# CS5340: Human Crowd Modelling with Kalman Filter and its variants

**Tu Weile** **Cui Xiuqun** **Nguyen Quoc Anh** **Marcus Yeo Rui Han** **Yap Wei Xuan**
A0265031R A0168706W A0274968E A0261964W A0183226H
National University of Singapore, School of Computing
{tuweile, xiuqun, e1124714, e0983358, e0310021}@u.nus.edu

## Abstract

In this project, our team proposes the exploration and investigation of several probabilistic models within the domain of crowd modeling, simulation, and object tracking. This includes the implementation and application of the naive Kalman filter and its other variants such as the non-linear Kalman filter and particle filter for evaluation of its effectiveness in prediction within crowd modeling.

## 1 Introduction

In the 2000s and early 2010s, probabilistic models became more prominent in crowd modelling, notably, Kalman filters. Kalman filters are utilised for predicting and updating the state of a system, given noisy sensor measurements, specifically in the domain of crowd modelling, Kalman filters are known to provide a reliable approach for tracking pedestrian motion, enabling accurate trajectory and velocity prediction. The methodology of the Kalman filter relies on a combination of prediction (based on the dynamic model) and model correction (using observed / sensor data), allowing for effective handling of uncertainties and noise in the environment.

In this project, our team have chosen to move in the direction of probabilistic models, specifically Kalman filters and Particle filters, exploring how each performs in the task of moving pedestrian trajectory prediction. **Note** that all figures/tables/graphs mentioned in the report are in Section **5**.

## 2 Approach

### 2.1 Dataset

The dataset used is the EWAP dataset, which consists of the captured positions and velocities of each pedestrian crossing (Pellegrini et al., 2009). This sequence was shot at 25 fps. The annotation, however, was done at 2.5 fps. We will assume that 1) the EWAP data is the ground truth for pedestrians' positions and velocities in the 2D X-Y plane, and 2) the sensors and kinematic system noise (e.g. from friction, air resistance) follow Normal Gaussian distributions.

### 2.2 Multivariate Kalman Filter

This section will go into detail about the design and implementation of the baseline multivariate **Kalman Filter (KF)** for pedestrian trajectory prediction, and the evaluation and optimization of the model.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

#### 2.2.1 Methodology

The observed and hidden variables defined for this model are as follows: 1) **Observed (X):** the noisy position and velocity values received from the sensor, and 2) **Hidden (Z):** actual ground truth of the pedestrians' position and velocity values. Figure 1 illustrates the methodology of this project.

#### 2.2.2 Kalman Filter Design - Model Parameters setup

For the modeling of the pedestrian path, the velocity equation will be used where:

$$\text{Position}_t = \text{Position}_{t-1} + \text{Velocity}_{t-1} \times \Delta t$$

This is followed by the Kalman Filter model in Figure 2, which showcases the Kalman Filter model as a Markov chain across the time steps $t$.

Given the above figures, the KF model parameters are defined in the Figure 3 below.

#### 2.2.3 Designing Kalman Filter Baseline Model

Unlike $F$, $B$ and $H$ as defined in Figure 3, $Q$ and $R$ are dependent on the environment and sensor equipment used. For $R$, the sensor's position and velocity's standard deviation (stddev) are assumed to be $0.3$m and $0.45$m respectively, which is reasonable given that the uncertainties of a sensor can be obtained from its manufacturer's specifications in reality. For $Q$, the X and Y axis position stddev are calculated from the difference between the predicted positions with the velocity equation and the actual positions at each time step. The X and Y axis velocity stddev are calculated from the difference between the X and Y axis velocities in the current and next time step. The aforementioned stddev in $Q$ are calculated across the entire noisy dataset for all pedestrians. The Euclidean Distance Position and Velocity RMSE of the baseline model over every pedestrian in each time step is represented in Figure 4 and Figure 5.

#### 2.2.4 Challenges in Classical Kinematic Variable of Acceleration

With our dataset comprising of both position and velocity, one might argue for augmenting our dataset with acceleration to better represent our model, but given the formulas of the predict and update steps in the Kalman filter, such an assertion would require us to imply values of acceleration for each time step in our dataset; an impracticality given the difficulty of ascertaining acceleration at that specific time step without consideration of the rest of the time. Furthermore, as acceleration is also influenced by gravity, drag, and air resistance, the computation of acceleration becomes intractable with our dataset.

