# Augmented State Neural-Transition Extended Kalman Filter

Seungcheol Oh, Seonggyu Jung, and Joongheon Kim
Department of Electrical and Computer Engineering, Korea University, Seoul, Republic of Korea
E-mails: {seungoh, rheotodtjdrb, joongheon}@korea.ac.kr

***Abstract*—We propose the Augmented State Neural-Transition EKF (ANT-EKF), a hybrid filtering framework that embeds a learnable neural transition model inside the extended Kalman filter. By augmenting the state with the neural network's parameters, ANT-EKF jointly tracks the physical state and adapts the dynamics model online using Jacobians obtained via autograd. We instantiate the method for (i) object trajectory tracking and (ii) RIS-assisted mobile channel tracking, where the measurement remains linear in the real-stacked channel. Simulations show that ANT-EKF achieves accurate tracking and reduces NMSE compared with EKF variants relying on fixed, hand-crafted dynamics.**

***Index Terms*—Extended Kalman Filter, Deep Neural Network**

## I. Introduction

The Kalman filter (KF) stands as a cornerstone of modern estimation theory as it provides a powerful and efficient recursive solution to the problem of inferring the latent state of a dynamic system from noisy measurements [1]. The extended Kalman filter (EKF) further broadens this applicability to non-linear, second-order systems [2], [3]. However, the celebrated optimality of these filters hinges on a critical assumption that a precise mathematical model governing the system's dynamics is perfectly known beforehand. The accuracy of this ideal model significantly affects the performance and stability of the EKF [4].

In practice, the system model is often a hand-crafted simplification, such as a constant velocity or constant acceleration model, which is assumed to be known and fixed throughout the tracking process. This approach creates a significant mismatch with the complex and highly nonlinear dynamics of real-world mobile channels and can often lead to filter divergence [5]. To bridge this gap, recent research has developed towards hybrid approaches that fuse data-driven approach with EKF [6], [7]. In particular, previous work leveraged DNN [6] and a long-short term memory (LSTM) [7] to learn complex mobility patterns from data, and used the predictions to assist a separate KF in the tracking task. While these data-driven approaches have shown improved performance, they operate on pre-trained, static model of the environment, which leaves the system vulnerable to environmental dynamics not encountered during its offline training phase [8], [9].

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2024-RS-2024-00436887) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation); and also by IITP grant funded by MSIT (RS-2024-00439803, SW Star Lab). *(Corresponding author: Joongheon Kim)*

To overcome the limitations of the conventional aforementioned offline design methods, we propose an augmented state neural-transition EKF (ANT-EKF), a novel EKF-based framework that enables online learning of the system dynamics. The core innovation of our work lies in augmenting the state vector of the EKF. Specifically, instead of only tracking the physical state of the system, we expand the state to include the parameters, the weights and biases of the neural network, that models the unknown state transition function. With each measurement update, the filter not only refines its estimate of the physical state but also simultaneously updates and adapts the parameters of the internal dynamics model. This allows the filter to learn and adapt to the complex, time-varying dynamics of the environment in real-time, without requiring any prior training or a predefined motion model. In this paper, we apply this novel approach to the object trajectory tracking and wireless channel tracking problems and show that ANT-EKF is suitable for producing significantly accurate results.

## II. Augmented State Neural-Transition EKF

The KF provides an optimal solution for linear systems, and its application to the nonlinear dynamics is dealt with EKF by approximating the nonlinear system locally with a linear one at each time step. This is achieved through the computation of Jacobian matrices, which linearize the nonlinear transition and observation functions in a neighborhood of the current estimate. The ANT-EKF extends the EKF framework to incorporate learnable dynamics. Instead of relying solely on hand-crafted models, ANT-EKF augments the state with the parameters of a neural network that governs the unknown part of the system dynamics. These parameters are updated online as part of the recursive filtering process and enable the model to adapt to complex and time-varying behaviors. Specifically, in ANT-EKF, the state is augmented with neural network parameters as $\boldsymbol{x}_k = [\boldsymbol{s}_k, \boldsymbol{\theta}_k]^\top$, where $\boldsymbol{s}_k$ denotes the physical system state and $\boldsymbol{\theta}_k$ the learnable parameters. The transition model is expressed as $\boldsymbol{x}_{k+1} = \Phi_{\boldsymbol{\theta}_k}(\boldsymbol{x}_k) + \boldsymbol{w}_k$, with process noise $\boldsymbol{w}_k \sim \mathcal{N}(0, Q)$, where $\Phi_{\boldsymbol{\theta}}(\cdot)$ is a nonlinear mapping parameterized by $\boldsymbol{\theta}$. The measurement model follows $\boldsymbol{y}_k = h(\boldsymbol{x}_k) + \boldsymbol{v}_k$, with measurement noise $\boldsymbol{v}_k \sim \mathcal{N}(0, \boldsymbol{R})$.

