Kalman filtering for object tracking

María Fernanda Herrera and David Savary

I. INTRODUCTION

Object tracking algorithms aim to determine the state of an object by means of sensor measurements. Kalman filter [3] is one of the most representative tracking algorithms and is conceptualized into two phases: "Prediction", which estimates the current state based on previous state and a state transition model, and "Correction", which refines that estimation through a measurement [1], [2], [4].

Kalman filter for object tracking is implemented in this project by considering constant velocity or constant acceleration state transition models, and measurement as the center position of an object obtained from video frame analysis.

II. METHOD AND IMPLEMENTATION

Kalman filtering is implemented into three different files. The main file *Lab3.0AVSA2020.cpp* reads video sequences, initializes and applies Kalman filtering and displays the results. The second and third file, *KalmanTracker.hpp* and *KalmanTracker.cpp*, contains the attributes and methods defined for *KalmanTracker* class, which allows to perform Kalman prediction, measurement recovery and Kalman correction. A more in detail description of the implementation and related theory is described below.

A. INITIALIZATION

KalmanTracker object is created at the beginning of every video sequence and takes as an input parameter the type of state model that should be considered, either constant velocity model or constant acceleration model. During the initialization, two main objects defined by *OpenCV* are created, *createBackgroundSubtractorMOG2()* along with the definition of its parameters and *KalmanFilter* along with the definition of its parameters depending on the state model. Besides, parameters related with main blob extraction are also defined.

B. TRACKING

Tracking is performed within the *track* method of *KalmanTracker* object. This method is called after each frame of the video sequence is recovered and is implemented by calling three main functions described as follows:

C. TRACINGK: PREDICTION

Prediction method, *predict()*, is already implemented within the *KalmanFilter* class and involves computing a priori state estimate and a priori error covariance based on the knowledge of previous state, state transition, previous error covariance and covariance of process noise. Thus, the prediction is performed only after the first measurement is obtained and is passed as the initial state.

D. TRACKING: MEASUREMENT

Measurement extraction is performed by the *makeMeasurement* method which is defined by the following different stages.

1) Foreground Detection: This stage transforms the original frame into the grayscale space and applies Gaussian Mixture based Background/Foreground Segmentation definded as a *createBackgroundSubtractorMOG2()* object according to the learning rate, variance threshold and history defined during the initialization process.

2) Noise Filtering: This stage filters small objects in the Background/Foreground Segmentation by applying morphological opening through the *morphologyEx* function according to a rectangular structuring element of size 3x3.

3) Blob Detection: Blob detection was implemented using the *Grass-Fire* algorithm within the *extractBlobs* method.

4) Final Measurement: The blob with highest pixel area is selected if it is bigger than a size threshold defined during initialization. Finally, the center position of the selected blob is returned as measurement. If no measurement is obtained, the values for the point defining the center are set to 0, 0.

E. TRACKING: CORRECTION

Prediction method, *correct()*, is already implemented within the *KalmanFilter* class and involves computing a posteriori state estimate and a posteriori error covariance based on the knowledge of the observation matrix, a measurement, covariance of measurement noise, and a priori state and error covariance obtained from prediction. Thus, the correction is performed only if a measurement is available.

III. RUNNING THE CODE

To run the implemented code, it is necessary to compile the code in a Linux machine with OpenCV installed by using the Makefile. The *make* command will generate a "*Lab3.0AVSA2020*" executable that can be run by passing as argument the path of the desired videos separated by a space, for example: "*./Lab3.0AVSA2020 path/to/video1.mp4 path/to/video2.mp4*".

IV. DEVELOPMENT OF TASK 3.1

The previously described code was implemented using the "singleball.mp4" video sequence as test sequence. The output obtained for this sequence is shown in figure reffig:task1im0 and is obtained by considering final parameters described in I. As it can be observed, the code returns from left to right the original frame, the frame converted to the gray scale space and foreground segmentation in the first row and the filtered foreground segmentation, Kalman filter per frame and

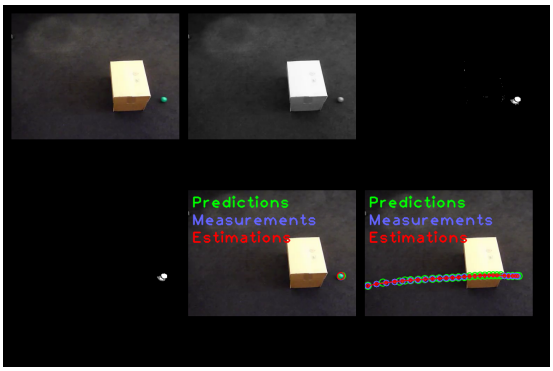

Fig. 1. Output sample for Kalman Filtering Implementation

trajectory obtained by applying Kalman filter in the second row.

