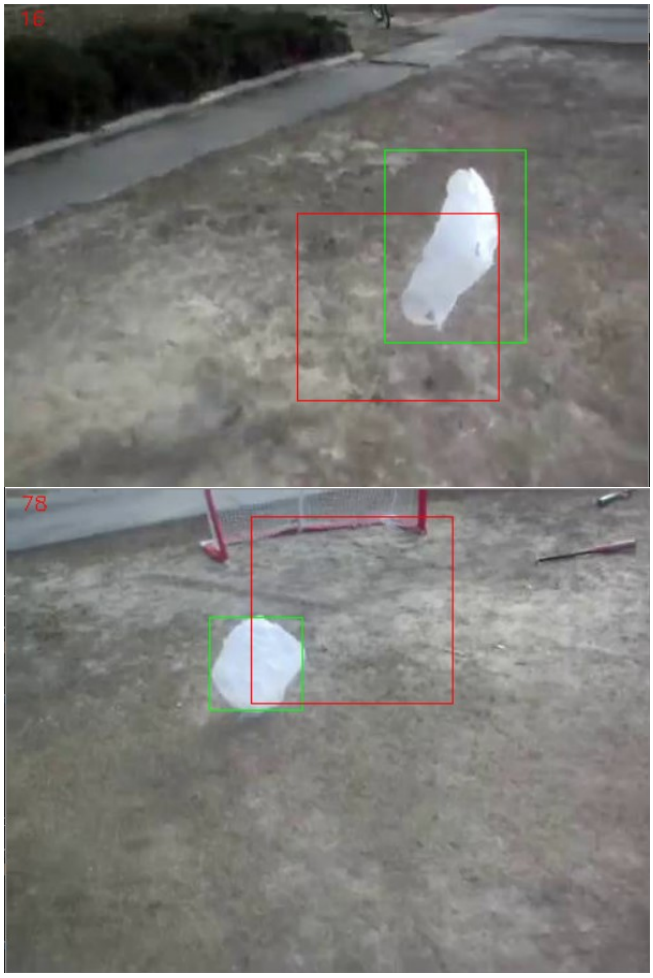

Fig. 4. Challenges of bag.mp4: Deformation (top) and size difference (bottom)

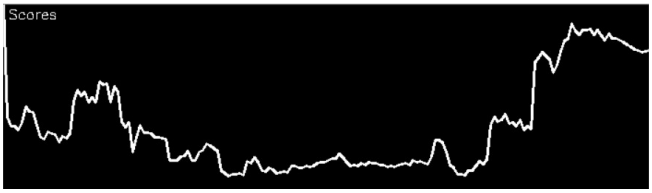

Fig. 5. Similarity scores on bag.mp4 with red channel and 225 candidates

As for the computation performance, color model is faster by one magnitude. When combined with the gradient model in only contributes around 15% to the computational need. Adding gradient in the fusion model only achieves $\sim$1% accuracy gain while increasing the computational need by $\sim$8 times.

In conclusion, using the color model only is a better choice, the fusion model has almost no accuracy gain while it greatly increases the processing power needed.

2) Ball: This video showed a red ball with white pattern being kicked back and forth between two people. The main challenges here were that at the moment of kicking it, the ball moves suddenly and rapidly and that the ball got closer

Model	Channel	Nr. Cand.	Acc. (%)	Time (ms/frame)
Color	G	225	37.2	6.9
		784	37.1	21.5
	R	225	38.7	7.3
		784	38.7	22.7
Gradient	-	225	5.9	52.6
		784	7.5	161.3
Fusion	G	225	35.5	61.8
		784	37.7	233.1
	R	225	38.9	65.6
		784	**39.7**	216.0

TABLE VII
RESULTS ON VIDEO BAG

to the camera multiple times, meaning its size varied too.

The results are comparable to the previous video's results, but had an overall better performance. This was probably due to the fact that the ball is easily distinguishable from the background given its strong colors. The models performed mostly consistently so only the outliers and a general overview will be given. Detailed results can be seen in Table VIII

The color model performed well, thanks to the strong colors of the object. Only using the green channel and few number of candidates was not effective. This was due to the sudden movement of the ball mentioned earlier, where the ball moved too much for the tracker to be able to find in the next frame. This can be seen in Figure 6 Interestingly, this could have proven a problem for the red channel too with this few number, but given that the ball is red, it seems the tracker was just able to match at that moment, and since it did not lose the object there, it was able to follow it for the rest of the video. Another reason why this moment is critical is because the sudden movement is combined with the fact that the ball is closer to the camera, meaning it was bigger than the model.

