# A Mnemonic Kalman Filter for Non-Linear Systems With Extensive Temporal Dependencies

Steffen Jung, Isabel Schlangen, *Member, IEEE*, and Alexander Charlish, *Senior Member, IEEE*

***Abstract*—Analytic dynamic models for target estimation are often approximations of the potentially complex behaviour of the object of interest. Its true motion might depend on hundreds of parameters and can involve long-term temporal correlation. However, conventional models keep the degrees of freedom low and they usually assume the Markov property to reduce computational complexity. In particular, the Kalman Filter assumes prior and posterior Gaussian densities and is hence restricted to linear transition functions which are often insufficient to reflect the behaviour of a real object. In this letter, a Mnemonic Kalman Filter is introduced which overcomes the Markov property and the linearity restriction by learning to predict a full transition probability density with Long Short-Term Memory networks.**

***Index Terms*—Dynamic models, single target tracking, long short-term memory, recurrent neural network, Kalman filter.**

## I. Introduction

THE careful design of dynamic models for target tracking is an important task since the chosen assumptions have a great influence on the quality of the state estimates. Some objects commonly exhibit complex behaviours, such as vehicles executing a series of standard manoeuvres or following specific routes, and it might be hard to formalise them in a compact and computationally feasible manner. Moreover, dynamic patterns usually contain long-term correlation, i.e. the states far in the past can influence state transitions at the current time (e.g. the departure port might influence the destination port).

Standard dynamic models are simplified descriptions of physical motion with a low number of parameters and usually include the Markov property, i.e. they are oblivious to most of the state history. One common version of the single-target Bayes recursion for Gaussian distributions is the well-known Kalman Filter (KF) [1], which depends on the dynamic model being linear and Markovian to preserve the Gaussianity of the propagated density. The Extended and Unscented KF (EKF/UKF) are approximations that relax the linearity assumption but still do not tolerate highly non-linear dynamics [2]. While ensuring tractability, such restrictions might poorly represent the true dynamic behaviour, leading to track loss in the presence of occlusions or low detection rates. Temporal information has previously been included via context data [3], if available, or by adding previous time steps to the state vector as, e.g., in auto-regressive modelling [4].

The very active field of machine learning has opened a non-analytic way of defining and utilising mathematical models. Instead of explicitly specifying the model equation, a neural network approximates the required function intrinsically by processing a set of training samples and optimising a possibly large number of hidden parameters to fit the training set most accurately. In this manner, neural networks have achieved impressive results in computer vision and acoustics [5], [6]. One specific type of network, the Recurrent Neural Network (RNN), was especially designed for processing sequential data such as language, which led to great advances in speech recognition, translation, image captioning and other applications [7]–[9]. To overcome the inherent exploding/vanishing gradient problem of RNNs, Long Short-Term Memory (LSTM) networks were introduced which are able to 'remember' temporal dependencies for much longer time periods [10]. LSTMs were already applied to model the KF matrices [11] or provide a simplified state prediction which only estimates the variance in each state dimension but not the full covariance [12]. In both cases, the transition function is still assumed to be linear. Furthermore, LSTMs were used in multi-object filters to estimate sets of target states without uncertainty information [13], [14] or to model a pixel-wise transition function for video tracking [15]. All of these methods were formulated with first-order Markov transitions, hence they are oblivious of the target state history. The Markov assumption has been relaxed in [16], but the proposed state transition is a linear transformation in terms of rotation and translation and it does not explicitly predict uncertainty.

In this letter, the single-target Bayes recursion is reformulated with a non-Markovian transition function to accommodate long-term dependencies (Section II). An LSTM network is trained to learn full Gaussian densities from sequences of input states in the manner of [17] (Section III-A). This Multivariate Density Long Short-Term Memory (MD-LSTM) network is then used in the KF prediction step to estimate the future state of a single object *along with its covariance* (Section III-B). The network produces a Gaussian output by definition but it is not restricted to linear transformations, being a universal approximator [18]. It overcomes the Markov property through long-term correlations that LSTMs are designed to encode. The resulting framework is a Mnemonic Kalman Filter (MKF) that learns complex, possibly highly non-linear object dynamics and can predict a full probability density of the target state dependent