### 2.3 Non-linear Kalman Filter

While the linear Kalman filter is useful in modeling human crowd predictions, non-linearity can exist within the process model. Take the example of acceleration, where it can be expressed as $\ddot{x} = \frac{0.034ge^{-x/22000}\dot{x}^2}{2\beta} - g$, such an equation comprising of multiple factors such as drag and gravity is expressed as a non-linear differential equation.

To resolve such a non-linear equation, the decomposition of the non-linear equation through **linearization** ($Ax = b$) into sets of linear equations is required before computation of an approximate solution. This endeavour, while useful in its own merit, will also mean that the linearization of a non-linear problem will always produce inexact answers due to the use of approximations and errors within our calculations, which would therefore cause the issue of **filter divergence**. To implement this, either the extended Kalman filter (EKF) or the unscented Kalman filter (UKF) can be used.

#### 2.3.1 Unscented Kalman Filter (UKF) and Extended Kalman Filter (EKF)

EKF resolves the issue of non-linearity by linearizing our non-linear functions around the current estimate with the **Jacobian matrix** through first-order linearization, whereas UKF solves the issue of non-linearity through the use of sigma points around the mean which are then propagated through the non-linear functions, capturing the true mean and covariance of the state estimate with the second order Taylor series expansion for non-linearity. While both methodologies are valid, for the purposes of this experiment, the UKF will solely be used as the non-linear Kalman filter.

The rationale for such a move can be attributed to the implementation of Jacobian matrices and its reliance on first-order linearization. As good domain knowledge of the model is required to derive the appropriate Jacobian matrices, complex models may render such calculations analytically intractable and error-prone. Furthermore, EKF may not necessarily capture the true model if higher-order terms have a significant impact. This means that a closed form solution for EKF is difficult, which would then require us to use iterative solutions to address such non-linearities.

#### 2.3.2 Unscented Kalman filter methodology

The methodology of the unscented Kalman filter (Labbe, 2014) expands on the linear Kalman filter by introducing a set of sigma points (Van Der Merwe, 2004) to propagate through the non-linear system, allowing the UKF to approximate the mean and covariance more accurately than EKF. This is achieved through the passing of the state $f()$ and measurement $h()$ functions along the sigma point object, which does not differ much from our standard linear Kalman filter, which likewise passes these functions as matrices $F$ and $H$ respectively. For our UKF, non-linear functions $f()$ and $h()$ are supplied in lieu of these state and measurement functions respectively.

### 2.4 Particle Filter

Although our models thus far are capable of producing optimal estimates for unimodal linear and non-linear, continuous, multivariate problems, nevertheless a severe limitation our team observed is that these models are unable to handle occlusions well and are not multimodal by nature, especially as given from our dataset (Pellegrini et al., 2009). While UKF can handle non-Gaussian noise as well, it should be noted that UKF can only handle such noise modestly due to its assumption of Gaussian noise in the process and measurement models. This is where particle filters steps in: it is flexible in **handling non-Gaussian noise** and **multimodal non-linear distributions** effectively.

#### 2.4.1 Particle Filter methodology

In particle filters, the core tenet is to represent the probability distribution of possible states through a set of particles that is generated from the **Monte Carlo (MC) method**. Each particle represents its possible position, heading and weight that indicates its likelihood, given the observed data. While these particles initialize with the same weight, the next state of the particles will be predicted based on the behaviours of the system before updating the weightage of the particles from measurement.

The weightage of the particles is then updated: particles that closely match the measurements are weighed higher than those that do not. The particles are then resampled periodically in order to focus on particles and regions with a higher likelihood, before computing the weighted mean and covariance to obtain an estimate. This helps in improving the overall accuracy of the estimations.

### 2.5 Evaluation Methods

To evaluate these approaches, two primary methods of evaluation are used: **Euclidean Distance**, and **RMSE metrics**. Euclidean Distance (ED) measures the direct distance between two points on a plane. This is relevant as it can evaluate the difference between predicted velocities and positions and ground truth velocities and positions on a individual per-prediction basis. Building further on this concept, the RMSE metric offers a deeper analysis through the square root of the mean of squared errors (Euclidean Distance errors) across the filter's predictions. At each step, the RMSE would represent the overall prediction quality of the filter for all preceding steps, up to that step itself. Likewise, the stepwise RMSE of the filter is plotted and analyzed. Together, both complementary metrics are used to discuss and analyze the linear Kalman filter's performance in predicting pedestrian movement, with focus on both individual errors and overall accuracy trends.

