# Lab3: Single Object Detection and Tracking Using Kalman Filer

Tamás BUKITS and Balázs Márk BÓDIS

## I. INTRODUCTION

In this assignment we were dealing with the problem of single object detection and tracking. The project began with 2 individual tasks which are the following. Task "A" was the implementation of a single object detection method based on foreground-background segmentation using the Gaussian Mixture Model. Task "B" was the implantation of the Kálmán Filter algorithm using the OpenCV library. The tasks A was done by Balázs Márk Bódis and task B was done by Tamás Bukits.

This report will focus on the collaborative part of the assignment, as the individual work we did was already documented. As the first part of the collaborative tasks we conducted experiments to analyze the effect of the Kálmán components on "toy data". Then we try to visually evaluate our method for real data and optimize our results based on visual clues. Finally we will wrap our report with the reflection on our work.

## II. METHODS

In this section we provide a compact list of methods used in this assignment. We do not go into detail here as we already discussed the methods in the individual tasks.

Methods used in this assignment:

### A. Single object detection:

1) Mixture of Gaussians Model - MOG2
2) Morphological Operations on binary mask
3) BLOb detection - Grassfire algorithm
4) Feature based BLOb selection
5) Location extraction from selected BLOb

### B. Object tracking:

- Kálmán Filter - Constant Velocity Model
- Kálmán Filter - Constant Acceleration Model

## III. DATA

*Task 3.1*

The data used in this task was a single video file of a ball rolling along the floor with a box in the frame that occludes the ball for a brief section of the video. The camera is stationary, thus segmentation is possible with MOG. There is a total of 45 frames.

*Task 3.2*

In this taks 4 similar sequences are introduced.

- "video2.mp4"
  The ball rolls from left to right. Expected challenges: ball moves fast compared to its size and framerate. It becomes blurred and not round in the individual frames.
- "video3.mp4"
  The ball hits the wall and changes direction. Expected challenges: ball moves fast, it changes direction and speed (shape changes as a result). Camera is not completely static, background wall moves a bit from the impact of the ball.
- "video5.mp4"
  The ball drops from the top edge of the frame. It bounces a 5+ times and then comes to a stop. Expected challenges: The speed change is more drastic. Frame rate can become problematic for detection. Camera is not static, MOG will produce exceeding amounts of noise.
- "video6.mp4"
  The ball rolls from left to right. It gets occluded by a box for 7 frames. Expected challenges: No measurement due to occlusion. Ball stops quickly at the end of its track.

*Task 3.3*

In this task 4 real world sequences are introduced.

- "boats_6950_7900_clip.mp4"
  A boat is sailing in the foreground while pedestrians and cars move in the background. Expected challenges: Moving background (water and other objects). Due to the use of segmentation for detection the main objects size will change as it will "collide" with other objects. The boats shape changes as it turns.
- "pedestrians_800_1025_clip.mp4"
  Single individual walks through static scene. Expected challenges: shadows can cause a problem in this scene. Due to the movement of the pedestrian its BLOb center will not exactly follow their movement.
- "abandonedBox_600_1000_clip.mp4"
  A cyclist moves through a static scene. Expected challenges: Low resolution and noise due to the analog camera system.
- "streetCornerAtNight_0_100_clip.mp4"
  A fast car moves through empty intersection. Expected challenges: Object is blurred due to its speed compared to the relatively low frame rate.

## IV. Implementation and Evaluation

*Task 3.1*

We combined our codes to create a single detector-tracking algorithm. The block diagram of the combined algorithm can be seen in *Figure 1.*[1]. After merging our algorithms

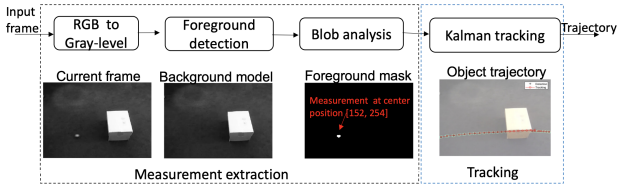


Fig. 1. Block diagram of the combined algorithm.[1]

we have the following structure of our code (python like pseudocode):

```
for frame in frames:
    mask = MOG(rgb2gray(frame))
    mask = morph_open(mask)
    blobs = extract_blobs(mask)
    coordinates = extract_biggest_blob(blobs, ...
    ... size_requirements)
    kalman.predict()
    state = kalman.update(coordinates)
```