Manuscript received February 12, 2020; revised May 21, 2020; accepted June 2, 2020. Date of publication June 8, 2020; date of current version June 25, 2020. This work was funded by the German Federal Ministry of Defence (BMVg). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Steeve Zozor. *(Corresponding author: Isabel Schlangen.)*

The authors are with the Department Sensor Data and Information Fusion (SDF), Fraunhofer Institute for Communication, Information Processing and Ergonomics (FKIE), 53343 Wachtberg, Germany (e-mail: steffen.jung@fkie.fraunhofer.de; isabel.schlangen@fkie.fraunhofer.de; alexander.charlish@fkie.fraunhofer.de).

Digital Object Identifier 10.1109/LSP.2020.3000679

on thousands of previous states. First experiments (Section IV) show an improved performance of the MKF in comparison to the standard KF, especially in the presence of occlusions or low detection rates.

## II. Single-Object Bayesian State Estimation

In the following, let us assume that a stream of data is received at discrete time steps with indices $t \in \mathbb{N}$. Bayesian state estimation recursively predicts the future $d_x$-dimensional state $\boldsymbol{x}_{t+1}$ of a dynamic system based on previous $d_z$-dimensional observations $\boldsymbol{z}_{0:t} = \{\boldsymbol{z}_0, \dots, \boldsymbol{z}_t\}$, and then corrects this belief according to the newly received measurement $\boldsymbol{z}_{t+1}$.

We wish to write the predicted Probability Density Function (PDF) of the next target state $\boldsymbol{x}_{t+1}$ conditioned on the previous observations $\boldsymbol{z}_{0:t}$ as an integral over the previous time step $t$. This is achieved with the *Chapman-Kolmogorov equation*

$$p_{t+1|t}(\boldsymbol{x}_{t+1}|\boldsymbol{z}_{0:t}) = \int f_{t+1|t}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{z}_{0:t}) p_{t|t}(\boldsymbol{x}_t|\boldsymbol{z}_{0:t}) \mathrm{d}\boldsymbol{x}_t, \tag{1}$$

where $p_{t|t}$ is the posterior PDF at time $t$ and $f_{t+1|t}$ denotes the transition function to time $t+1$. Note that the observations $\boldsymbol{z}_{0:t}$ carry information on the previous (hidden) states $\boldsymbol{x}_{0:t} = \{\boldsymbol{x}_0, \dots, \boldsymbol{x}_t\}$, and they lead to estimates $\tilde{\boldsymbol{x}}_{0:t} = \{\boldsymbol{x}_{0|0}, \dots, \boldsymbol{x}_{t|t}\}$ which are ideally sufficient statistics of $\boldsymbol{z}_{0:t}$.[1] Usually, past states $\boldsymbol{x}_{0:t-1}$ are disregarded in the state transition from $t$ to $t+1$, i.e. the state dynamics are regarded as a first-order Markov process $f_{t+1|t}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{z}_{0:t}) \hat{=} f_{t+1}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)$. However, in the following, a non-Markovian transition function $f_{t+1|t}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{z}_{0:t}) \hat{=} f_{t+1|t}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \tilde{\boldsymbol{x}}_{0:t})$ is used that is conditioned on all previous estimates $\tilde{\boldsymbol{x}}_{0:t}$. With this, we arrive at a non-Markovian Chapman-Kolmogorov equation of the form

$$p_{t+1|t}(\boldsymbol{x}_{t+1}|\boldsymbol{z}_{0:t}) = \int f_{t+1|t}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \tilde{\boldsymbol{x}}_{0:t}) p_{t|t}(\boldsymbol{x}_t|\boldsymbol{z}_{0:t}) \mathrm{d}\boldsymbol{x}_t. \tag{2}$$

Upon receiving a new measurement $\boldsymbol{z}_{t+1}$, the posterior PDF $p_{t+1|t+1}$ is obtained from (2) using *Bayes' Rule:*