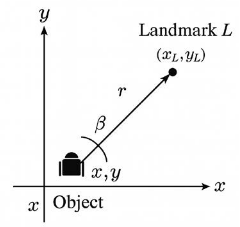


Fig. 1: Moving object and measurement landmark.

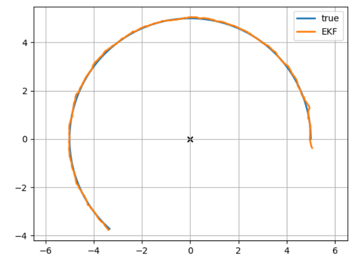


Fig. 2: True and estimated trajectory of the object.

## III. Application I: Object Trajectory Tracking

### A. System Model

Consider a mechanical object moving along a circular trajectory given by

$$x(t) = x_c + R\cos\omega t, \tag{1}$$
$$y(t) = y_c + R\sin\omega t, \tag{2}$$

where $(x_c, y_c)$ are the center coordinates, $R$ is the radius and $\omega$ is the angular velocity. The velocity of the object is obtained as

$$\boldsymbol{v}(t) = \dot{\boldsymbol{p}}(t) = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} -R\omega\sin\omega t \\ R\omega\cos\omega t \end{bmatrix}. \tag{3}$$

We introduce a nonlinear function $f_{\boldsymbol{\theta}} : \mathbb{R}^2 \to \mathbb{R}^2$ that captures the unknown dynamics of the object, mapping the position $p$ to a velocity term. The mapping is parameterized by a DNN with its parameters $\boldsymbol{\theta}$, which is a collection of all weights and biases of the network. We enable online training by augmenting the parameters $\boldsymbol{\theta}$ into the system state. The augmented state is thus defined as $\boldsymbol{x} = [\boldsymbol{p}, \boldsymbol{\theta}]^\top \in \mathbb{R}^{2+n_\theta}$, where $\boldsymbol{p} = [x, y]^\top$ and $n_\theta$ denotes the object's position and total number of parameters, respectively. Therefore, the $k+1$-th state evolves from $k$-th state according to

$$\boldsymbol{x}_{k+1} = \underbrace{\begin{bmatrix} \boldsymbol{p}_k + f_{\boldsymbol{\theta}_k}(\boldsymbol{p}_k)\Delta t \\ \boldsymbol{\theta}_k \end{bmatrix}}_{\Phi(\boldsymbol{x}_k)} + \boldsymbol{w}_k, \tag{4}$$

where $\boldsymbol{w}_k \sim \mathcal{N}(0, \boldsymbol{Q})$ and $Q = \text{diag}(\sigma_p^2 \boldsymbol{I}_2, \sigma_\theta^2 \boldsymbol{I}_{n_\theta})$ and $\Delta t$ is the sampling interval between $k$ and $k+1$. Here, the function mapper, parameterized by DNN is formally described as

$$f_{\boldsymbol{\theta}}(\boldsymbol{p}) = W_L\sigma(W_{L-1}\sigma(\ldots\sigma(W_1\boldsymbol{p} + b_1)) + b_{L-1}) + b_L, \tag{5}$$

where $\sigma(\cdot) = \tanh(\cdot)$.