*Results:* Here we did not conduct parameter search, the results of the algorithm can be seen in *Figure 2.*. Here we would like to introduce our visual representations: white cross = measurement, red bounding box: prediction using measurement, green bbox: prediction with no measurement used. The following parameters were used, these will serve as a starting point for any further experiments:

```
#parameters for MOG2
learning rate = 0.001
history = 10
varThreshold = 20
isShadow = true #detect_shadows?
#parameters for creating mask
int connectivity = 8;
opening size = (3, 3) #for morpological operation
MorphShapes = MORPH_RECT #using rectangular kernel
#parameters for selecting BLOb
int min width = 9;
int min height = 9;
#parameters for Kalman filter
kalman_mode=ConstantAcceleration
transition_el = [1.0,0.5,1.0,1.0,0.5,1.0]
noise_el = [25.0,10.0,1.0,25.0,10.0,1.0]
uncertainty_elements = 25.0
uncertatinty_scaling = 10
```

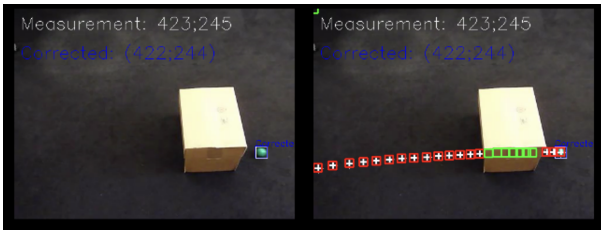


Fig. 2. Task 3.1 output

*Task 3.2*

In this task we first tried to tune the MOG2 modules parameters along with other steps of the object detection algorithm over the 4 sequences.

With the suggested parameters there were very weak detections in sequence ”video5.mp4”. This was mainly due to the high amount of noise in the background on the white wall. Due to the noise and the high varThreshold for MOG2 the balls size did not reach the suggested minimum size requirements: 50 pixels. To fix this, we first decreased the size required. But as we can see from *Figure 3.* the ball is mostly detected as shadow in the top parts of the image. – As turning off shadow detection would hurt our detection in other sequences (as can be seen in *Figure 4.*) we opted to empirically test lower thresholds and change the required size parameters. We landed on:

```
varThreshold = 10
min_width = 20
min_height = 10
```

To combat the noise we created with lowering the threshold (can be seen in *Figure 5.*), we increased the size of the opening morphological operation. We set it to Size (3,15), since the ball was taller due to its movement in this, most problematic sequence. After the opening size increase the output can be seen in *Figure 6.* To counter this effect we introduced a closing morphological operation. At this point we had low noise levels, and we had a clear BLOb for most images where the ball was visible, so we could move on to tracking using the Kálmán filter. For all sequences we first tried using the constant acceleration model and if that did not work work for some reason we tried the constant velocity model.