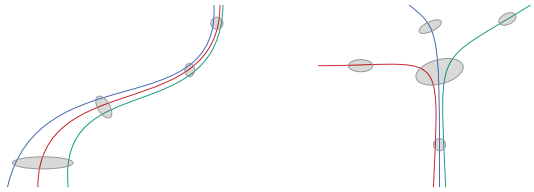
$$p_{t+1|t+1}(\boldsymbol{x}_{t+1}|\boldsymbol{z}_{0:t+1}) = \frac{p_{t+1|t}(\boldsymbol{x}_{t+1}|\boldsymbol{z}_{0:t}) g_{t+1}(\boldsymbol{z}_{t+1}|\boldsymbol{x}_{t+1})}{\int p_{t+1|t}(\boldsymbol{x}|\boldsymbol{z}_{0:t}) g_{t+1}(\boldsymbol{z}_{t+1}|\boldsymbol{x}) \mathrm{d}\boldsymbol{x}}, \tag{3}$$

$g_{t+1}(\boldsymbol{z}|\boldsymbol{x})$ being the association likelihood between $\boldsymbol{z}$ and $\boldsymbol{x}$.

To arrive at a variation of the KF from (2) and (3), the PDFs are defined to be Gaussian. Analytically, the Gaussianity is only guaranteed to be preserved under linear transformations, meaning that the dynamic and measurement models are assumed linear in traditional literature. This restriction naturally limits the modelling choices. In Section III, we will describe a neural network that is able to predict a full Gaussian PDF from a sequence of states according to a non-Markovian and possibly non-linear transition model.

## III. Mnemonic Kalman Filter

In this section, the Mnemonic Kalman Filter (MKF) is introduced which uses an MD-LSTM network for the object state prediction. The filter is called *mnemonic* since the underlying learning algorithm constructs an internal model that is used to remember long-term dynamic dependences and therewith learns to predict behavioural patterns of an object of interest.



(a) Object motion may be more uncertain in specific areas. (b) High uncertainty can arise at junctions with multiple pathways.

Fig. 1. Possible sources of uncertainty in behavioural patterns.

### A. Multivariate Density Long Short-Term Memory

For the MKF transition $f_{t+1|t} = f_{\mathrm{MKF}}$, a specialised LSTM generates a multivariate normal distribution $\mathcal{N}(\boldsymbol{x}_{t+1}|\hat{\boldsymbol{x}}_{t+1|t}, \hat{\boldsymbol{P}}_{t+1|t})$ from the current estimate $\boldsymbol{x}_{t|t}$ as well as the estimates $\tilde{\boldsymbol{x}}_{0:t-1}$ inputted at previous time steps, where $\hat{\boldsymbol{x}}_{t+1|t}$ and $\hat{\boldsymbol{P}}_{t+1|t}$ are the predicted mean and covariance. By learning a Gaussian PDF, it is possible to encode model-inherent uncertainty in addition to the state vector, see Fig. 1. The two core aspects of this network, i.e. encoding Gaussianity and learning long-term information, are described below.

*1) Learning a Gaussian PDF:* The first step of constructing the desired transition function $f_{\mathrm{MKF}}$ is to define a way to train an arbitrary neural network to predict a multivariate Gaussian density $\mathcal{N}(\boldsymbol{x}|\hat{\boldsymbol{x}}_Y, \hat{\boldsymbol{P}}_Y)$ from a multivariate random variable $\boldsymbol{Y}$. According to [17], this can be achieved by constructing the network such that it produces $d(d+3)/2$ outputs forming the multivariate mean $\hat{\boldsymbol{x}}_Y$ (having $d$ entries) and the Cholesky decomposition $\hat{\boldsymbol{C}}_Y$ of the covariance $\hat{\boldsymbol{P}}_Y$ (which has $d(d+1)/2$ non-zero entries).[2] The network is then trained by minimising a suitable *loss function*, i.e. the negative log likelihood of the multivariate normal distribution (see [17])

$$\mathcal{L}(\boldsymbol{y}^1, \dots, \boldsymbol{y}^N) = \sum_{i=1}^{N} -\log(\mathcal{N}(\boldsymbol{y}^i|\hat{\boldsymbol{x}}_Y, \hat{\boldsymbol{P}}_Y)). \tag{4}$$

with respect to the elements of $\hat{\boldsymbol{x}}_Y$ and $\hat{\boldsymbol{P}}_Y$ based on $N$ realisations $\{\boldsymbol{y}^1, \dots, \boldsymbol{y}^N\}$ of $\boldsymbol{Y}$. The explicit implementation of the loss function and its derivatives with respect to each of the output variables is derived and explained with great detail in [17] and is hence omitted here.