V. DEVELOPMENT OF TASK 3.2

A. DATA DESCRIPTION

For this task, 4 different video sequences are used to study the performance of Kalman filtering. All of the sequences show a scene of a tennis ball that follows different trajectories as foreground and a background that contains a dark color region (floor) or/and a light color region (wall). The differences between these sequences are:

1) Background: : *"video1.mp4"* shows a dark color background, which allows to easily perform foreground extraction, while and other sequences a mix of dark and light color background.

2) Trajectories: : *"video2.mp4"* and *"video6.mp4"* show almost straight trajectories, while *"video3.mp4"* and *"video5.mp4"* show abrupt changes of trajectory.

3) Occlusions: : While *"video2.mp4"* and *"video3.mp4"* does not show occlusions, *"video5.mp4"* has pseudo-occlusions given by the similarity of the object with the background and *"video6.mp4"* has real occlusions given by another object.

B. EXPERIMENTAL METHODOLOGY

In order to analyze the effect of each parameter of the Kalman filter, a set of baseline parameters for both constant velocity and constant acceleration models were defined, those parameters were taken from [5] for measurement and process noise covariance, while the error covariance matrix was initialized with 1000 along the diagonal. The value of each non zero value was increased and decreased individually for the measurement covariance matrix, process covariance matrix and error covariance matrix for each scene. The effect of such variations was visually analyzed and a those effects were integrated in order to obtain a better tracking for each sequence. After that, further adaptations on background/foreground segmentation and blob extractions were performed based on the knowledge of the scene characteristics to improve tracking results in some video sequences.

Fig. 2. Backgorund/Foreground segmentation for video2.mp4

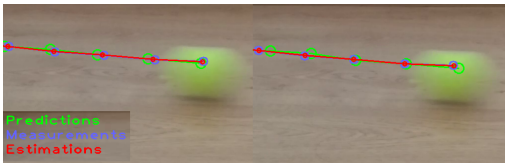

Fig. 3. Ball trajectory for video2.mp4 sequence described by Kalman filer based on constant velocity model (left) and Kalman filer based on constant acceleration model (right) according to baseline parameters in I

C. RESULTS AND ANALYSIS

1) Video2.mp4: This video sequence shows a clear background/foreground segmentation as shown in figure 2. This clear segmentation and the fact that the ball follows a simple trajectory allows to obtain a good tracking easily by means of the measurements and predictions. In fact, even when modifying each parameter of the covariance matrices one by one, similar results were obtained for both Kalman filtering based on velocity and Kalman filtering based based on acceleration. However, Kalman based on acceleration shows a higher consistency between predictions, measurements and estimations since this model may define better the movement behavior of the ball. A comparison of this behavior is shown in figure 3, where the center of the circles that define predictions, measurements and estimations are closer for the acceleration case. It was also observed that modifying the values of the prior covariance may benefit or affect the performance of the filter. When decreasing the second non zero value of the prior covariance, related to the velocity in horizontal direction, the prediction stage gets worst. On the other hand, decreasing the third non zero value, related to the acceleration in horizontal direction, returns better prediction values. Sample images showing this effect are shown in figure 4. The fact that acceleration model returns better results and the fact that those results improve when decreasing the prior covariance of the acceleration in x axis suggest that the ball trajectory is better described by the constant acceleration model rather than the constant velocity model. Kalman filter defined for this sequence as specified in in I, is shown in figure 5.

