Reconstruction of trajectories with data loss by Kalman filtering: an application to Morris water maze tests

Gerardo Miramontes-de León[1], **Iván Gozález-Zamora**[1], **Arturo Moreno-Báez**[2], and **Claudia Sifuentes-Gallardo**[1]

[1] *Autonomous University of Zacatecas (UAZ), Faculty of Electrical Engineering, Master of Science in Engineering,* López Velarde 801, Centro, Zacatecas , Zac., México, 98000.

gmiram@ieee.org, aivngz@gmail.com, clausifuen@yahoo.com.mx

[2] *Autonomous University of Zacatecas, Faculty of Electrical Engineering, Industrial Electronics Engineering,* López Velarde 801, Centro, Zacatecas, Zac., México, 98000.

morenob20@uaz.edu.mx

Abstract

The Kalman filter was applied in the reconstruction of trajectories that present data loss. The trajectories were obtained by tracking a rat's swim from video sequences in the Morris water maze tests. These video are being used in neuroscience studies in spatial memory tests. In this work, the Kalman filter showed to be a good alternative to estimate position and velocity when some measurement data have been lost. The reconstruction was successfully accomplished for long and short data losses. The percent root mean square error shows a value less than 2% in the worst case when short paths were tested. For longer paths the error is less than 0.5%. The algorithm can be applied in other behavioral tests for different kind of mazes. For example, in the Guinea pig maze and elevated Y maze, where is not unusual to have obstruction of the observation path.

Keywords— Kalman filtering, Morris maze, data reconstruction.

I Introduction

The Kalman filter is a mathematical procedure that operates by means of a prediction and correction mechanism. In essence, this algorithm predicts the new state of a system from its previous estimation, adding a correction term proportional to the prediction error, in such a way that the latter is statistically minimized. The filter was introduced by Rudolf E. Kalman (1960) [10]. In this work, the filter was applied in the reconstruction of swimming trajectories in the Morris water maze. The data were obtained by a computer vision system that performs the tracking of a rat in a pool.

The detection and tracking of an object on an aquatic surface is complicated by the reflections of lights or other strange objects on the water. In [17] it is proposed to solve the problem in different ways: a) incorporating a trajectory correction algorithm in the video capture system, and b) applying elements of artificial intelligence to take into account the prehistory of the object to estimate its present and future position. It should be mentioned that sometimes the complication is due to occlusion, or unwanted shadows on the object being tracked.

In this work, there is no access to the video capturing system. It is known, however, the available data files present loss of data due to several factors, like light reflections and undesirable shadows.

The paper is organized as follows. In Section II the experiment called Morris water maze is described, together with materials and methods, like the Kalman filter in subsection II.1 and the system model used in subsection II.2. Section III shows the performance of the filter in different tests. In a first test the trajectory was obstructed intentionally by an object. Later, the Kalman filter was applied to data files obtained from experiments at the Morris water maze. Some concluding remarks are given in Section IV.

II Materials and methods

One test to measure spatial memory and recognition functions is the Morris water maze [16, 22]. This test is used in neuroscience, in laboratory rats, to estimate the effect of medications on neuronal activity [19] and the effect of stressful conditions, as in [4]. In other works, there is also interest in extracting movement patterns from video data to generate simulations based on multiple agents [25], or using data mining on the movement patterns of laboratory rats to simulate behavior patterns of movement [24].

The system consists of a pool, as shown in Fig. 1, that has a diameter between 1.5 m to 1.8 m, a submerged platform that can be removed in some phases of the test.

Figure 1: *Typical Morris water maze.*

The vision system includes specialized software to track the trajectory of the rat's swim, and also provides other parameters. The parameters of interest are the escape latency, the total distance traveled, the average speed, the total time of the test, among others.

As mentioned in the previous section, the data may suffer disturbances,either due to effects of changes in lighting, or due to misalignment in the placement of the camera. Occasionally, the tracking system loses some data from the sequence. In [15] is reported a system for the analysis of videos in AVI format that was used in the Health Sciences Laboratory of the Autonomous University of Zacatecas by the research group in Health and Environmental Sciences. As shown in Fig. 1, the setup includes an escape platform and some visual markings. In a first phase, after several attempts, the rat is able to locate and memorize the location of the platform.

The test measures the time it takes to find the escape platform. Other parameters of interest are the speed and the total distance traveled. In a second phase, the platform is removed and, in addition to the above parameters, the number of times the rat crosses the area where the platform was located is measured. In this way an estimate of spatial memory can be made. Memory can be affected by internal factors, such as some neuronal disease, or external factors, such as noise [4], among many others.

Figure 2 shows the tracking of the rat's swim, marked by a green box on the object and the coordinates of its position in the upper left corner.