*2) Learning Temporal Information:* Like in [16], temporal information will be included in the transition $f_{\mathrm{MKF}}$ by using an LSTM neural network. LSTM networks are *recurrent*, i.e. they always feed their internal state back as an additional input such that the current network output depends on many previous inputs. In contrast to basic RNNs, LSTMs have a special structure that make it possible to remember relevant information over long time sequences, see [10].

[1] Note that $x_t$ denotes the hidden target state at time $t$, whereas $\boldsymbol{x}_{t|t'}$ is the state estimate at time $t$ conditioned on $\boldsymbol{z}_{0:t'}$.

[2] Here, the Cholesky decomposition is used to guarantee that $\hat{\boldsymbol{P}} = \hat{\boldsymbol{C}}^T \hat{\boldsymbol{C}}$ is indeed a symmetrical, positive definite covariance matrix.

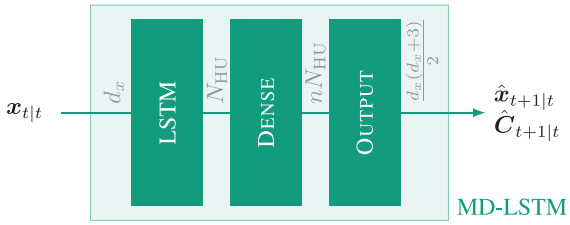



Fig. 2. The MD-LSTM network architecture for $f_{\mathrm{MKF}}$.

In order to use an LSTM for tracking, the network has to be trained through backpropagation on a set of sample trajectories such that it learns the dynamics that the samples follow. For robustness, an additional fully connected layer is incorporated which adds another level of abstraction to the learned information. An output layer reduces the number of hidden units to $d_x(d_x+3)/2$. The activation function used for the LSTM and the dense layer is $tanh$, and the output layer is linear. The number of parameters in each layer is as follows:

- *Input:* For each time step $t$, the network receives a state estimate $\boldsymbol{x}_{t|t}$, i.e. each input is $d_x$-dimensional.
- *LSTM layer*: The number of hidden units, $N_{\mathrm{HU}}$, in the LSTM layer is a degree of freedom which is set according to the complexity of the underlying dynamics.
- *Dense layer:* $nN_{\mathrm{HU}}$ hidden units are set for this layer, i.e. an $n$-fold multiple of $N_{\mathrm{HU}}$ with $n \in \mathbb{N}$.
- *Output:* As described in III-A1, the necessary number of outputs to obtain $\hat{\boldsymbol{x}}_{t+1|t}$ and $\hat{\boldsymbol{C}}_{t+1|t}$ is $d_x(d_x+3)/2$.

The full network used in this article is shown in Fig. 2.

### B. Filter Implementation

Once the network is trained in the manner of Section III-A, it can be used to predict Gaussian PDFs inside the KF. The trained predictor $f_{\mathrm{MKF}}$ takes the previous $d_x$-dimensional posterior mean $\boldsymbol{x}_{t|t}$ and returns a Gaussian density $\mathcal{N}(\boldsymbol{x}_{t+1}|\hat{\boldsymbol{x}}_{t+1|t}, \hat{\boldsymbol{P}}_{t+1|t})$ by reverting the obtained Cholesky decomposition. White process noise with covariance $\boldsymbol{Q}_{t+1}$ is added to compensate for training inaccuracies. Consequently, the resulting predicted mean and covariance are

$$\boldsymbol{x}_{t+1|t} = \hat{\boldsymbol{x}}_{t+1|t}, \tag{5}$$

$$\boldsymbol{P}_{t+1|t} = \hat{\boldsymbol{C}}_{t+1|t}^T \hat{\boldsymbol{C}}_{t+1|t} + \boldsymbol{Q}_{t+1}. \tag{6}$$