2) Video3.mp4: The video sequence shows almost no noise from the background/foreground segmentation, while the trajectory of the ball changes abruptly in vertical direction. However, the ball is not occluded during the sequence, suggesting that a small measurement covariance will provide

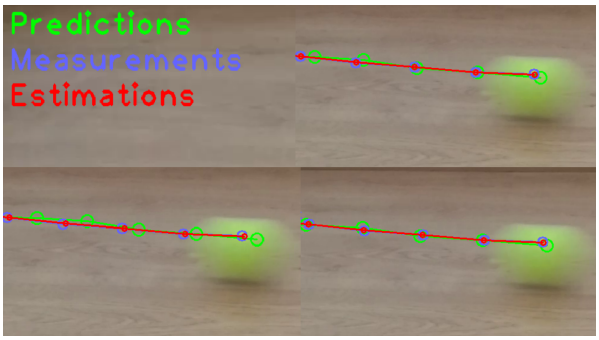

Fig. 4. Ball trajectory for video2.mp4 sequence described by Kalman filer based on constant acceleration model according to baseline parameters (upper-right), decreasing second non zero value from prior error covariance to 100 (bottom-left) and decreasing third non zero value from prior error covariance to 10 (bottom-right).

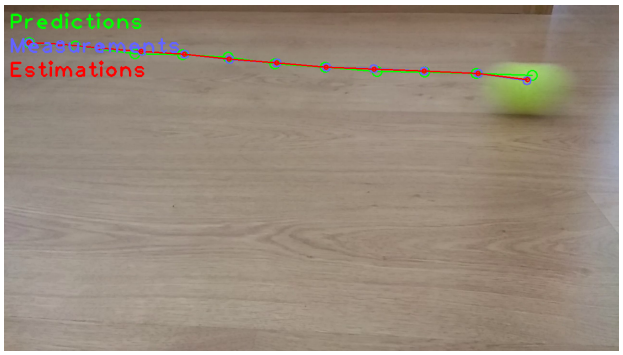

Fig. 5. Ball trajectory for video2.mp4 sequence described by Kalman filer based on constant acceleration model according to final parameters in in I

a good tracking. Figure 6 shows different results obtained for Kalman filtering based on both constant velocity (left column) and constant acceleration (right column) models considering baseline parameters, optimized parameters based on Kalman filer and optimized parameters based on background/foreground segmentation. It can be observed that the constant velocity model with baseline parameters does not match both predictions, with estimations and measurements before the ball hits the wall, while the acceleration model tries to adjust this matching faster with the same baseline parameters. Since the ball overlaps with a blob coming from background noise when hitting the wall, the center of the blob defined by the measurement changes and this introduces a small error in the Kalman filter for both models. The second row of figure 6 shows a correction of this effect by increasing the first value of the measurement covariance matrix, which will decrease the degree of consideration of measurements. However, this also requires to increase the value of prior error covariance related to velocity in horizontal direction, the second value. This allows to integrate measurements at the beginning of the scene, while Kalman filter adapts the state and error covariance, but also allows to decrease the consideration of measurements when the noise error occurs. Another approach to improve the tracking is to increase the learning rate of the Backround Subtractor to adapt faster to illumination changes. After that, values of

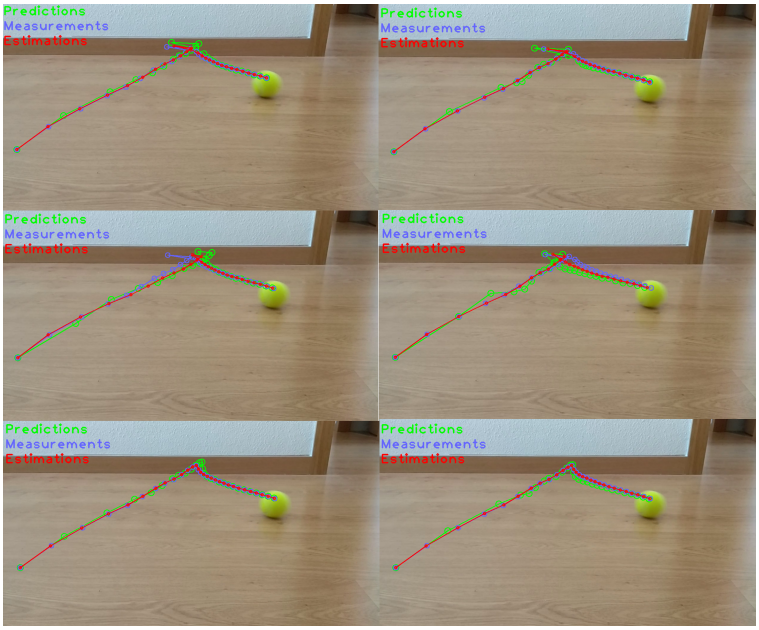

Fig. 6. Ball trajectory for video3.mp4 sequence described by Kalman filer based on constant velocity model (left column) and constant acceleration model (right column). First row describes results obtained from baseline parameters. Second row describes results obtained by increasing measurement covariance to 1000 for the horizontal axis, increasing prior error covariance to 10000 for the second non zero parameter and decreasing process covariance to 0.1 for the acceleration parameters. Third row describes results when a higher learning rate was applied and is Kalman parameters according to final parameters in I