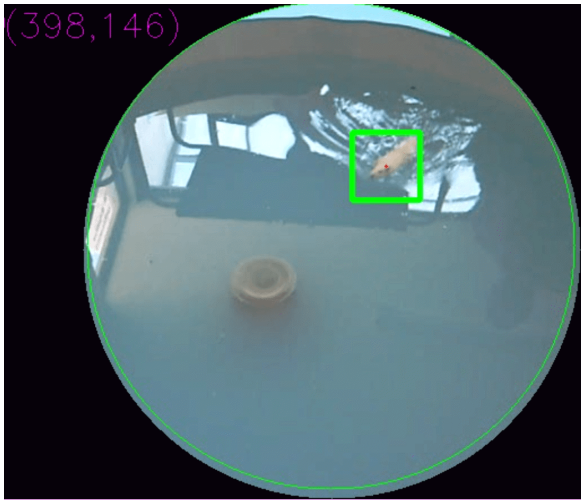


Figure 2: *Tracking on the Morris water maze.*

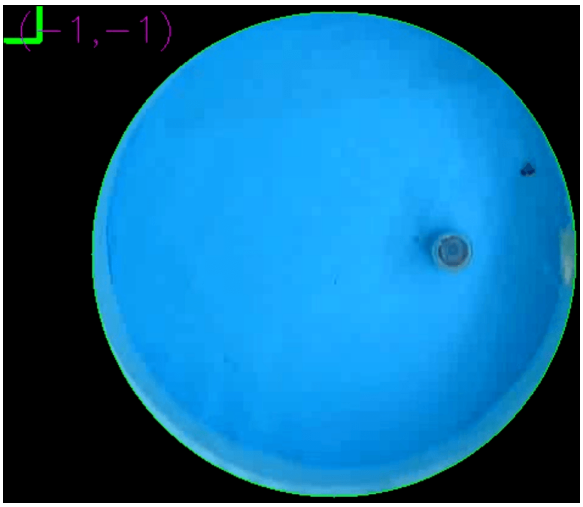
In Fig. 3(a) the effect of light reflections is shown, since there are reflections from the windows. In Fig. 3(b), data loss occurred due to poor illumination of the object to be tracked. The loss of data is shown when the green box does not enclose the object of interest. The name P3R1 file corresponds to the keywords used in the neuroscience laboratory. For example, P3 is the test number three and R1 is the rat number one. Other files names used by the Health and Environmental Sciences group are not relevant in this work because they have no effect in any of the results.

The interest of applying a reconstruction algorithm goes beyond the Morris water maze tests. There are other types of tests where lost of data occurs frequently. That is the case, for example, in the Guinea pig maze shown in Figure 4. This maze requires walls high enough to prevent the Guinea pig jumps from one cell to another. So, these walls can hide the Guinea pigs or can cause shadows, limiting the effectiveness of the tracking video system.

II.1 Brief description of the Kalman filter

The Kalman filter is a computational algorithm for estimating the state vector of a process, in a way that minimizes the covariance of the error. It is a recursive filter since it works with the past, present and predicted future state. An important application is in guided navigation systems, vehicle control, and signal processing. To see different approaches in the development of the Kalman filter, [6, 7, 3, 1] can be consulted.Other approaches in the development of the following equations can be found in [2, 9]. In this work, the first part is based on [26], then the development is presented in more detail.

(a) Data loss due to reflections.



(b) Data loss due to low illumination.

Figure 3: *Data loss by reflections and shadows.*

The Kalman filter attacks the general problem of estimating the state $\mathbf{x}$ of a discrete-time process which is governed by a linear stochastic difference equation:

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_{k-1} + \mathbf{W}_{k-1} \tag{1}$$

with measurement

$$\mathbf{z}_k = \mathbf{C}\mathbf{x}_k + \mathbf{V}_k \tag{2}$$

where $\mathbf{W}_k$ and $\mathbf{V}_k$ are noise process and noise measurement, with probability distributions

$$\begin{aligned} P(W) &\approx N(0, \mathbf{Q}), \text{ where } \mathbf{Q} = E[\mathbf{W}_k \mathbf{W}_k^T] \\ P(V) &\approx N(0, \mathbf{R}), \text{ where } \mathbf{R} = E[\mathbf{V}_k \mathbf{V}_k^T], \end{aligned} \tag{3}$$

$\mathbf{Q}$ and $\mathbf{R}$ are covariance matrices. $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are discrete-time system matrix, input matrix, and output matrix, respectively. $\mathbf{u}$ is the input or control vector.

An important condition for the Kalman filter, as it is shown later, is to assume $\mathbf{W}_k$ and $\mathbf{V}_k$ to be statistically independent

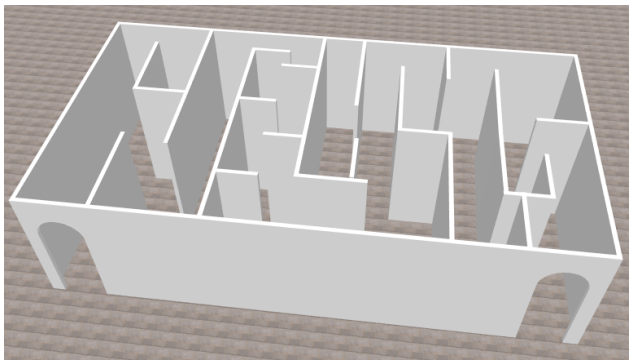


Figure 4: *A Guinea pig maze.*

white Gaussian noise sources. The Kalman gain, denoted as K, is obtained by minimizing the error covariance estimate. Defining an *a priori* estimate $\hat{\mathbf{x}}_k^-$, with the process knowledge before step k; an *a posteriori* estimate $\hat{\mathbf{x}}_k$ at the time step k, then it is possible to define two errors:

$$\begin{aligned} e_k^- &= \mathbf{x}_k - \hat{\mathbf{x}}_k^-, \text{ } \textit{a priori} \text{ error} \\ e_k &= \mathbf{x}_k - \hat{\mathbf{x}}_k, \text{ } \textit{a posteriori} \text{ error} \end{aligned} \tag{4}$$

and two error covariance matrices

$$\begin{aligned} P_k^- &= E[e_k^- e_k^{-T}], \text{ } \textit{a priori} \text{ error covariance} \\ P_k &= E[e_k e_k^T], \text{ } \textit{a posteriori} \text{ error covariance,} \end{aligned} \tag{5}$$

where the upper index "-" denotes *a priori* estimates.

The basis of the Kalman filter is an equation that allows to compute *a posteriori* estimate $\hat{\mathbf{x}}_k$ as a linear combination of *a priori* estimate $\hat{\mathbf{x}}_k^-$ and a weighted difference between the measurement $\mathbf{z}_k$ and the predicted measurement $\mathbf{C}\hat{\mathbf{x}}_k^-$, i.e.

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K(\mathbf{z}_k - \mathbf{C}\hat{\mathbf{x}}_k^-). \tag{6}$$

The term

$$(\mathbf{z}_k - \mathbf{C}\hat{\mathbf{x}}_k^-) \tag{7}$$

in (6) is called *innovation* or *residual*. K is a gain factor, an $n \times m$ matrix, that minimizes the *a posteriori* covariance error. By substitution of $\mathbf{z}_k$ in (6)

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K(\mathbf{C}\mathbf{x}_k + \mathbf{V}_k - \mathbf{C}\hat{\mathbf{x}}_k^-) \tag{8}$$

and

$$P_k = E[e_K e_k^T] = E[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x} - \hat{\mathbf{x}}_k)^T]. \tag{9}$$

The requirement is to minimize P_k with respect to K.

Expanding P_k, we have

$$\begin{aligned} P_k = E[((\mathbf{I} - K\mathbf{C})(\mathbf{x}_k - \hat{\mathbf{x}}_k^-) - K\mathbf{V}_k) \\ ((\mathbf{I} - K\mathbf{C})(\mathbf{x}_k - \hat{\mathbf{x}}_k^-) - K\mathbf{V}_k)^T]. \end{aligned} \tag{10}$$

The error $(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)$ is not correlated to $\mathbf{V}_k$, so that by taking the expected value, P_k reduces to

$$\begin{aligned} P_k = (\mathbf{I} - K\mathbf{C})E[(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)^T](\mathbf{I} - K\mathbf{C})^T \\ + KE[\mathbf{v}_k \mathbf{v}_k^T]K^T. \end{aligned} \tag{11}$$

Letting

$$E[(\mathbf{x}_k^- - \hat{\mathbf{x}}_k^-)(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)^T] = P_k^- \tag{12}$$

i.e. the *a priori* estimate of P_k, then

$$P_k = (\mathbf{I} - K\mathbf{C})P_k^-(\mathbf{I} - K\mathbf{C})^T + KRK^T \tag{13}$$

is an error covariance *update*.

Finally, taking the derivative with respect to K, letting the result equal to zero, and solving for K, the Kalman gain is

$$K = P_k^-\mathbf{C}^T(\mathbf{C}P_k^-\mathbf{C}^T + R)^{-1}. \tag{14}$$

A brief description of the Kalman filter is explained in the Algorithm 1. The Kalman filter can be developed in two stages: a prediction stage, and an update stage.

Algorithm 1 Kalman filter

1: Given initial estimates $\hat{\mathbf{x}}_k^-$ and P_{k-1}^-
2: **procedure** PREDICTION STAGE
3: Project state forward:
4: $\hat{\mathbf{x}}_k \leftarrow \mathbf{A}\hat{\mathbf{x}}_k^- + \mathbf{B}\mathbf{u}_k^- + \mathbf{w}_k^-$
5: Project error covariance forward:
6: $P_k^- \leftarrow \mathbf{A}P_{k-1}^-\mathbf{A}^T + Q$.
7: **procedure** CORRECTION STAGE
8: Calculate Kalman gain:
9: $K \leftarrow P_k^-\mathbf{C}^T(\mathbf{C}P_k^-\mathbf{C}^T + R)^{-1}$.
10: Update estimate with measurement $\mathbf{z}_k$
11: ie: $\hat{\mathbf{x}}_k \leftarrow \hat{\mathbf{x}}_k^- - K(\mathbf{z}_k - \mathbf{C}\hat{\mathbf{x}}_k^-)$.
12: Update covariance error with optimal K_k:
13: $P_k \leftarrow (I - K_k\mathbf{C})P_k^-$.
14: **goto** *2* and let $\hat{\mathbf{x}}_k^- \leftarrow \hat{\mathbf{x}}_k$ and $P_{k-1}^- \leftarrow P_k$.