It is important to note that $f_{\mathrm{MKF}}$ is not necessarily linear, however it returns a Gaussian by definition which makes it suitable to be used in the KF. Therefore, the proposed MKF can be updated using the conventional Kalman update [1]

$$\boldsymbol{y}_{t+1} = \boldsymbol{z}_{t+1} - \boldsymbol{H}_{t+1}\boldsymbol{x}_{t+1|t}, \tag{7}$$

$$\boldsymbol{S}_{t+1} = \boldsymbol{H}_{t+1}\boldsymbol{P}_{t+1|t}\boldsymbol{H}_{t+1}^T + \boldsymbol{R}_{t+1}, \tag{8}$$

$$\boldsymbol{K}_{t+1} = \boldsymbol{P}_{t+1|t}\boldsymbol{H}_{t+1}^T \boldsymbol{S}_{t+1}^{-1}, \tag{9}$$

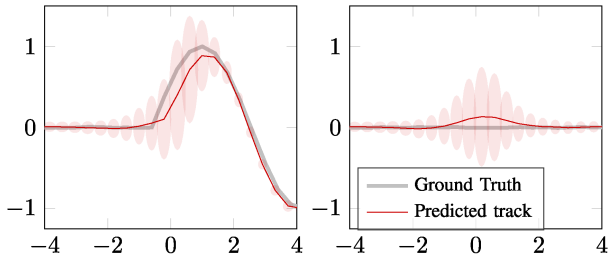
$$\boldsymbol{x}_{t+1|t+1} = \boldsymbol{x}_{t+1|t} + \boldsymbol{K}_{t+1}\boldsymbol{y}_{t+1}, \tag{10}$$

$$\boldsymbol{P}_{t+1|t+1} = (\boldsymbol{I}_{d_x} - \boldsymbol{K}_{t+1}\boldsymbol{H}_{t+1})\boldsymbol{P}_{t+1|t} \tag{11}$$

with measurement matrix $\boldsymbol{H}_{t+1}$, noise covariance $\boldsymbol{R}_{t+1}$, and the $d_x \times d_x$ identity matrix $\boldsymbol{I}_{d_x}$. The general workflow of the proposed MKF recursion is displayed in Algorithm 1.

TABLE I
TRAINING PARAMETERS FOR THE SHOWN EXPERIMENTS

| Section | $\sigma$ | $\eta$ | $p_{\mathrm{drop}}$ | $N_{\mathrm{e}}$ | $N_{\mathrm{HU}}$ | $n$ |
|---|---|---|---|---|---|---|
| IV-A | $1.0\Delta$ | $2.5 \cdot 10^{-4}$ | 0.0 | 8500 | 250 | 2 |
| IV-B | 0.02 | $1.0 \cdot 10^{-5}$ | 0.2 | 12672 | 256 | 1 |




(a) Linear to sinusoidal transition. (b) Fully linear trajectory.

Fig. 3. Exp. 1: Predictions of an MD-LSTM.

**Algorithm 1:** Pseudo-Code for The Proposed MKF.

*Parameters:* Predictor $f_{\mathrm{MKF}}$; $\boldsymbol{Q}_{t+1}$, $\boldsymbol{H}_{t+1}$, $\boldsymbol{R}_{t+1}$
*Input:* Initial Gaussian $\mathcal{N}(\boldsymbol{x}_0|\boldsymbol{x}_{0|0}, \boldsymbol{P}_{0|0})$
**while** *tracking* **do**
  *MD-LSTM prediction:*
  Predict $\hat{\boldsymbol{x}}_{t+1|t}$, $\hat{\boldsymbol{C}}_{t+1|t}$ from $\boldsymbol{x}_{t|t}$ using $f_{\mathrm{MKF}}$;
  Calculate $\boldsymbol{x}_{t+1|t}$ and $\boldsymbol{P}_{t+1|t}$ using (5), (6);
  *KF Update:*
  **if** *measurement* $\boldsymbol{z}_t$ *received* **then**
    Calculate $\boldsymbol{x}_{t+1|t+1}$ and $\boldsymbol{P}_{t+1|t+1}$ via (7)–(11);
  **end**
**end**

The proposed algorithm's online computational complexity only differs from a standard KF by a small overhead of $\mathcal{O}(N_{HU})$, leading to an overall complexity of $\mathcal{O}(N_{HU} + d_x^3)$, where $d_x^3$ stems from the matrix inversion $\boldsymbol{S}_{t+1}^{-1}$ in Eq. (9).