As illustrated by Fig. 1, a measuring device at landmark $(x_L, y_L)$ provides measurement expressed as $\boldsymbol{y}_k = h(\boldsymbol{x}_k) + \boldsymbol{v}_{\boldsymbol{k}}$, where $\boldsymbol{v}_k \sim \mathcal{N}(0, \boldsymbol{R})$, where $h(\boldsymbol{x}) = [r, \beta]^\top$, $r = \sqrt{(x - x_L)^2 + (y - y_L)^2}$, $\beta = \text{atan}^2(y - y_L, x - x_L)$, and $R = \text{diag}(\sigma_r^2, \sigma_\beta^2)$. Since $h(\cdot)$ is a nonlinear function, we find its Jacobian such that the EKF takes advantage of the local linearity. The Jacobian is described as

$$\boldsymbol{H}_k = \begin{bmatrix} \partial r/\partial \boldsymbol{p} & \boldsymbol{0}_{1\times n_\theta} \\ \partial\beta/\partial\boldsymbol{p} & \boldsymbol{0}_{1\times n_\theta} \end{bmatrix}, \tag{6}$$

where $\dfrac{\partial r}{\partial \boldsymbol{p}} = \dfrac{1}{r}[x - x_L, y - y_L]$, and $\dfrac{\partial \beta}{\partial \boldsymbol{p}} = \dfrac{1}{r^2}[-(y - y_L), x - x_L]$. The complete EKF algorithm for object tracking is then described by Algorithm 1.

**Algorithm 1:** ANT-EKF for Object Tracking

**Input:** Initial state $\hat{\boldsymbol{x}}_{0|0}$, covariance $\boldsymbol{P}_0$, process noise $\boldsymbol{Q}$, measurement noise $\boldsymbol{R}$

**Output:** Estimated states $\hat{\boldsymbol{x}}_{k|k}$ and covariances $\boldsymbol{P}_{k|k}$

**Initialization:** Set $k = 0$.
**repeat**
  **Prediction:** $\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{\Phi}(\hat{\boldsymbol{x}}_{k-1|k-1})$,
  $\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k\boldsymbol{P}_{k-1|k-1}\boldsymbol{F}_k^\top + \boldsymbol{Q}$, where $\boldsymbol{F}_k = \dfrac{\partial\boldsymbol{\Phi}}{\partial\boldsymbol{x}}$.
  **Update:**
  Innovation: $\hat{\boldsymbol{y}}_k = h(\hat{\boldsymbol{x}}_{k|k-1})$.
  $\boldsymbol{S}_k = \boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^\top + \boldsymbol{R}$.
  $\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^\top\boldsymbol{S}_k^{-1}$.
  $\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k[\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k]$.
  $\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_{k|k-1}$.
  Set $k \leftarrow k + 1$.
**until** *stopping criterion is reached*;

### B. Experiment

Fig. 2 compares the true two-dimensional trajectory (blue) with the ANT-EKF estimate (orange) for the range-and-bearing sensor located at the center. The two curves are nearly indistinguishable along the arc, indicating that after a brief initialization the filter locks onto the motion and displays very small position error. This is because the learned transition provides a good prior each step, while the EKF update corrects any drift and keeps the estimate on the circular path.

## IV. Application II: Mobile Channel Tracking

### A. System Model

Consider an RIS-assisted SISO system with a single basestation (BS) and a single user equipment (UE) given as Fig 3. The RIS with $N$ cells forms a reflection link between BS and UE by controlling the phase of the reflected signals. We use $\boldsymbol{G}^{(t)} \in \mathbb{C}^N$ and $\boldsymbol{h}_r^{(t)} \in \mathbb{C}^N$ to denote the channel between BS and RIS, and between RIS and UE, respectively. The channel function $\boldsymbol{h}^{(t)}$ is defined as $\boldsymbol{h}^{(t)} = \boldsymbol{G}^{(t)}\text{diag}(\boldsymbol{\nu}^{(t)})\boldsymbol{h}_r^{(t)} \in \mathbb{C} =$

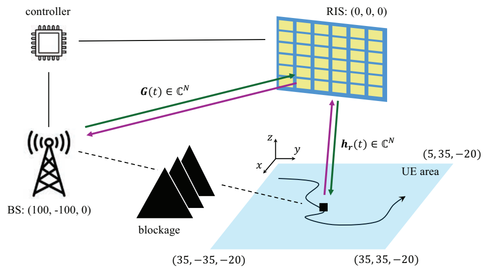


Fig. 3: Overall RIS assisted channel tracking system.