II.2 System model

The system is modeled by a vector of states $\mathbf{X}$, and a set of equations called the system model. The observation time has the form $t_k = t_0 + \Delta t$, where Δt is the sampling interval. We define X_k as the state in time t_k. Also, we assume that Δt is small so that we can use a linear model. The state vector is:

$$\mathbf{X}_k = (x_k\, y_k\, V_{x_k}\, V_{y_k}) \tag{15}$$

The state equation is

$$\begin{bmatrix} x_k \\ y_k \\ V_{x_k} \\ V_{y_k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ V_{x_{k-1}} \\ V_{y_{k-1}} \end{bmatrix} + \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} \tag{16}$$

and the measurement equation is

$$\begin{bmatrix} x_k \\ y_k \\ V_{x_k} \\ V_{y_k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ V_{x_k} \\ V_{y_k} \end{bmatrix} + \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \tag{17}$$

where x_k and y_k represent the position coordinates; V_{x_k} represents the speed in the direction x, and V_{y_k} represents the speed in the direction y. In addition, w_k and μ_k model system errors and measurement errors, respectively. It should be noted that the measurement vector corresponds to the data obtained by the video analysis, that is, it is only necessary to read the data file obtained in [15].

The parameters Q and R refer to the process noise covariance and measurement noise covariance matrices. In the context of tracking objects in video, R means the detection error. The R matrix describes the uncertainty about the location of the object. So, for the (x, y) coordinates, the corresponding diagonal values of R should be a few pixels. In this application R was set to:

$$R = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \tag{18}$$

On the other hand, Q specifies how much the actual motion of the object deviates from the model. A rule of thumb is to set Q not equal to zero.

In this application Q was set to:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{19}$$

The sampling time corresponds to a video of 30 frames per second (fps).

III Results and Discussion

To show the performance of the Kalman filter in the reconstruction of trajectories with data loss, a test was made in the tracking of an object which is obstructed by another object. It is worth noting that this is an extreme case, where there is considerable data loss.

In Fig. 5, frames of a video sequence are presented, where the tracking algorithm shows, with points in blue, the detected position of the object. The tracking algorithm presented in [15] delivers data points based on the color of the object. In this case the object of interest is in red, while the points in blue are the detected position.

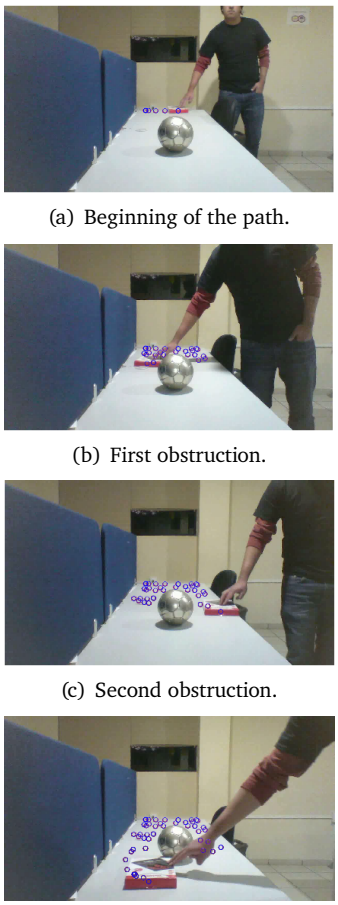
Following the sequence of Fig. 5(a) to Fig. 5(d) it is shown how trajectory data is lost due to obstruction by another object. This is an example of a big data loss, because the obstruction remains for several sampling points.

After applying the Kalman filter, the trajectory shown in Figure 6 was obtained. It can be seen how the filter has included new points in the trajectory, that is, the points shown in red are the result of the estimation made by the filter, at the corresponding sampling points.

It can be seen, the Kalman filter has reconstructed at least three missing paths, according to the image shown in Figure 6. One important point here is, as soon as a new data sample is available, the filter delivers some points that follow an estimated trajectory, which gets close, with some error, to the real trajectory in the next sample points.

III.1 Reconstruction in the Morris water maze

The Kalman filter was applied to data obtained from some video files. It is worth to remark, these are not simulation data



(a) Beginning of the path.

(b) First obstruction.

(c) Second obstruction.

(d) The following path after obstructions.