# 3 Experiments and Results

## 3.1 Kalman Filter

### 3.1.1 Kalman Filter - State and State Covariance Matrix Initialization

Initialization of hyperparameters is one of the most important aspects in modeling. First, two sets of states and state covariance matrices are experimented. The first experiment, named naive_init, initialized the state from naive assumptions: Assuming every pedestrian's initial position can be anywhere, which is at the expectation of a uniform distribution consisting of all data points, and their initial starting velocity is a normal distribution with mean being a human's average velocity (1.34112m/s) and a standard deviation of an (arbitrary) $\frac{1}{3}$ of the mean. The second experiment, named stat_init, inferred these statistics from the observations. Each pedestrian's initial state would be their first observed position and velocity, and the covariance matrix is initialized with the variances of observed states (A diagonal matrix with the state variances as the diagonal elements).

After evaluating with the full dataset, the naive_init's RMSE is found to be $2.108$, while that of stat_init is $1.866$, an $11.5\%$ decrease in error. KF is robust against initialization, so it is observed the covariances in both cases decreased and plateaued equally quickly. Thus, it can attribute such a decrease in error to the massive error in the first step of naive_init, where it assumes that the pedestrian starts in the middle of the field, before adjusting their position after the first observation.

### 3.1.2 Kalman Filter - Optimisation

With the measurement noise assumed to be given in a real-world setting, $Q$ has to be optimized to represent the environment the KF model is being used as closely as possible. The optimization process consists of 2 steps: **sensitivity analysis** (Vipond, 2023) (to determine the order of each tunable parameter's significance in changing the evaluation criteria) and **step-wise regression** (Fritz and Berger, 2015) (to iteratively converge to the optimal set of parameters). For the process noise matrix, the 4 tunable parameters are the environment x position, y position, x velocity, and y velocity's stddev. Figure 6 shows the sensitivity analysis results whereby the change in the sum of the mean Euclidean Distance position/velocities Errors across the entire dataset ($ED_{Sum}$) when each parameter changes by $\pm 10\%$. Figure 7 shows the convergence process of the step-wise regression following the iterative order shown in Figure 6's conclusion to reach an $ED_{Sum}$ value of **0.560**.

Figure 4 and 5 show the performance of the KF model using the optimized Q over the whole video for every pedestrian per each time step. From the final RMSE values in Figures 4 and 5, it is observed that the **optimized** KF Model **Position** ED RMSE **(0.307m)** is **27.6**$\%$ lower than that of the **noisy** data set position RMSE **(0.424m)** and is also **11.3**$\%$ lower than that of the **baseline** KF model position RMSE **(0.346m)**. Moreover, the **optimized** KF Model **Velocity** ED RMSE **(0.340m/s)** is **46.4**$\%$ lower than that of the **noisy** data set velocity RMSE **(0.634m/s)** and is also **30.8**$\%$ lower than that of the **baseline** KF model velocity RMSE **(0.491m/s)**. This significant reduction in ED error between the model's predicted position/velocity values and the ground truth data shows the effectiveness of the multivariate KF model in object tracking even with a noisy sensor.

### 3.1.3 Kalman Filter - Missing Random Data

Testing the robustness of KF against missing observations is important in pedestrian tracking because often the path is riddled with occlusions, sensor failures, and noise. To test this, our team compared the performance of a KF running on full observations, and a KF running on observations that is missing parts of its data. Our team hopes to simulate cases where there should be observations in these specific timestamps, but for one reason or another, those data points are deemed corrupted or missing. Implementation-wise, this means for random indices, the KF would predict but not be updated with observations, and thus the next estimated state would purely be inferred with the transition matrix and estimation from the last timestep.

Our team then evaluate the 2 cases on the full dataset, with **20-90%** of the observations missing. From this, our team finds that KF is effective for this experiment, with only **1%** more error compared to full KF with **20%** of the observations missing. Even with 90% of the data missing, only **32%** more error was observed. This robustness can be attributed to the fact that in this simple scene, most pedestrians mainly go in a roughly straight line, with little variance in velocity or few jerks in movement. This trait makes a lot of the accurate estimation probable even with just equations of motion. Our team also observed particular cases where the KF with missing observations performed better than full KF, which can be due to the noise itself giving KF erroneous signals that deviate from the ground truth. Therefore ignoring such noise and just assuming the person walking in a straight line would decrease the error. However, our team expects that, with the same modeling methods and a different environment that causes pedestrians to constantly adjust their path, KF would not be so effective against missing observations. Results and visualization can be seen in Table 1 and Figure 8.