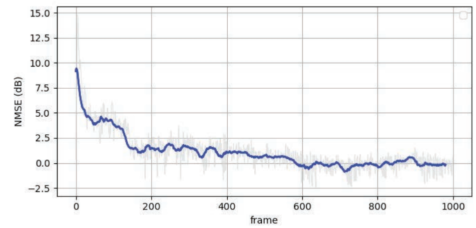


Fig. 4: Normalized mean squared error (dB).

$(\boldsymbol{\nu}^{(t)})^\top \boldsymbol{h}_c^{(t)}$, where $\boldsymbol{\nu}^{(t)}$ is the RIS reflection coefficient and $\boldsymbol{h}_c^{(t)}$ is the cascaded channel between BS and UE.

In the Cartesian coordinate system shown in Fig. 3, the BS and RIS are located at $(100, -100, 0)$ and $(0, 0, 0)$, respectively. A single UE position follows a curved trajectory which is defined by the polar coordinate system above the UE area. For the exact channel modeling based on Rician fading, we followed the model of [10]. Define $\tilde{\boldsymbol{h}} = [\Re\{\boldsymbol{h}\}, \Im\{\boldsymbol{h}\}]^\top \in \mathbb{R}^{2N}$, then, the augmented state is defined as $\boldsymbol{x} = [\tilde{\boldsymbol{h}}, \boldsymbol{\vartheta}]^\top \in \mathbb{R}^{2N+M_\vartheta}$, where $\boldsymbol{\vartheta}$ is the parameter of the function mapper defined in the state transition described as

$$\boldsymbol{x}_{k+1} = \underbrace{\begin{bmatrix} \tilde{\boldsymbol{h}}_k + f_{\boldsymbol{\vartheta}_k}(\tilde{\boldsymbol{h}}_k) \\ \boldsymbol{\vartheta}_k \end{bmatrix}}_{\boldsymbol{g}(\boldsymbol{x})} + \boldsymbol{w}_k, \qquad \boldsymbol{w}_k \sim \mathcal{N}(0, \boldsymbol{Q}). \tag{7}$$

The measurement is taken via pilot signal from the user to the BS described as $\boldsymbol{z}_k = \sqrt{P_u}\,\boldsymbol{A}(\boldsymbol{\nu}_k)\,\tilde{\boldsymbol{h}}_k + \boldsymbol{v}_k$, where $\boldsymbol{v}_k \sim \mathcal{N}(0, \boldsymbol{R}) \in \mathbb{R}^2$ and $\boldsymbol{A}(\boldsymbol{\nu}) = \begin{bmatrix} \Re\{\boldsymbol{\nu}^\top\} & -\Im\{\boldsymbol{\nu}^\top\} \\ \Im\{\boldsymbol{\nu}^\top\} & \Re\{\boldsymbol{\nu}^\top\} \end{bmatrix} \in \mathbb{R}^{2\times 2N}$. Algorithm 2 describes the rest of the EKF procedure for this application.

---

**Algorithm 2:** ANT–EKF for Mobile Channel Tracking

---

**Input:** $P_u,\ Q,\ R,\ \hat{\boldsymbol{x}}_{0|0},\ \boldsymbol{P}_{0|0},\ \{\boldsymbol{\nu}_k, \boldsymbol{z}_k\}_{k=1}^T$