Figure 5: *Data loss by obstruction of the object.*

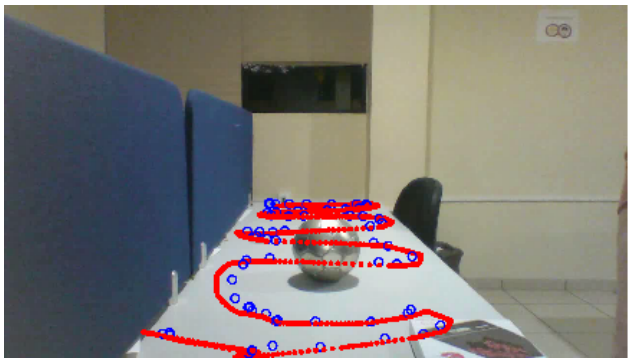


Figure 6: *Reconstructed trajectory by Kalman filter.*

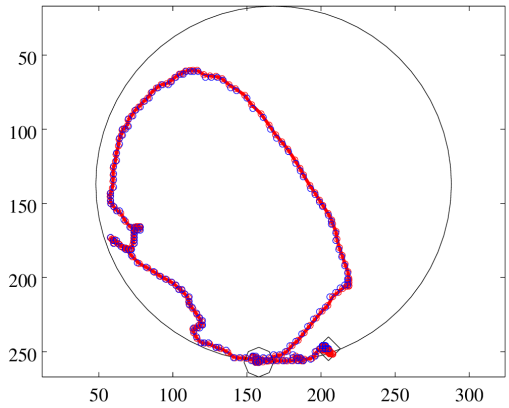


Figure 7: *Original and reconstructed trajectory for Datz data file.*

points, because in that case it would be easy to measure the true estimation error.

Figures 7 and 8 show the results for two data files. In every case, data points in blue are those (x, y) points obtained from the analysis of the video sequence. Data points in red are the estimated data points delivered by the Kalman filter.

In the first two data files, there is no significant number of missing points. In these two cases the filter follows the original trajectory very closely. Even when the length of the data file FR2L is larger than Datz, there is only one isolated red point, in the last case. That means, it is a missing data point at that sampling time.

On the analyzed P3R1 file, shown in Figure 9, it can be observed more missing data points, according to the isolated red points. Here is when the Kalman filter is useful.

A more demanding case is in when the original rat tracking data had lost several sampling data points. So, it is expected the Kalman filter "fills" that missing points. Figures 10 and 11 show the path reconstructed by the Kalman filter.

As it was shown in Figure 3(a), sometimes there is loss of data due to low illumination on the object of interest. However, the Kalman filter recovers the trajectory of the rat, as shown in Figure 12. Those are the points at the right side of the path.

III.2 Root mean square error in the reconstructed trajectory

It can be seen from Figure 6, the Kalman filter is able to reconstruct, within some error margin, the best possible trajectory. As it was described in Section II.1, the algorithm is a two stages process: a prediction stage, and an update stage. This last stage is also known as the correction stage. That means, there is also an error estimation in each sample period. In fact, when the tracking trajectory changes abruptly, the bigger the error the bigger the corection made by the filter.

One clarifying point is the following, sometimes, the Kalman filter is compared to other estimation methods. For example, for nonlinear systems, the comparison is made using the extended Kalman filter [21], the unscented Kalman filter [8], and an unbiased FIR filter [18]. The success of the comparison relies on (position and velocity) simulated data. In addition, the simulated data can be perturbed with noise to model an hypothetical measurement problem. A similar simulation example can be found in [26].

A further comment is the following, when faced with ana-

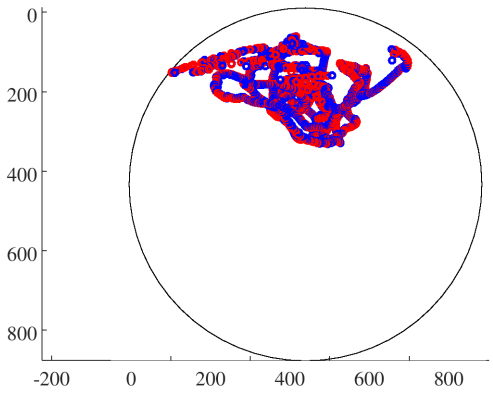


Figure 8: *Original and reconstructed trajectory for FR2L data file.*

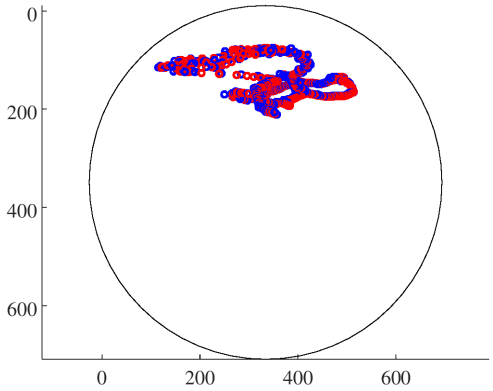


Figure 9: *Original and reconstructed trajectory for P3R1 file.*

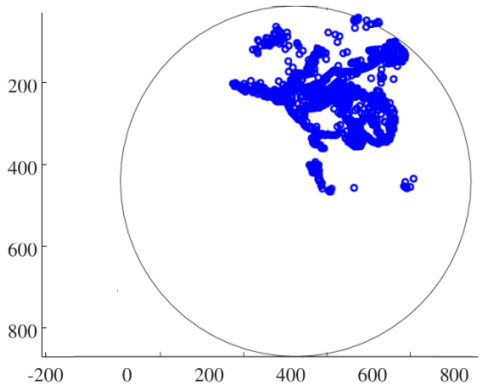


Figure 10: *Original trajectory FR4L.*

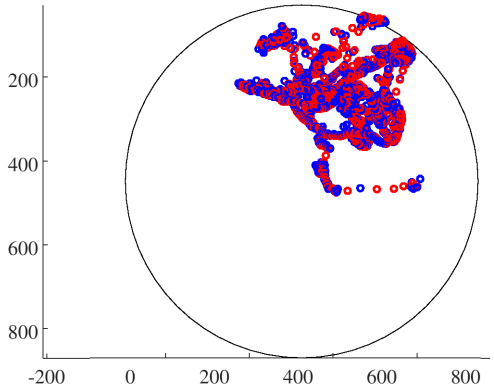


Figure 11: *Reconstructed trajectory FR4L.*

lyzing time series data, fitting splines may seem interesting. That approach essentially is a fitting approach rather than a modelling approach, so it is not considered here.