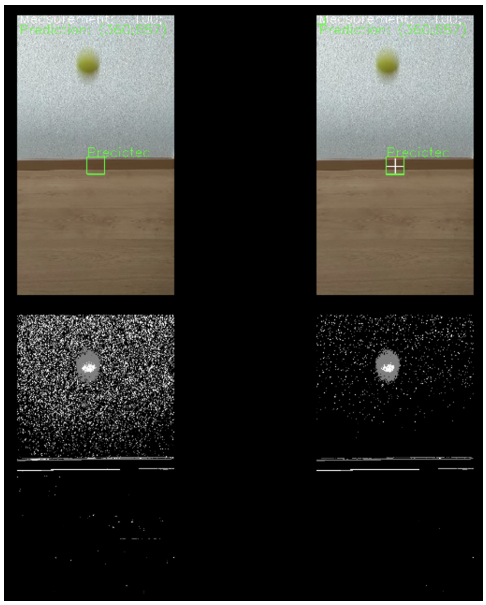

Fig. 3. video5.mp4 with varThreshold = 16, the ball is mostly detected as shadow

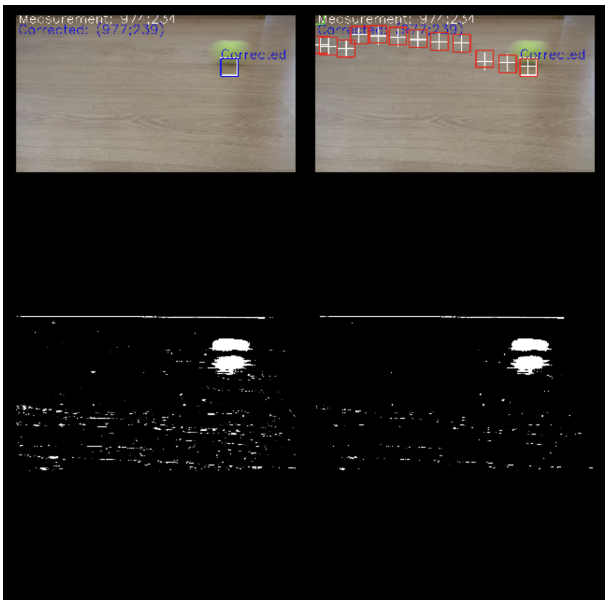

Fig. 4. video2.mp4 with no shadow detection, the ball is often confused with its shadow on the flooring

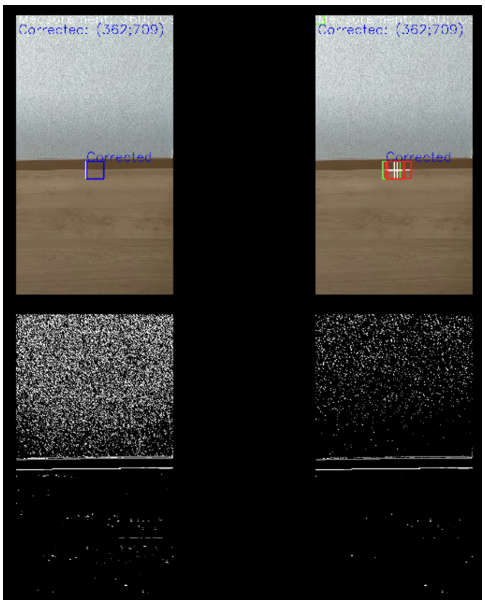

Fig. 5. video5.mp4 with inadequate morphological operations, the background noise creates larger BLObs

*video2.mp4:* In this sequence our we used the starting parameters for Kálmán filter which can be seen in Section *Task 3.1*. Our results can be seen in *Figure 7.* We think our results are optimal.

*video3.mp4:* In this sequence our we used the starting parameters for Kálmán filter which can be seen in Section *Task 3.1*. Our results can be seen in *Figure 8.* We think our results are adequate, but there is a problem with the constant acceleration model, namely at the end of the sequence the ball slows down to an almost stop. This leads to the Kálmán filter predicting that it will come back into the frame as the acceleration is constant and the velocity vectors direction

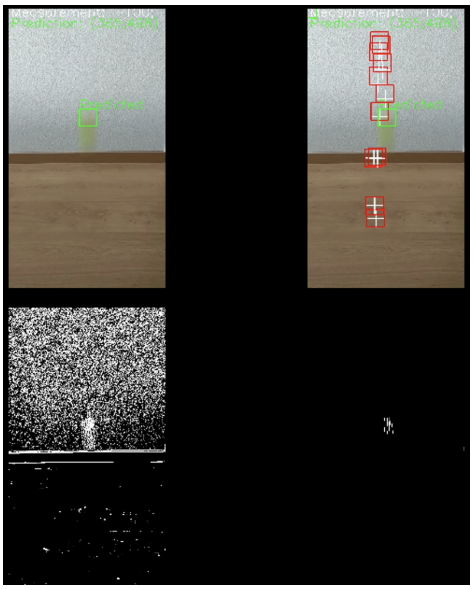

Fig. 6. video5.mp4 with opening morphological operation, the actual BLOb of the ball is also greatly degraded with the noise