The gradient model performed poorly again on this sequence too. This was probably due to the ball's pattern. The high contrast between the pattern and base color probably defined the gradient model, but the problem was that this pattern changed a lot, also the outline of the ball could not be tracked properly since its size varied too.

The fusion model performed consistently, but as before only gained ∼1-2% accuracy over the color model but got significantly slower because of the gradient component. The only thing worth mentioning here is the combination of green channel with few number of candidates. Alone this color model performed poorly (20.5%) but when combined with the gradient model (which also performed poorly on its own: 15.1%) this fusion model was on par with the other combinations. This was most likely because thanks to the gradient model the tracker did not loose the object in that critical moment mentioned above and beyond that it was able to keep track of the object.

Because of the different size of the object and the fixed model size, the accuracies are not that high even though the tracker was almost perfectly following the ball. It can be seen in Figure 7 that the tracker was consistently getting

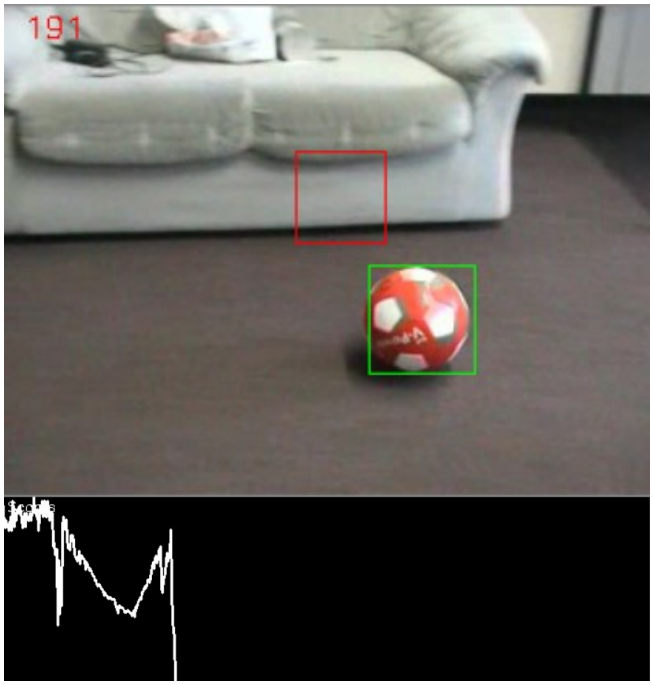

Fig. 6. Color model with green channel and 225 candidates. This is the moment right after the ball got kicked, the tracker lost the object and never found it again as it can be seen it the scores too.

high scores, meaning it was stably following the ball. The two visible decreases in similarity scores were due to size difference of the object and the model.

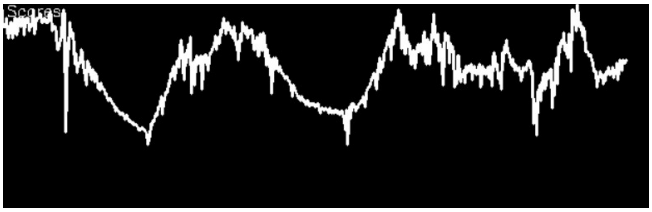

Fig. 7. Similarity scores on ball.mp4 with green channel and 784 candidates

In conclusion, generally fusion model does not offer a significant accuracy gain that would justify the more than 20 times higher computational cost. It is only recommended when a lower performance color model has to be improved.

3) Road: This video showed a motorbike speeding through a curvy road. The main challenges here were the occlusion cause by the trees, a slight change in the size of the object and the orientation change of the object. These can be seen in Figure 8.

All the color models performed similarly, except the green channel with higher number of candidates. The others performed relatively good but they all had the same problem. At the occlusion at frame 110 they loose track of the object but later regain it around frame 160. This can be seen in the similarity scores too in Figure 9. Also the color models seemed to be less stable, jumping around the object rather then keeping fixed to it. These two facts result in the lower

Model	Channel	Nr. Cand.	Acc. (%)	Time (ms/frame)
Color	G	225	20.5	2.5
		784	66.8	7.2
	R	225	65.6	2.4
		784	66.2	7.6
Gradient	-	225	15.1	50.1
		784	9.5	166.0
Fusion	G	225	67.3	55.9
		784	**68.1**	203.7
	R	225	67.0	59.7
		784	67.1	205.7

TABLE VIII
RESULTS ON VIDEO BALL

Fig. 8. Challenges in road.mp4: Occlusion (top) and orientation change (bottom)

accuracy scores when compared to the gradient model.