**Output:** $\{\hat{\boldsymbol{x}}_{k|k}, \boldsymbol{P}_{k|k}\}_{k=1}^T$

**Initialization:** Set $k \leftarrow 1$.

**repeat**

- **Prediction:** $\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{g}(\hat{\boldsymbol{x}}_{k-1|k-1})$, $\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^\top + Q$, where $\boldsymbol{F}_k = \begin{bmatrix} \boldsymbol{I}_{2N} + \boldsymbol{J}_{\boldsymbol{h}} & \boldsymbol{J}_{\boldsymbol{\vartheta}} \\ \boldsymbol{0} & \boldsymbol{I}_{M_\vartheta} \end{bmatrix}$,
- $\boldsymbol{J}_{\boldsymbol{h}} = \frac{\partial f_{\boldsymbol{\vartheta}}(\tilde{\boldsymbol{h}})}{\partial \tilde{\boldsymbol{h}}}\Big|_{\hat{\boldsymbol{x}}_{k-1|k-1}}, \boldsymbol{J}_{\boldsymbol{\vartheta}} = \frac{\partial f_{\boldsymbol{\vartheta}}(\tilde{\boldsymbol{h}})}{\partial \boldsymbol{\vartheta}}\Big|_{\hat{\boldsymbol{x}}_{k-1|k-1}}$.
- **Update:** Innovation: $\hat{\boldsymbol{z}}_{k|k-1} = \boldsymbol{h}(\hat{\boldsymbol{x}}_{k|k-1}) = \boldsymbol{H}_k\,\hat{\boldsymbol{x}}_{k|k-1}$, with $\boldsymbol{H}_k = \left[\sqrt{P_u}\,\boldsymbol{A}(\boldsymbol{\nu}_k)\ \ \boldsymbol{0}_{2\times M_\vartheta}\right]$, $\boldsymbol{r}_k = \boldsymbol{z}_k - \hat{\boldsymbol{z}}_{k|k-1}$,
- $\boldsymbol{S}_k = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^\top + R$,
- $\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^\top \boldsymbol{S}_k^{-1}$.
- $\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \boldsymbol{r}_k$,
- $\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \boldsymbol{P}_{k|k-1}$
- Set $k \leftarrow k + 1$.

**until** $k > T$

---

### B. Experiment

Fig. 4 shows the per-frame NMSE (gray) and its moving average (blue) during RIS-assisted channel tracking with ANT-EKF. The error starts high in the initial transient, when the filter has little information about the cascaded channel and the neural transition parameters, but drops quickly as pilot measurements are assimilated. After 100–200 frames the curve flattens and continues a slow descent toward  0 to 1 dB, indicating sub-dB estimation error relative to channel power. The small oscillations reflect measurement noise, pilot-phase diversity, and natural channel fluctuation, while the absence of upward drift shows the filter/parameter updates remain stable as the model adapts online.

## V. Conclusion

We introduced the ANT-EKF, which jointly estimates the system state and a learnable transition model by augmenting the EKF state with neural-network parameters. Experiments on object tracking and RIS-assisted channel tracking show accurate trajectory estimates and steadily decreasing NMSE. Future work will extend ANT-EKF to more complex dynamical settings.

## References


[1] R. E. Kalman, "A new approach to linear filtering and prediction problems," 1960.

[2] G. Welch, G. Bishop *et al.*, "An introduction to the kalman filter," 1995.

[3] S. Haykin, *Kalman filtering and neural networks*. John Wiley & Sons, 2004.

[4] L. Ljung, "Asymptotic behavior of the extended kalman filter as a parameter estimator for linear systems," *IEEE Transactions on Automatic Control*, vol. 24, no. 1, pp. 36–50, 2003.

[5] S. Singhal and L. Wu, "Training multilayer perceptrons with the extended kalman algorithm," *Advances in neural information processing systems*, vol. 1, 1988.

[6] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[8] S. H. Lim, S. Kim, B. Shim, and J. W. Choi, "Deep learning-based beam tracking for millimeter-wave communications under mobility," *IEEE Transactions on Communications*, vol. 69, no. 11, pp. 7458–7469, 2021.

[9] J. Yu, X. Liu, Y. Gao, C. Zhang, and W. Zhang, "Deep learning for channel tracking in irs-assisted uav communication systems," *IEEE Transactions on Wireless Communications*, vol. 21, no. 9, pp. 7711–7722, 2022.

[10] H. Han, T. Jiang, and W. Yu, "Active beam tracking with reconfigurable intelligent surface," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.