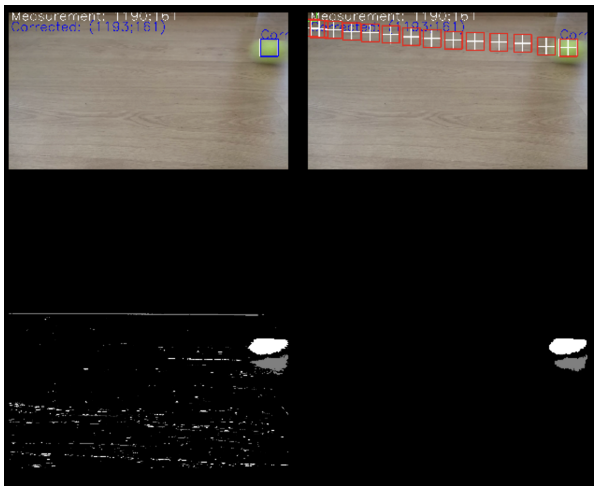

Fig. 7. video2.mp4, with constant acceleration model, no parameter changes

changes. To avoid this here we used the constant velocity model, this can be seen in *Figure 9.*. The overall movement of the ball is much more realistic in this case.

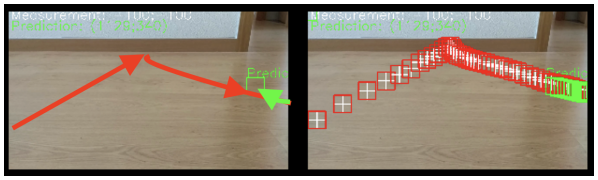

Fig. 8. video3.mp4, with constant acceleration model,
red: trajectory when measured,
green: trajectory when predicted

*video5.mp4:* In this sequence our we used the starting parameters for Kálmán filter which can be seen in Section *Task 3.1*. Our results can be seen in *Figure 10.* As we can see

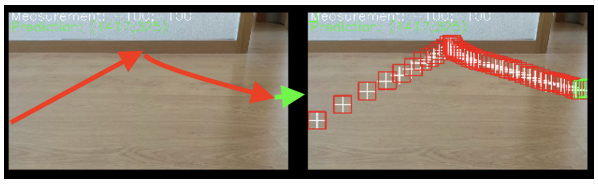

Fig. 9. video3.mp4, with constant velocity model,
red: trajectory when measured,
green: trajectory when predicted

(our predictions are close to the correct place. When running the constant velocity model, we can see that some of the predictions are out of place when the ball is not measured (*Figure 11.*). We conducted one more experiment, raising the uncertainty scaling for the constant acceleration model. We found that this was slightly improving our results (as seen in *Figure 12,* and we settled on 10e5.

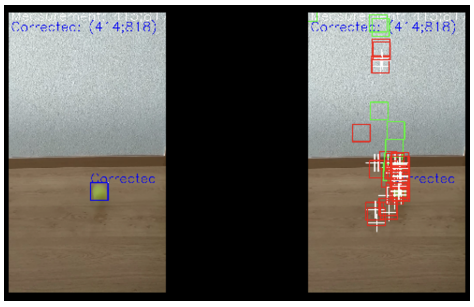

Fig. 10. video5.mp4, with constant acceleration model

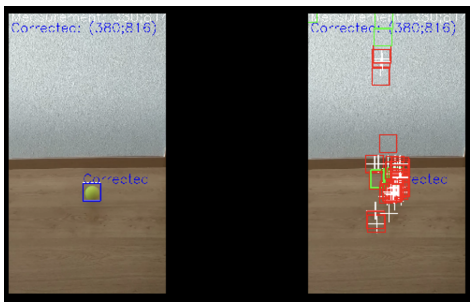

Fig. 11. video5.mp4, with constant velocity model

*video6.mp4:* In this sequence our we used the starting parameters for Kálmán filter which can be seen in Section *Task 3.1.* Our results can be seen in *Figure 13.* We increased the uncertainty scaling to 10e5, the same as for the previous video, which was an improvement (*Figure 14.*). Finally we tried increasing the uncertainty elements, we settled on 125. The results of the experiment can be seen in *Figure 15.*

*Task 3.3*

We used the suggested parameters for EOM and we used uncertainty elements = 125 and uncertainty scaling = 10e5 as a starting point.