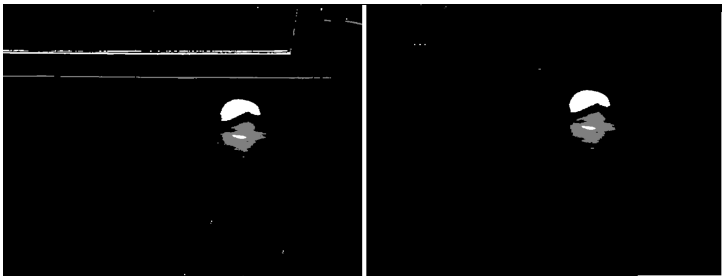

Fig. 7. Comparison of background/foreground segmentation for video3.mp4 sequence with background subtractor learning rate equal to 0.001 (left) and learning rate equal to 0.005 (right).

prior covariance can be set to very small values, as well as measurement covariance value. The last approach is shown in the last row of the same figure 6, while the difference in background/foreground segmentation for a small and large learning rate can be observed in figure 7. Kalman filter is defined for the second sequence and third approach as specified in TABLE 1.

3) Video5.mp4: In order to apply Kalman filter to this video sequence, parameters related to the background/foreground segmentation were modified. First, the learning rate was set to a higher value to deal with background noise from illumination changes. Then, shadow detection was disabled to recover more measurements from foreground detection. If shadow detection would not be disabled, some measurements would not be recovered due

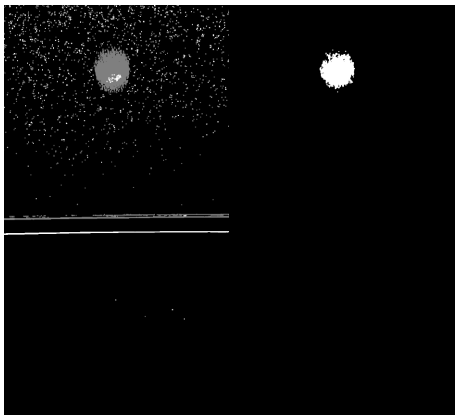

Fig. 8. Comparison of background/foreground segmentation for video5.mp4 sequence with background subtractor learning rate equal to 0.001 (left) and learning rate equal to 0.01 and no shadow detection (right).

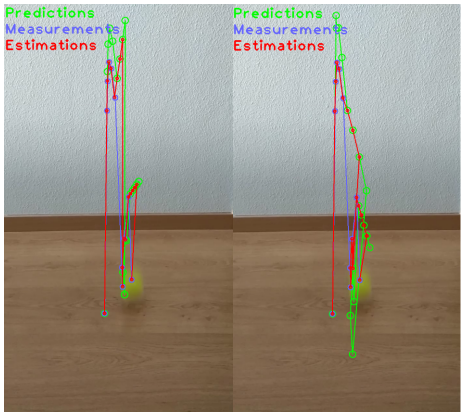

Fig. 9. Ball trajectory for video5.mp4 sequence described by Kalman filer based on constant velocity model (left) and constant acceleration model (right) according to final parameters in I

to the similarity of the ball and light color background in gray scale. The result of previously defined segmentation adjustment can be described in figure 8. Then, Kalman Filter was modified from the baseline by decreasing the values of the measurement covariance matrix for both Kalman based on constant velocity and constant acceleration since there are no occlusions in the scene and the measurements can be trusted. Values for process covariance related to horizontal position, velocity and acceleration were also decreased, since most of the variations happen along the vertical axis. Finally, values for process covariance related to vertical position are highly increased since the ball is bouncing and changes abruptly its vertical direction, as well as decreases velocity. A sample of the results obtained for Kalman filtering based on constant velocity and acceleration is shown in figure 9, where acceleration model describes better the ball trajectory. Kalman filter is defined for this third as specified in TABLE 1.