## IV. EVALUATION

In this section, two experiments demonstrate the functionality of an MD-LSTM predictor and compare the full MKF (plotted in red below) to a standard KF (plotted in blue). All sample trajectories are corrupted with additional white noise with standard deviation $\sigma$ to mimic data variation. The network is trained on a system with an i7-6700 processor and 16 GB of RAM with learning rate $\eta$, dropout rate $\mathrm{p_{drop}}$ and $N_{\mathrm{HU}}$ LSTM hidden units (resp. an $n$-fold multiple $nN_{\mathrm{HU}}$ units in the dense layer) over $N_{\mathrm{e}}$ iterations as listed in Table I. Version 1.0.0-beta4 of deeplearning4j was used for the implementation.

### A. The MD-LSTM as a Predictor

This experiment aims to show that MD-LSTMs can in fact learn different dynamics of a single target and their transitions at once, even if they partially coincide. To illustrate this, consider one target which moves in $x$ direction of a two-dimensional Cartesian coordinate system with a constant velocity $\Delta = 0.1$. The first half of the path is always linear; at the origin, the target either changes into a sinusoidal movement according to $\sin(t\Delta)$

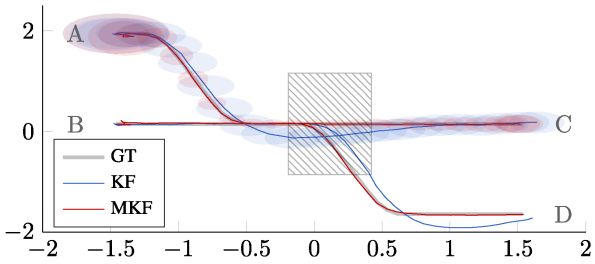



Fig. 4. Exp. 2: Ground truth and posterior densities for both filters, showing the estimated covariances for trajectory 1.

with probability 0.5 or else continues its linear path. The two cases are displayed in grey in Fig. 3.

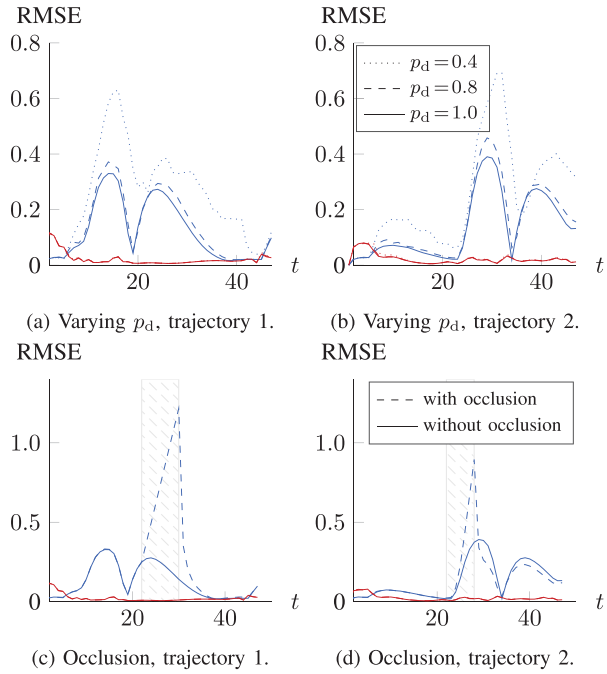
During training, the network is presented with $x/y$ positions of both trajectories that are additionally corrupted by white noise with standard deviation $\sigma = 1.0\Delta$ to simulate variation among the samples. By training on many trajectories consecutively, the network learns to predict the next 2-dimensional target position $\hat{\boldsymbol{x}}_{t+1|t}$ and its covariance $\hat{\boldsymbol{P}}_{t+1|t}$ based on the previous states $\boldsymbol{x}_{0:t}$. The tentative predictions $\hat{\boldsymbol{x}}_{t+1|t}$ form the red trajectories shown in Fig. 3(a) and 3(b), while the red ellipses are the learned covariances based on the intrinsic uncertainty in the training samples. The plots clearly show how the learned densities encode the uncertainty introduced by the variety in the training set. Firstly, the estimated means get a bias towards the respective other trajectory at the junction, which is a wanted effect since the output density shall represent all possible trajectories at once. Secondly, the variance only increases vertically at the junction since the horizontal velocity is the same in both cases.