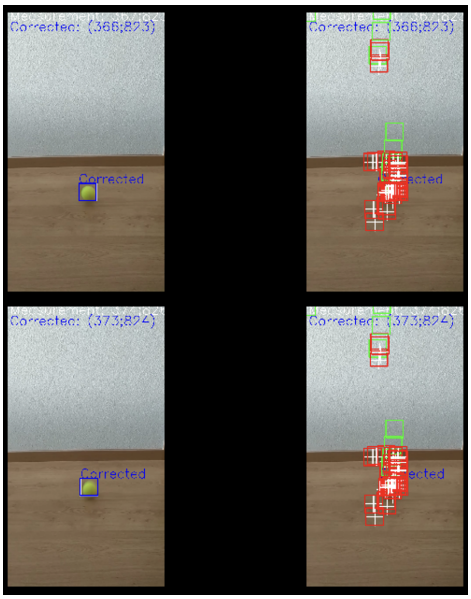

Fig. 12. video5.mp4, with constant acceleration model,
uncertainty scaling = 10e3 - TOP
uncertainty scaling = 10e5 - BOTTOM

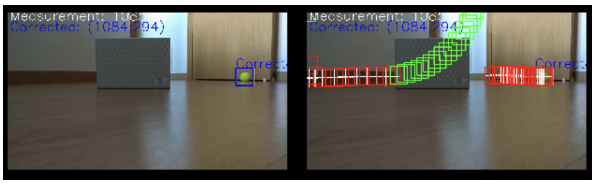

Fig. 13. video6.mp4, with constant acceleration model

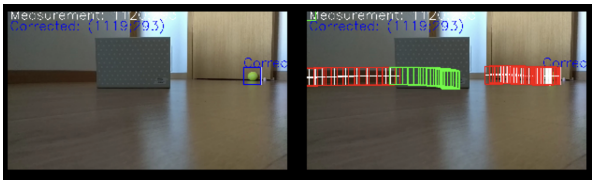

Fig. 14. video6.mp4, with constant acceleration model, higher uncertainty scaling

*pedestrians_800_1025_clip.mp4:* Before we could tackle the tracking issue we needed to alter a few parameters. Here we started with the suggested parameters for EOM. We first reduced the required size for the width of the object to 20, as the pedestrian was not that wide. Second, we noticed that many pixels of the foreground were falsely detected as shadows (can be seen in *Figure 16.*), so we turned off shadow detection. We tried both constant velocity and acceleration models. In this case we decided that constant velocity is better, as the pedestrian moves in a relatively steady pace overall, but has a lot of small trajectory diversions along the way. Our thought was that these small movements can impact the acceleration model greatly while the velocity model is more robust against them. The results can be seen in *Figure 17.* Due to the uneven pace of the walking both approaches struggle, but overall the constant velocity model works better.

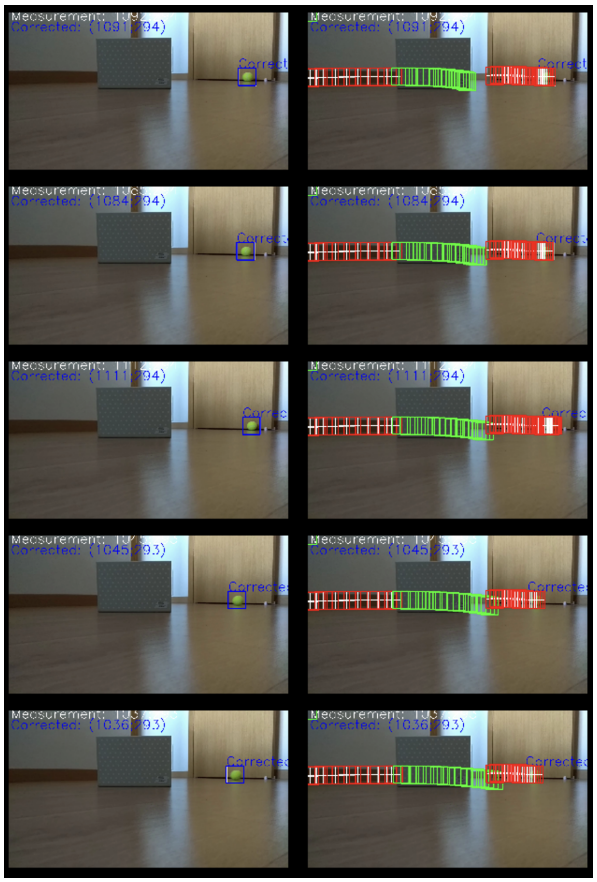

Fig. 15. video6.mp4, with constant acceleration model, higher uncertainty elements: 50, 75, 100, 125, 150