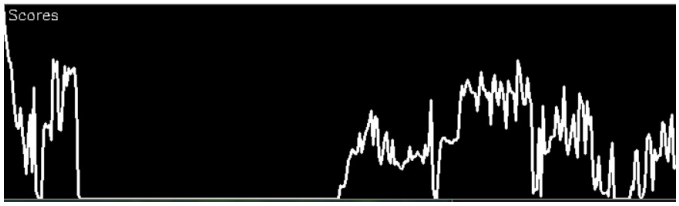

Fig. 9. Similarity scores on road.mp4 with red channel and 784 candidates

With the exception of the fusion model with green channel, all models performed worse with more number of candidates rather than fewer. This was probably because the model got too far during the occluded parts and thus it was not able to regain track afterwards. This can be seen in Figure 10.

The case of the fusion model with green channel is worth mentioning since separately both performed poorly (6.7% and 7.3%) but together they improved significantly to 24.2%, but still worse then some individual models.

Fig. 10. Because of the occlusion the model got far from the object so after this point it will not be able to find it again

The best accuracy score was achieved with the gradient model and low number of candidates. This model proved more stable, did not jump around the object and also did not loose the object after the occlusion, it almost immediately regained it. The only thing this model could not handle was the orientation change. At the very end, around frame 340 the bike started to take a turn as a result it was more of a top view from it, which was too different for this model and it completely lost track of it. This can be seen in the score numbers too in Figure 11. Using more number of candidates the model became more unstable, at the second, longer occlusion the estimate got too far from the object and never regained it.

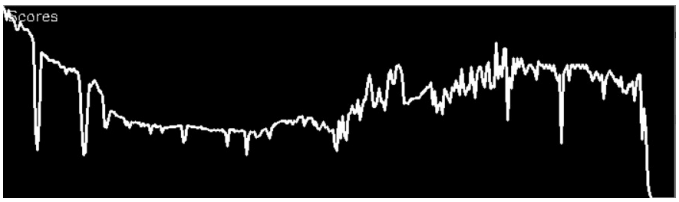

Fig. 11. Because of the occlusion the model got far from the object so after this point it will not be able to find it again

As mentioned before the fusion model with green channel gained in accuracy compared to the individual parts but all other fusion models performed worse than their individual parts. The highest loss was with green channel and low number of candidates, individually both models were among the best (28.7% and 44.3%) but combined it was one of the worst (7.8%). In this case they probably only increased their weaknesses causing them to loose the object at the first occluded part and never finding it again.

In conclusion, almost all models performed better with low number of candidates. The color models performed

Model	Channel	Nr. Cand.	Acc. (%)	Time (ms/frame)
Color	G	225	28.7	1.9
		784	6.7	6.3
	R	225	31.6	2.1
		784	20.2	6.9
Gradient	-	225	**44.3**	51.3
		784	7.3	182.1
Fusion	G	225	7.8	51,5
		784	24.2	187.4
	R	225	30.7	55.3
		784	6.5	190.3

TABLE IX
RESULTS ON VIDEO ROAD

acceptably but were unstable, while the gradient model performed the best just could not handle the orientation change. The fusion model did not prove better at all, so again its use is not recommended.

V. CONCLUSION

In conclusion, histogram-based models can be implemented in different manners. Color-based or gradient-based models were experimented with in this lab, as well as a combination of both. The choice of the model as well as the tuning of their parameters depend on the video sequence. There is also a tradeoff between accuracy and processing time. The fusion model, combining both color-based and gradient-based features have proven to show worse results than only using one. However, this statement would need further investigation with a flexible bounding box size.

VI. TIME LOG

Below the amount of time spent on each aspect of the project is detailed. The times contain both participant's time summed up.

1) Code implementation: : 6 hours, designing the code structure and, implementing it

2) Evaluation of color based model: : 3 hours

3) Evaluation of gradient based model: : 3 hours

4) Evaluation of fusion model: : 3 hours

5) Report: : 6 hours, assembling and writing

REFERENCES

[1] P. Fieguth and D. Terzopoulos, "Color-based tracking of heads and other mobile objects at video frame rates," Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, USA, 1997, pp. 21-27

[2] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231), Santa Barbara, CA, 1998, pp. 232-237