### *B. Tracking With Varying Detection Rates and Under Occlusion*

This experiment analyses how the MKF exploits long-term dependencies to reconnoitre ambiguities at crossings with the help of previous target locations. For this purpose, the network is trained on noisy $x/y$ positions of two Dubins trajectories shown in Fig. 4. Pathway 1 starts at A ending in C and pathway 2 goes from B to D. The performance of the MKF is compared with a standard KF using a Near Constant Velocity (NCV) model in the prediction. The KF target state is of the form $\boldsymbol{x}_t = (x, \dot{x}, y, \dot{y})$ with both position and velocity components of the object in $x$ and $y$. For a given time interval $T = 1$ s we use the transition $\boldsymbol{F}_t = \left(\begin{smallmatrix} 1 & T \\ 0 & 1 \end{smallmatrix}\right) \otimes \boldsymbol{I}_2$ and process noise matrices $\boldsymbol{Q}_t^{\mathrm{KF}} = 0.008 \cdot \left(\begin{smallmatrix} T^4/3 & T^3/2 \\ T^3/2 & T^2 \end{smallmatrix}\right) \otimes \boldsymbol{I}_2$ [19]; for the MKF, we set $\boldsymbol{Q}_t^{\mathrm{MKF}} = 0.002 \cdot \boldsymbol{I}_2$. Both filters adopt a linear measurement model according to (7)–(11) with $\boldsymbol{H}_t^{\mathrm{KF}} = \left(\begin{smallmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{smallmatrix}\right)$, $\boldsymbol{H}_t^{\mathrm{MKF}} = \boldsymbol{I}_2$ and $\boldsymbol{R}_t = 0.16 \cdot \boldsymbol{I}_2$.

Beside the ground truth, Fig. 4 displays the full posterior Gaussian densities of the two filters on the example of target 1. Here, the KF is much less certain about the target position than the MKF whose covariance is only large at the beginning. Moreover, the KF trajectory has a noticeable overshoot at the turns while the MKF follows the ground truth accurately.

Fig. 5(a) and 5(b) display the filters' performance in terms of the Root Mean Square Error (RMSE) over 100 Monte Carlo (MC) runs under different probabilities of detection $p_{\mathrm{d}} \in \{0.4, 0.8, 1.0\}$; the target is detected in the beginning in both cases to ensure proper initialisation. The MKF barely shows any increase in error for decreasing $p_{\mathrm{d}}$ and it is not affected by the turns in either trajectory since it learned where they occur based on the respective start point A or B. The KF, however, suffers greatly under low detection rates, especially in the areas around the turns. Fig. 5(c) and 5d show the RMSE over 100 MC runs for $p_{\mathrm{d}} = 1.0$ in case the target is either detected or occluded in the hatched area in Fig. 4. The overshoot of target 1 after the manoeuvre and the turn of target 2 result in a sharp error increase for the KF during the occlusion. The MKF, on the other hand, can bridge the missing detections because it learned temporal dependencies across the full track history, leading to almost no loss in performance.




(a) Varying $p_{\mathrm{d}}$, trajectory 1. (b) Varying $p_{\mathrm{d}}$, trajectory 2.

(c) Occlusion, trajectory 1. (d) Occlusion, trajectory 2.

Fig. 5. Exp. 2: Performance of KF(—) and MKF(—) with different detection rates or under occlusion.