Fig. 16. pedestrians_800_1025_clip.mp4, EOM shadow detection on and off respectively

*abandonedBox_600_1000_clip.mp4:* With the basic setting for the EOM we did not have any detections, since the object is smaller. We set min_width = min_height = 10. We searched for the optimal minimum height as the main problem was that the cyclist and its shadow was confused by the detector. From *Figure 22.* we can see that the optimal value is 20 as it will discard all shadow detections but will have enough detections for the cyclist that the trajectory reconstruction is possible using the constant velocity model. With this we basically made a tradeoff for less measurements, but more reliable ones. Here we are relying on our tracking more. We noticed that the MOG model Incorporated the cyclist into the model while he was staying around. We decreased the

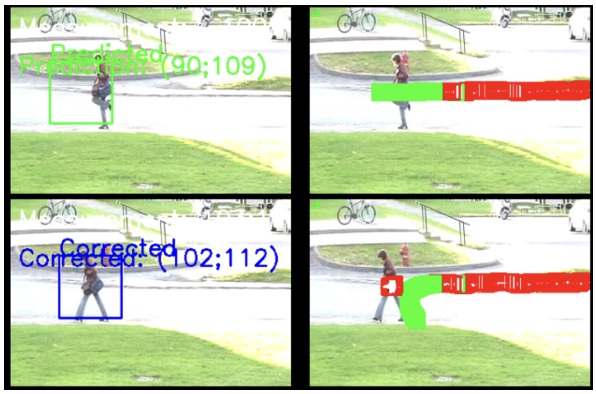


Fig. 17. pedestrians_800_1025_clip.mp4, tracking with constant velocity and constant acceleration models respectively


Fig. 18. pedestrians_800_1025_clip.mp4, tracking with constant velocity and constant acceleration models respectively

learning rate to 0.0001. This solves the issue for the clip, but for longer clips, with more background changes we need to look into this parameter.

*streetCornerAtNight_0_100_clip.mp4:* We used the suggested parameters and the starting parameters for our Kálmán filter. We found our results very similar with the constant acceleration and velocity models. Finally we used the constant velocity model. The output can be seen in *Figure 18.*

*boats_6950_7900_clip.mp4:* The main problem in this sequence is the detection of the boat. We opted to trying to detect the sail, as the center. For this we lowered the required width to 10 pixels. The result of the bkg segmentation can be seen in *Figure 19.* We used the constant velocity model, the reason being that when the boat turns the pixels change fast, and the top and bottom half of the boats sail gets disconnected. First the top half is detected, but then it becomes immeasurable (we did not set the thresholds lower for stability). This results in the behaviour of the constant acceleration model that we already observed in previous sequences that it can change direction and speed very quickly with a single bad measurement (look *Figure 20.*. We decided that in this case, using the constant velocity model (*Figure 21*) is more efficient in terms of parameter tuning time and the performance is adequate as the boats are on a simple trajectory with mostly constant speed. The constant velocity model follows the boat decently even when there are 10+ frame measurements missing.

## V. RUNNING THE CODE

In order to compile, link and run the project, it is only needed run make in the project's src folder and the run the ./main command. We included all parameters used as comments (where we deemed them necessary for convenience).