#### 3.1.4 Kalman Filter - Missing Consecutive Data

With the model optimised on process noise, our team evaluates the model's performance again on missing data, but this time, the set of missing observations was consecutive. In doing so, the aim is to stress-test the filter's capabilities in accurately predicting position and velocity even despite having long step-periods of missing data. Our team noted that the performance was impacted more significantly. With data omitted from the final 15% of the steps, the filter performed worse compared to KF inference on full data:

**(1) Position RMSE drop:** $50.4\%$ **(2)Velocity RMSE drop:** $8.4\%$ **(3) Overall RMSE drop:** $28.86\%$

The sharp drop in RMSE across the board proves that the KF indeed performs poorly when large portion of data is missing. This can be explained by the KF's fundamental reliance on the Markov assumption, where the future state of a process relies solely on the current state. With 3.1.3, the steps where data is withheld from the filter are randomized and the filter is still able to rely on its current state's provided data, when available. As such, the effects of missing data are not as pronounced, evident in Table 1. As such, when presented with a scenario where long sequences need to be predicted with complete loss of data, the filter is essentially forced to predict 'blindly' for a prolonged period. Therefore, our team conclude that while robust if reliable sensor data is present, the filter is not adept at independent prediction for too long at once and indeed has a heavy reliance on sensor data. If the observation data itself is too noisy with too many occlusion, this can adversely impact the filter's performance.

### 3.2 Unscented Kalman Filter

For the implementation of the UKF for our dataset, our experiment involves utilizing the sensor's position and velocity uncertainties from the linear Kalman filter configurations along with the covariance matrix. From this, experimental observations show that the UKF was able to predict the next state accurately, albeit with increasing uncertainty due to the increasing covariance for each time step as shown in Figure 9, where the grey shaded ovals denote the covariance of the prediction in that particular time step. Such experimental observations can be attributed to the increasing uncertainty during prediction as UKF propagates the state and its covariance forward in every prediction step.

Moving onto the evaluation metrics, from Figure 10, our team observes that the **RMSE value for UKF is at 0.061m**, which is comparatively better than the linear KF methodologies that our team has implemented above. Our team also observes a decreasing trend in the RMSE as the agent progresses, although a sharp rise in RMSE can be attributed to the agent's sudden movement changes from Figure 9. This demonstrates that despite the optimizations methodologies applied for linear KF, the use of non-linear KF such as the unscented Kalman filter can be a better gauge in representing the non-linearity within the process model. This is especially the case with our problem statement, where the measurements contain elements of non-linearity such as calculating the exact distance moved by our agent within a 2D plane: $z^2 = \sqrt{x^2 + y^2}$. In these instances, the use of non-linear Kalman filters may justify a better state estimation than their linear Kalman filter counterparts.

### 3.3 Particle Filter

Our experiment with **particle filter (PF)** likewise uses the same aforementioned parameters. From our configuration, experimental observations show that the particle filter was able to predict the state estimates fairly accurately with **500 particles** in each timestep as shown in Figure 11, where our team observes the close proximity of the predicted measurements against actual measurements. Given the mechanisms of the particle filter (2.4) along with the non-linearities of our state model, this showcases the particle filter's adaptability to changing system dynamics on its propagation and resampling mechanism in environments where system properties can change unpredictably.

Observing the evaluation metrics from Figure 12, our team observes that the **RMSE value for particle filter is at 0.173m**, which is a sudden contrast against our non-linear UKF models which scored predictably better (3.2). Such discrepancies can be attributed to the fact that the variance imposed on the position and velocity for the agent is too high for Monte Carlo sampling and that when iterated over 500 particles, the accuracy of such a model is not exactly concise given the computation of the weighted mean and covariance in order to obtain a state estimate. Furthermore, the optimizations for the linear and non-linear Kalman filters cannot exactly be replicated entirely for particle filter due to the differing computing mechanism between Kalman and particle filters, which contributes to the poor RMSE score.

# 4 Conclusion and Models Comparison

In conclusion, our team has explored the various Kalman filters and their variants, showcasing our model evaluations/analysis and the order of least X/Y position RMSE is as follows: **KF(0.307m) > PF(0.173m) > UKF(0.061m)** with UKF performing the best. Despite this, a notable improvement is to fully optimize the methodologies for unscented Kalman filter and particle filter, given less time constraint. Furthermore, our team would also like to explore the use of conjugate artificial process noise (Wigren et al., 2018) to improve the particle filter in high-dimensional data, which can alleviate scenarios of degeneracy of particle weights within datasets of high dimensionality.