## V. Conclusion

This work presented the Mnemonic Kalman Filter, a variation of the Kalman Filter which incorporates arbitrarily non-linear, non-Markovian dynamic models in the prediction step. To ensure that the predicted density is still Gaussian, a neural network architecture based on Long Short-Term Memory was proposed which constructs a multivariate normal distribution from a given input state estimate but also with a dependence on previous estimates. Simulations showed that the proposed filter performs much better in the presence of occlusions or low detection rates in comparison to the classic Kalman Filter since it has learned long-time temporal dependencies. It can further predict a full probability density which reflects both the target state and uncertainty in the underlying complex target behaviour. Future work shall provide a more thorough analysis in less controled environments and explore the potential of MD-LSTM networks for multi-target tracking.

## REFERENCES


[1] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME–J. Basic Eng.*, vol. 82, pp. 35–45, 1960.

[2] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," in *Proc. Signal Process., Sensor Fusion, Target Recognit. VI*, 1997, pp. 182–193.

[3] S. Coraluppi, C. Carthel, P. Braca, and L. Millefiori, "The mixed Ornstein-Uhlenbeck process and context exploitation in multi-target tracking," in *Proc. 19th Int. Conf. Inf. Fusion*, 2016, pp. 217–224.

[4] S. M. Pandit and S.-M. Wu, *Time Series and System Analysis With Applications*, vol. 3. New York, NY, USA: Wiley, 1983.

[5] M. Alam, M. D. Samad, L. Vidyaratne, A. Glandon, and K. M. Iftekharuddin, "Survey on deep neural networks in speech and vision systems," 2019, *arXiv:1908.07656*.

[6] M. I. Razzak, S. Naz, and A. Zaib, "Deep learning for medical image processing: Overview, challenges and the future," in *Classification in BioApps*. Berlin, Germany: Springer, 2018, pp. 323–350.

[7] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2013, pp. 6645–6649.

[8] L. H. Son, A. Allauzen, and F. Yvon, "Continuous space translation models with neural networks," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Lang. Technol.*, 2012, pp. 39–48.

[9] R. Socher, A. Karpathy, Q. V. Le, C. D. Manning, and A. Y. Ng, "Grounded compositional semantics for finding and describing images with sentences," *Trans. Assoc. Comput. Linguistics*, vol. 2, pp. 207–218, 2014.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Commun.*, vol. 9, pp. 1735–80, Dec. 1997.

[11] H. Coskun, F. Achilles, R. DiPietro, N. Navab, and F. Tombari, "Long short-term memory Kalman filters: Recurrent neural estimators for pose regularization," in *Proc. IEEE Int. Conf. Comput. Vision*, 2017, pp. 5524–5532.

[12] D. Iter, J. Kuck, P. Zhuang, and C. Learning, "Target tracking with Kalman filtering, KNN and LSTMs," 2016, pp. 1–7.

[13] M. Emambakhsh, A. Bay, and E. Vazquez, "Deep recurrent neural network for multi-target filtering," in *Proc. Int. Conf. Multimedia Model.*, 2019, pp. 519–531.

[14] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 4225–4232.

[15] M. Emambakhsh, A. Bay, and E. Vazquez, "Convolutional recurrent predictor: Implicit representation for multi-target filtering and tracking," *IEEE Trans. Signal Process.*, vol. 67, no. 17, pp. 4545–4555, Sep. 2019.

[16] S. Jung, I. Schlangen, and A. Charlish, "Sequential Monte Carlo filtering with long short-term memory prediction," presented at the 22nd Int. Conf. Inf. Fusion, Ottawa, ON, Canada, 2019.

[17] P. Williams, "Using neural networks to model conditional multivariate densities," *Neural Comput.*, vol. 8, pp. 843–54, 1996.

[18] A. M. Schäfer and H. G. Zimmermann, "Recurrent neural networks are universal approximators," in *Artificial Neural Networks – ICANN*, S. D. Kollias, A. Stafylopatis, W. Duch, and E. Oja, Eds. Berlin, Germany: Springer, 2006, pp. 632–640.

[19] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I: Dynamic models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333–1364, Oct. 2003.