In this work, we are dealing with experimental, not simulated, data. Even in the experimental setup of Figure 5, we cannot compare a real trajectory, but a data trajectory, which contains acquisition (video capturing) and measurement errors. The Kalman filter is proposed as a solution for this particular application. Comparing the performance of the Kalman filter with other estimation algorithms is not the main scope of this work.

If there are some missing points, even when the Kalman filter delivers an estimation, the error will be calculated only on each new data sample. A good example of the assessment of a tracking problem can be found in [11], where real and desired joint trajectories are compared by means of root mean squared error graphs, from real data of the implementation.

A quantitative measure can be made by taking the root mean squared error (RMSE) between blue and red trajectories shown in Figures 7 to 12. According to [23] a way to compare two images can be done by

$$RMSE = \sqrt{\frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}(p[x,y] - \hat{p}[x,y])^2} \tag{20}$$

where the size of the image is $M \times N$, p and $\hat{p}$ are the original image pixels and the estimated image pixels, respectively.

Because we have trajectory (x, y) data points, the overall error index root mean square error (RMSE) was determined using the following expression:

$$RMSE_x = \sqrt{\frac{1}{N}\sum_{x=1}^{N}(e_x)^2} \tag{21}$$

$$RMSE_y = \sqrt{\frac{1}{N}\sum_{y=1}^{N}(e_y)^2}$$

where e_x and e_y are the difference between the input trajectory coordinates and the estimated trajectory coordinates at each sample point, and N is the number of (x, y) points used.

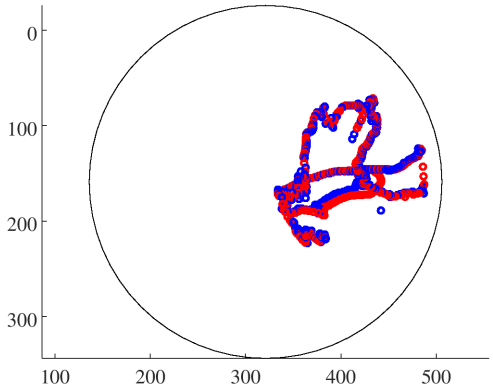


Figure 12: *Reconstructed trajectory after low illumination data loss (at right side of the path) P3R1b.*

$RMSE_x$ and $RMSE_y$ are the root mean square error in the x and y direction, respectively.

Finally,

$$RMSE_{xy} = \sqrt{RMSE_x^2 + RMSE_y^2} \tag{22}$$

where $RMSE_{xy}$ represents the 2D root mean square error.

In addition to the calculation of $RMSE_{xy}$, the percentage error was obtained. In this way, a better insight of the error is shown. To do so, it was considered the total length in meters of the trajectory, which is obtained by calibrating the experimental setup, and the accumulated $RMSE_{xy}$.

Table 1 shows, for each trial, the trajectory length in meters, the total $RMSE_{xy}$ in meters, and the percentage error. From the results, it can be observed the performance of the filter improves for longer paths, i.e, the $RMSE_{xy}$ gets a value less than 0.5 % for FR2L and FR4L tests.

Table 1: *$RMSE$ between original and estimated trajectory using Kalman*

Data file	distance (m)	$RMSE_{xy}$ (m)	% error
Datz	2.6379	0.045338	1.7187
FR2L	6.9905	0.028867	0.41295
P3R1	4.2923	0.044839	1.0459
FR4L	10.506	0.048616	0.46273
P3R1b	3.2974	0.04309	1.3068

III.3 Kalman and double exponential smoothing comparison

For a comparison of the Kalman filter performance against other estimation method, in this section, the double exponential smoothing is presented. Exponential smoothing is based on a moving average filter, so it is first reviewed.

A type of finite impulse response (FIR) filter is the moving average calculation or moving average (MA) filter. The MA filter is useful to analyze data points to smooth out short-term fluctuations by creating a series of averages of different subsets of the full data set. For details about the basics of smoothing filters see [20, 14, 13].

A moving average of order m, called m-MA, can be written as

$$\hat{y_n} = \frac{1}{m}\sum_{k=-K}^{K} y_{n+k} \tag{23}$$

where $m = 2K + 1$. The estimate of the trend-cycle at time n is obtained by averaging values of the time series within K periods of n.

Weighted moving average is a variation of the m order MA method. In general, a weighted m-MA can be written as

$$\hat{y_n} = \sum_{k=-K}^{K} a_k y_{n+k} \tag{24}$$

where the weights are given by $[a_{-k}, ..., a_k]$. The weights are symmetric so that $a_k = a_{-k}$. The simple m-MA is a special case where all of the weights are equal to $1/m$.