4) Video6.mp4: The last scene includes occlusions of the ball and fails to extract the ball since it moves away from the camera and its size decreases. Blob detection can be solved by decreasing the size thresholds specified for blob detection

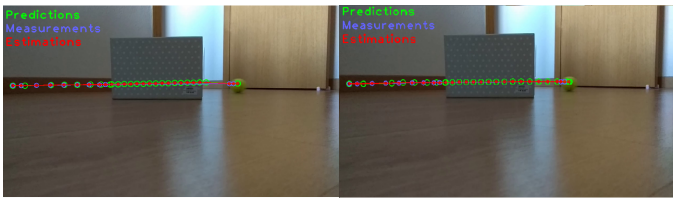

Fig. 10. Ball trajectory for video6.mp4 sequence described by Kalman filer based on constant velocity model (left) and constant acceleration model (right) according to final parameters in I

to 30. On the other hand, occlusions are solved by means of Kalman filtering. For both constant velocity and constant acceleration based filters, the value of process covariance related to velocity in the horizontal direction was increased in order to fit the trajectory of the ball while its occluded. For the constant acceleration case, the value of process covariance related to acceleration in vertical direction was decreased in order to avoid generating a parabolic trajectory. Finally, the values of measurement covariance for the vertical direction were increased in order to smooth the measurement variations when either the bottom or upper part of the ball is recovered as the central position. A sample of the results obtained for Kalman filtering based on constant velocity and acceleration is shown in figure 10, where acceleration model describes again better the ball trajectory. Kalman filter is defined for this fourth sequence as specified in TABLE 1.

VI. DEVELOPMENT OF TASK 3.3

A. DATA DESCRIPTION

The data used for this task consists in four different scenes which resemble more of a possible real world use case for the developed code. All of the sequences are more complex than the previous analyzed videos in terms of illumination, noise and number of moving objects in the scene. The four analyzed videos were:

1) abandonedBox: This scene has two main challenges: the noise and the trajectory of the target object. The captured scene has low quality and the video has a "grainy" look to it that is not frame consistent, which difficults the foreground extraction. Regarding the complexity of the trajectory of the target object, the main problem is that there is a sudden decrease in speed, which results in the object coming to a stop for a long period of time. The Kalman tracker has to be able to deal both with the noisy measurements and the changes in speed with precision and without drifting.

2) boats: The high complexity of this scene is due to the flickering in the background as well as other moving objects in the background. The water in the background causes flickering that has to be filtered out by the foreground detector. The other objects in the background are a more complex problem, as they intersect the target object, and there is not a direct way of removing them from the foreground mask. On top of that, the target object moves slowly, which can cause problems with medium or high learning rates in the foreground segmentator.

Video Sequence	Covariance of Process Noise	Covariance of Measurement Noise	Covariance of Error	Background / Foreground Segmentation and Blob Extraction
Baseline	$\begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 25 & 0 \\ 0 & 25 \end{bmatrix}$	$\begin{bmatrix} 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix}$	Learning rate of 0.001, shadow detection enabled, variance threshold 1, history 50 and Min width and min height equal to 50
Singleball.mp4	$\begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}$	$\begin{bmatrix} 25 & 0 \\ 0 & 25 \end{bmatrix}$	$\begin{bmatrix} 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix}$	Learning rate of 0.001, shadow detection enabled, variance threshold 1, history 50 and Min width and min height equal to 50
Video2.mp4	$\begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 25 & 0 \\ 0 & 25 \end{bmatrix}$	$\begin{bmatrix} 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix}$	-
Video3.mp4	$\begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}$	$\begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \end{bmatrix}$	Learning rate of 0.005
Video5.mp4	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix}$	Disable Shadow detection Learning rate of 0.01
Video6.mp4	$\begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 900 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}$	$\begin{bmatrix} 25 & 0 \\ 0 & 1000 \end{bmatrix}$	$\begin{bmatrix} 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix}$	Min width and min height equal to 30

TABLE I

SUMMARY OF SPECIFIED PARAMETERS FOR EACH SCENE OF TASKS 3.1 AND 3.2 ACCORDING TO KALMAN FILTERING BASED ON CONSTANT ACCELERATION STATE MODEL.