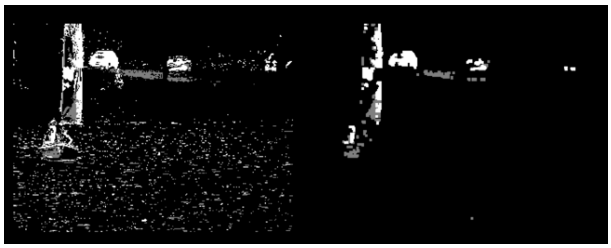

Fig. 19. boats_6950_7900_clip.mp4, EOM with suggested parameters

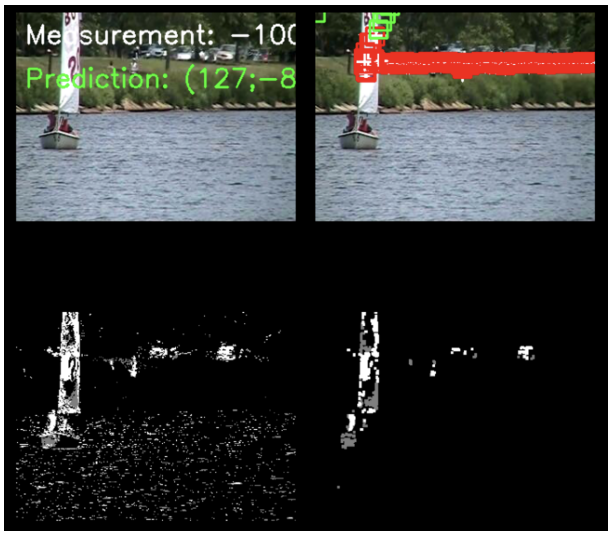


Fig. 20. boats_6950_7900_clip.mp4, Kálmán with constant acceleration

## VI. USAGE

For task 3.1 just run the code. For task 3.2 to change the parameters please alter the kalman.cpp script. We provided the locations in the code where the script can be altered if needed. For task 3.3 please change the parameters of MOG in the main.cpp script. To change between constant acceleration or velocity model please use the isConstantVelocity flag.

To run the scipts for different files please alter the filenames to the first sequence that needs to be run or change the filename in the script. The path for the dataset downloaded from moodle is:

```
/home/avsa/Documents/dataset_lab3/lab3.x/[filename]
```

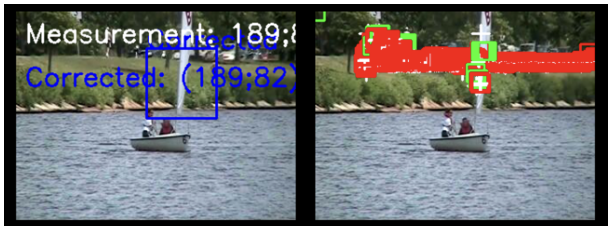


Fig. 21. boats_6950_7900_clip.mp4, Kálmán with constant velocity

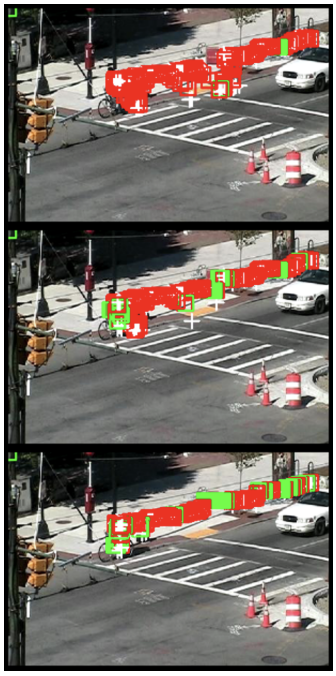

Fig. 22. abandonedBox_600_1000_clip.mp4, tracking with constant velocity, minimum height = 10, 15, 20 respectively

## VII. APPENDIX

### A. Individual Work Reflection

For the measurement extraction the hardest part was finding the correct hyper-parameters. For Kalman-filtering the hardest part was to hard-code the matrices, without making any mistakes in it.

### B. Code Merging Feedback

The code merging was smooth since the 2 parts was following the principals of the clean coding, including code encapsulation and good comments. The code was organized in different functions, classes which made the main file cleaner and easier to understand each others code.

### C. Time Log

- Individual task: 5-6 (per person)
- Data analysis: 1.5 hours
- Code Merging: 1 hour
- Parameter Tuning and analysis: 5 hours
- Report: 4 hours

Total: 17 hours

### D. Pair Programming Reflection

Pair programming technique was useful when we sat together and merged the project from our individual parts. This technique is used in real scenarios as well when software engineers have to combine their differently developed tasks. Parameter tuning might not be done in pair working, instead one person should change the parameters the other person tests the model with this setting.

## REFERENCES

[1] Juan Carlos San Miguel - Juancarlos.Sanmiguel@uam.es Applied Video Sequence Analysis Lab 3 description: "Kalman filtering for object tracking" Collaborative work (part II)