Finally, exponential smoothing methods (exponential moving average, EMA) are weighted averages of past observations, with the weights decaying exponentially as the observations get older.

EMA uses weighted averages, where the weights decrease exponentially, the further the data come from the past, the smallest the weights, i.e, the smallest weights are associated to the oldest data.

$$y_{n+1} = \alpha y_n + \alpha(1-\alpha)y_{n-1} + \alpha(1-\alpha)^2 y_{n-2} + \dots \tag{25}$$

where $0 \leq \alpha \leq 1$ is the smoothing parameter. The rate at which the weights decrease is controlled by the parameter α.

For the calculation of the DEMA we use the following steps:

- Step 1: Calculate the exponential moving average of order m, EMA_m
- Step 2: Apply an EMA with the same order m to EMA_m and get a smoothed EMA.
- Step 3: Multiply two times the EMA_m and subtract the smoothed EMA.

The equation can be written as:

$$\text{DEMA} = 2 \times \text{EMA}_m - \text{EMA}. \tag{26}$$

After this brief review, we compare the Kalman filter to MA methods. According to LaViola, a faster alternative to Kalman filter predictors, with no need of measurement models, is the double exponential smoothing or DEMA as it is called here [12]. The author describes the details of a predictor experiment and claims the DEMA predictor is faster, easier to implement, and perform equivalently to the Kalman and extended Kalman filtering predictors.

An additional support for this comparison is given in [5], where it is shown that double exponential smoothing can model motion by a simple linear trend equation.

The results, including $\%RMSE_{xy}$, for the DEMA algorithm are given in Table 2.

Table 2: *$RMSE$ between original and estimated trajectory using DEMA*

Data file	distance (m)	$RMSE_{xy}$ (m)	% error
Datz	2.6379	0.029764	0.78714
FR2L	6.9905	0.024518	0.35073
P3R1	4.2923	0.037691	0.87811
FR4L	10.506	0.044768	0.42611
P3R1b	3.2974	0.025732	0.78037

For a comparison between the Kalman filter and the DEMA filter, Table 3 shows the $\%RMSE_{xy}$ obtained in each case. The error in the DEMA filter is slightly smaller than Kalman error in three tests. Only in the last two tests, the Kalman filter shows almost 50% smaller error than the DEMA filter. This appears to be a surprising result, but indeed agrees with the statement made in [12].

Table 3: *$\%$ $RMSE_{xy}$ between Kalman and a DEMA filter*

Data file	Kalman	DEMA filter
Datz	1.7187	0.78714
FR2L	0.41295	0.35073
P3R1	1.0459	0.87811
FR4L	0.24784	0.42611
P3R1b	0.41263	0.78037

Finally, a mandatory comparison is the time spent by each method. Just to name a few, for two data files, Kalman filter needed 23.09 and 12.977 seconds for the P3R1 and the Datz files respectively, while DEMA only needed 0.11 and 0.035 seconds.

III.4 Velocity profile estimations

The description of the system includes (see Eq. 15) other unmeasured states, in this case the velocity of the object. So, it is possible to estimate the swimming speed between samples of the video sequence. In addition, once (x, y) position estimates are obtained, and knowing the sample rate, velocities can be calculated between a pair of consecutive data points. A distance calculation can be performed by the well known rule of distance between two points.

Figure 13 shows the velocity profile in the x direction for one data file. Obtaining the velocity profile is important in neuroscience studies where the test involves also the effects in the neuronal motor system.

A good representation of the behavior of the rat during the tests, i.e., spatial memory, movement speed related to neuronal motor system, among others, is very important.

In all behavioral studies, when tracking a living being, is common to face data loss problems. In this work it is shown one of several approaches, that is, the Kalman filter.

IV Conclusions

The Kalman filter was proposed for the estimation of trajectories in the Morris water maze, based on incomplete data.

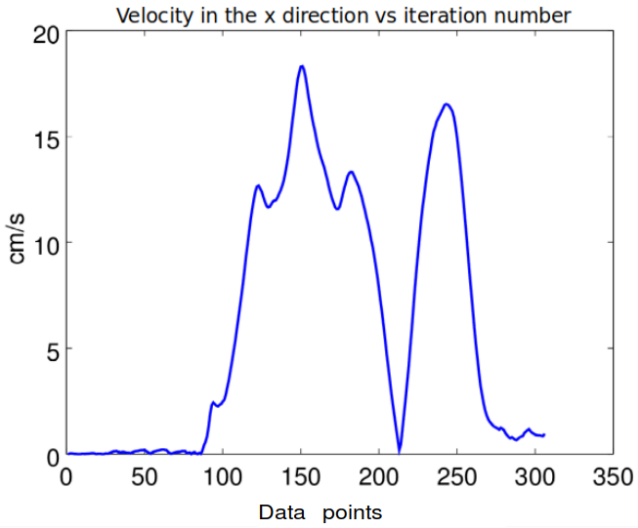


Figure 13: *Velocity estimation of the rat's swim.*

In this case, the data do not come from a simulation but are laboratory data. It was shown that the Kalman algorithm has a satisfactory performance in the estimation of incomplete data, besides that it allows to estimate other parameters of the test, such as the swim speed, and the total distance traveled. With the data obtained, important information can be obtained for studies in neurosciences, such as the distribution of the trajectory in the maze, the speed of swimming to study effects in the motor system, and the escape latency in spatial memory tests. The algorithm can be applied in other behavioral tests where different kind of mazes are used, for example in the Guinea pig maze, and elevated Y maze, where is not unusual to have obstruction of the observation path.