3) pedestrians: The challenge of this scene is mainly to deal with the bad illumination and big shadows that appear in the image.

4) streetCornerAtNight: This scene tests the ability of the tracker to track properly fast moving objects. Apart from that, the scene is poorly illuminated and the target object is surrounded by an intense halo of light that could introduce noise to the foreground segmentation.

B. EXPERIMENTAL METHODOLOGY

For each of the sequences, the initial parameter values for the foreground segmentation were taken from the baseline for the task 3.2, and the initial parameters for the Kalman filter were set according to [5]. After that, the parameters for the foreground segmentation were adjusted in order to achieve the best possible foreground mask in which, with the goal of removing all of the objects except for the target object. This includes adjusting the learning rate, the history size, the foreground threshold and the shadow detection. After that, the connectivity and min width and height values were adjusted in order to make the object dejection as accurate and consistent as possible. Finally, the Kalman filter was tuned to make the object tracking smooth and precise. This cycle was repeated several times for every sequence until the final result was satisfactory.

C. RESULTS AND ANALYSIS

For every scene, the constant acceleration model was used, as for all of the previous videos it was proven to be better then the constant velocity model. All of the exact parameters are detailed in TABLE 2.

1) abandonedBox: As mentioned before, the sequence has a lot of noise, so it is complicated to remove all the noise in the image and still have the target object as a single big blob in the image. In order to do so, the shadow detection was turned of in the background segmentator and the *"varThreshold"* was lowered to 5. This made the image more noisy, but the main blob more consistent after removing the noise with the morphological operations. Although after these changes the mask has more noise, the target object is no longer separated in multiple blobs in the majority of the frames in the sequence. The difference after adjusting these parameters is shown in figure 11. The connectivity value set to 8 and the min sizes for the max blob extraction set to 15 showed good results extracting the main blob. In order to smooth the estimation of the tracker, the values in the *"processNoiseCov"*, *"measurementNoiseCov"* and *"errorCovPost"* matrices were tuned so that when some noisy measurements occur (especially at the beginning of the sequence, when the target object gradually appears in the scene) the estimated trajectory is smooth. This was achieved mainly by raising the values in the *"measurementNoiseCov"*

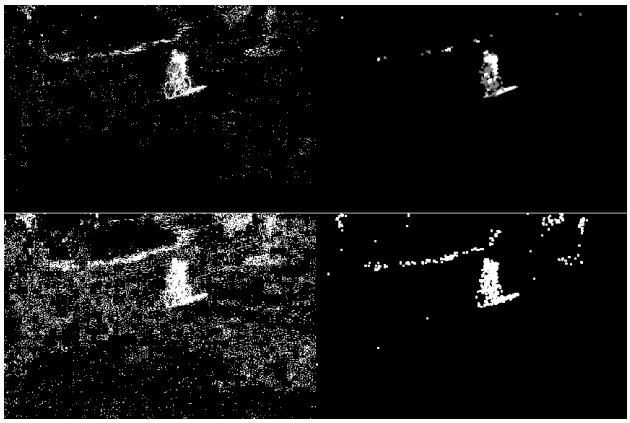

Fig. 11. Comparison of background/foreground segmentation for abandonedBox sequence with default (top) and our (bottom) parameters. The images on the left are the unfiltered masks, while the images on the left are filtered to remove noise.

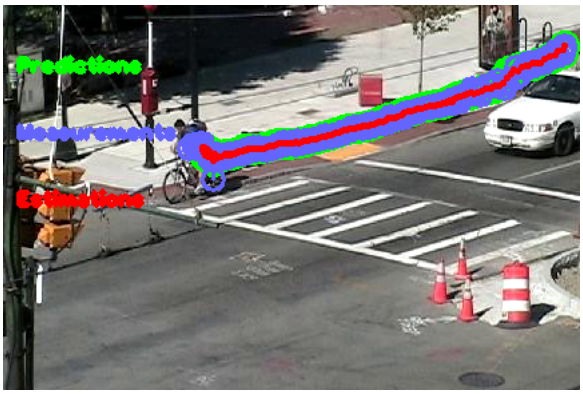

Fig. 12. Final result for the abandonedBox sequence.

matrix and lowering the values of the *"processNoiseCov"* matrix. The resulting trajectory of this configuration applied to the *"abandonedBox"* sequence can be seen in figure 12. The result is is perfectly accurate, apart for a minimum drifting at the end of the sequence, when the target object comes to a stop for a long period of time. However even though this small drift exists, the estimated position of the target object does never appear out of said object.

2) boats: Similarly to the previous video, there is a big amount of noise in the same area of the video, but because of the slow movement of the target object, the learning rate of the foreground segmentation can not be raised to lower it. The solution in this case was to let the morphological operations filter out most of the noise and lowering the learning rate and *"varThreshold"* of the background segmentator to make the target object blob as consistent as possible. The problem then would be that the moving cars in the background would intersect with the target object blob, making the estimated state shift slightly from left to right. The second problem (shown in figure 13) that could not be solved was the target object splitting into two blobs due to white objects in the background (the target object is also white). The error in the measurement is difficult to smooth

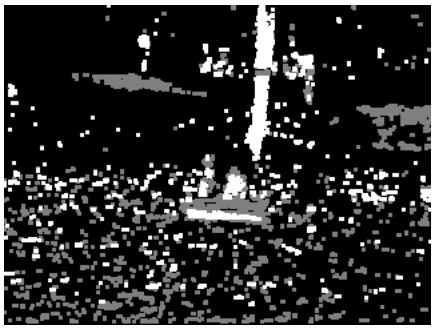

Fig. 13. The main blob (the sail of the boat) is separated in two blobs due to a white object in the background.

Fig. 14. Final result for the boats sequence.

by the Kalman filter, as it is not random noise. Nevertheless the error in the tracking can be somewhat corrected by prioritizing the prediction rather than the measurement and by trusting more the acceleration component of the state that changes slower than the position and velocity. The result can be observed in figure 14.

3) pedestrians: The main changes to the foreground segmentator deactivating the shadow detection and lowering the *"varThreshold"* value to make the blob of the target object more solid. As the main blob is selected based on the number of pixels in it rather than by the size of its box, the target object is still considered as the main blob, even though the shadow detection is deactivated in the foreground segmentation step. For the Kalman filter, the base value of the *"errorCovPost"* was found too high, and was lowered to achieve an accurate tracking. The tracked trajectory is shown in figure 15.

4) streetCornerAtNight: As the scene is shot at night and there are various light sources apart from the target object, the *"varThreshold"* of the foreground segmentator was raised to a high value (175) to minimize other background light to be detected. Some smoothing was applied to the Kalman filter by rising the values of the *"measurementNoiseCov"* and the position components of the *"processNoiseCov"*, as the light emitted by the target object causes the main blob change its size and shape during the sequence, making the detected trajectory irregular with the default Kalman parameters. The difference between the default Kalman parameters and the proposed parameters can be seen in 16.

Video Sequence	Covariance of Process Noise	Covariance of Measurement Noise	Covariance of Error	Background / Foreground Segmentation and Blob Extraction
abandonedBox	$\begin{bmatrix} 0.25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.01 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.01 \end{bmatrix}$	$\begin{bmatrix} 650 & 0 \\ 0 & 650 \end{bmatrix}$	$\begin{bmatrix} 10^5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10^5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10^5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10^5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10^5 \end{bmatrix}$	Learning rate changed to 0.0005, shadow detection disabled, variance threshold 5, history 50 and Min width and min height equal to 15
boats	$\begin{bmatrix} 200 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 200 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix}$	$\begin{bmatrix} 10^5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10^5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10^5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10^5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10^5 \end{bmatrix}$	Learning rate of 0.0001, shadow detection enabled, variance threshold 10, history 50 and Min width and min height equal to 15 and 20
pedestrians	$\begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 25 & 0 \\ 0 & 25 \end{bmatrix}$	$\begin{bmatrix} 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix}$	Learning rate of 0.001, shadow detection disabled, variance threshold 8, history 50 and Min width and min height equal to 10
streetCornerAtNight	$\begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & .1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.01 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25 & 0 & 0 \\ 0 & 0 & 0 & 0 & .1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.01 \end{bmatrix}$	$\begin{bmatrix} 700 & 0 \\ 0 & 700 \end{bmatrix}$	$\begin{bmatrix} 10^5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10^5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10^5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10^5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10^5 \end{bmatrix}$	Learning rate of 0.001, shadow detection enabled, variance threshold 175, history 50 and Min width and min height equal to 15

TABLE II

SUMMARY OF SPECIFIED PARAMETERS FOR EACH SCENE OF TASK 3.3 ACCORDING TO KALMAN FILTERING BASED ON CONSTANT ACCELERATION STATE MODEL.