References

[1] P. G. Aurélien Valade and J.-Y. Fourniols. "A Study about Kalman Filters Applied to Embedded Sensors". In: *Sensors* (2017).

[2] S. M. Bozic. *Digital and Kalman Filtering, An introduction to discrete-time filtering and optimun linear estimation*. 2nd. Hasteld Press, 1994.

[3] R. G. Brown and P. Y. C. Hwang. *Introduction to Random Signal and Applied Kalman Filtering*. 4 th. John Wiley & Sons, Inc., 2012.

[4] Y. D. Burke. "Efecto Protector de los Esteroides Neuroactivos Progesterona y Dehidroepiandrosterona en la Población Neuroglial del Hipocampo de Ratas Macho Adultas, Afectadas por el Hacinamiento y el Ruido". PhD thesis. 2006.

[5] Christopher Chatfield. 1st. Boca Raton, Florida: Chapman and Hall/CRC, 2000. ISBN: 1-58488-063-5.

[6] R. Faragher. “Understanding the basis of the Kalman filter via a simple and intuitive derivation”. In: *IEEE Signal Process Mag.* 29.5 (2012), pp. 128–132.

[7] L. Galleani and P. Tavella. “Time and the Kalman Filter”. In: *IEEE Control Syst.* 30.2 (2010), pp. 44–65.

[8] *Unscented Filtering and Nonlinear Estimation*. Vol. 92. IEEE, 2004, pp. 401–422.

[9] *Kalman Filtering Theory and Application*. H. W. Sorenson editor. IEEE Press, 1985.

[10] R. E. Kalman. “A New Approach to Linear Filtering and Prediction Problems”. In: *Transaction of the ASME-Journal of Basic Engineering* (1960), pp. 34–45.

[11] J. Kern et al. “Development of an embedded control system by means of dsPIC applied in a 4 DOF robot”. In: *IEEE Latin America Transactions* 14.5 (2016), pp. 2099–2106.

[12] J. J. LaViola. “An experiment comparing double exponential smoothing and Kalman filter-based predictive tracking algorithms”. In: *IEEE Virtual Reality, 2003. Proceedings.* 2003, pp. 283–284.

[13] J. Luo, K. Ying, and J. Bai. “Properties of Savitzky-Golay digital differentiators”. In: *Digital Signal Processing* 15 (2005), pp. 122–136.

[14] P. Meer and I. Weiss. “Smoothed Differentiation Filters for Images”. In: *Journalof Visual Communications and Image Representation* 3.1 (1992), pp. 58–72.

[15] *Analysis of AVI Files for Mice Behavior Experiments in the Morris Water Maze*. Electronics, Robotics and Automotive Mechanics Conference (CERMA) IEEE, 2011, pp. 131–136.

[16] R. G. M. Morris. “Spatial localization does not require the presence of local cues”. In: *Learning and Motivation* 12 (1981), pp. 239–250.

[17] TV. Mukhina and SO. Bachurin. “Versatile computerized system for tracking and analysis of water maze tests”. In: *Behavior Research Methods, Instruments, & Computers* 33.3 (2001), pp. 371–380.

[18] R. Olivera et al. “Optimal States Estimation of an LTI System Using the Unbiased FIR Filter”. In: *IEEE LATIN AMERICA TRANSACTIONS* 13.3 (2015), pp. 609–612.

[19] IT Pereira and R Burwell. “Using the Spatial Learning Index to Evaluate Performance on the Water Maze”. In: *Behavioral Neuroscience* 129.4 (2015), pp. 533–239.

[20] A. Savitzky and M. J. E. Golay. “Smoothing and Differentiation by simplified Least Squares Procedures”. In: *Analytical Chemistry* 36.8 (1964), pp. 1627–1639.

[21] D. Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Hoboken, NJ, USA, 2006: John Wiley & Sons, 2006.

[22] R. J. Steel and R. G. Morris. “Delay-dependent impairment of a matchning-to-place task with chronic and intrahippocampal infusion of the NMDA-antagonist D-AP5”. In: *Hippocampus* 9 (1999), pp. 118–136.

[23] K. S. Thyagarajan. *Digital Image Processing with Application to Digital Cinema*. Burlington, MA 01803, USA: Focal Press, 2006.

[24] M. Tufail et al. “Mining Movement Patterns from Video Data to Inform Multi-agent Based Simulation”. In: International Workshop on Agents and Data Mining Interaction, 2014, pp. 38–51.

[25] *Multi Agent Based Simulation Using Movement Patterns Mined from Video Data*. Research and Development in Intelligent Systems XXXII, 2015, pp. 275–287.

[26] G. Welch and G. Bishop. *An introduction to the Kalman Filter*. `https://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf`. 2001.