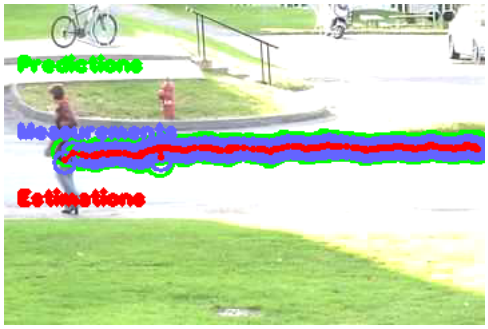

Fig. 15. Final result for the pedestrians sequence.

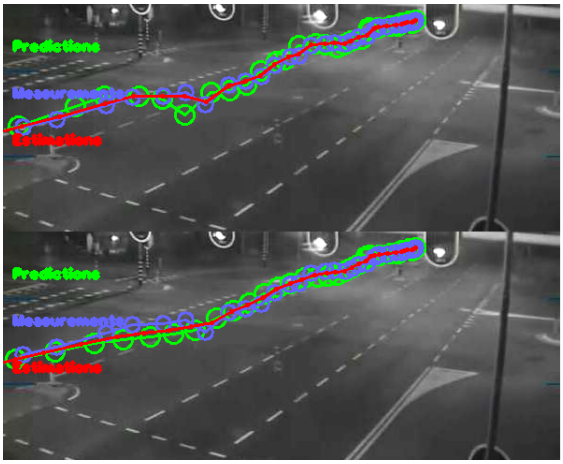

Fig. 16. Final trajectory for the streetCornerAtNight sequence with the default Kalman parameters (top) and with the proposed Kalman parameters (bottom).

VII. CONCLUSIONS

There are a lot of factors that introduce uncertainty in the object tracking problem, making it a complex task. For example, the process of measuring the state can be altered with all kinds of noises, and the measured objects can sometimes be absent or confused by another object in the scene. Even when the scene has low noise, objects can be close to each other, appearing as only one object with a different shape. This unpredictable behaviour can, most of the times, be minimized by processing the measurement across different stages, which in our case includes background segmentation, morphological operations to reduce noise and main blob extraction. The behavior can also be minimized through the design of an appropriate tracker, such as Kalman filter to reduce the uncertainty of the measured state that could not be removed during the preprocessing stage in order to estimate a more accurate trajectory.

VIII. TIME LOG

A. Maria Fernanda Herrera Perez

1) *Algorithm development at its first stage:* : 7 hours.
2) *Algorithm evaluation - Task 3.1 3.2:* : 5 hours.
3) *Writting report based on evaluation:* : 6 hours.

B. David Savary Martinez

1) Algorithm development, design and structuring optimization: : 7 hours.

2) Algorithm evaluation - Task 3.3: : 6 hours.

3) Writting report based on evaluation: : 5 hours.

REFERENCES

[1] Challa, S., Morelande, M., Mušicki, D., Evans, R. (2011). Introduction to object tracking. In Fundamentals of Object Tracking (pp. 1-21). Cambridge: Cambridge University Press.

[2] Kenshi S. (2017). Kalman Filter for Moving Object Tracking: Performance Analysis and Filter Design, Kalman Filters - Theory for Advanced Applications. Iintechopen. Available from: https://www.intechopen.com/books/kalman-filters-theory-for-advanced-applications/kalman-filter-for-moving-object-tracking-performance-analysis-and-filter-design

[3] Kalman, R. E. Others (1960). A new approach to linear filtering and prediction problems. Journal of basic Engineering (82, pp. 35-45).

[4] Colaboradores de Wikipedia (2020). Filtro de Kalman. Consulted on: April 2021 at https://es.wikipedia.org/w/index.php?title=Filtro_de_Kalmanoldid=130420936.

[5] San Miguel, J.C. Applied Video Sequence Analysis - Unit III - Visual Tracking Prediction. Image Processing and Computer Vision Master Degree. Consulted on: April 2021.