# 5 Figures

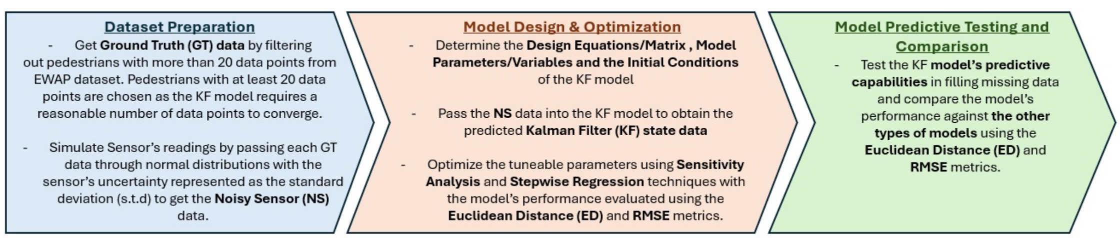
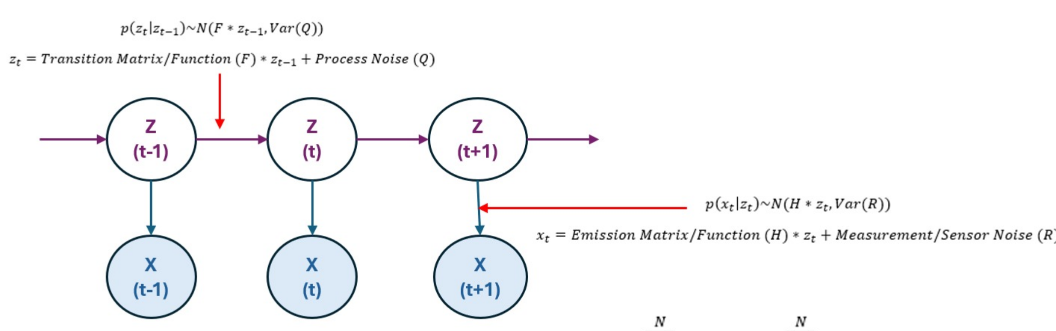



Figure 1: Kalman Filter Design, Implementation, Optimisation & Evaluation methodology




$$Joint\ Distribution = p(x_1,..,x_N,z_1,..,z_N) = p(z_1)(\prod_{t=2}^{N} p(z_t|z_{t-1}))(\prod_{t=1}^{N} p(x_t|z_t))$$

Figure 2: Kalman Filter Markov Diagram

| KF Model Parameters | |
|---|---|
| State Variables (x) | $\begin{bmatrix} \text{x\_position}, & \text{x\_velocity}, & \text{y\_position}, & \text{y\_velocity} \end{bmatrix}$ |
| State Transition Function (F) | $\begin{bmatrix} [1, & \Delta\text{Time}, & 0, & 0 \;] \\ [0, & 1, & 0, & 0 \;] \\ [0, & 0, & 1 & \Delta\text{Time}, ] \\ [0, & 0, & 0, & 1 \;] \end{bmatrix}$ |
| Process Noise Matrix (Q) | $\begin{bmatrix} [\text{X\_position\_noise}, & 0, & 0, & 0 \;] \\ [\;0, & \text{X\_velocity\_noise}, & 0, & 0 \;] \\ [\;0, & 0, & \text{Y\_position\_noise}, & 0 \;] \\ [\;0, & 0, & 0, & \text{Y\_velocity\_noise} ] \end{bmatrix}$ |
| Control Function (B) | 0 (No control input for human crowds) |
| Measurement Emission Function (H) | $\begin{bmatrix} [1, & 0, & 0, & 0] \\ [0, & 1, & 0, & 0] \\ [0, & 0, & 1 & 0] \\ [0, & 0, & 0, & 1] \end{bmatrix}$ |
| Measurement Noise Matrix (R) | $\begin{bmatrix} [\text{position\_sensor\_noise}, & 0, & 0, & 0 \;] \\ [\;0, & \text{velocity\_sensor\_noise}, & 0, & 0 \;] \\ [\;0, & 0, & \text{position\_sensor\_noise}, & 0 \;] \\ [\;0, & 0, & 0, & \text{velocity\_sensor\_noise} ] \end{bmatrix}$ |

Figure 3: Kalman Filter design, implementation, optimisation & evaluation methodology

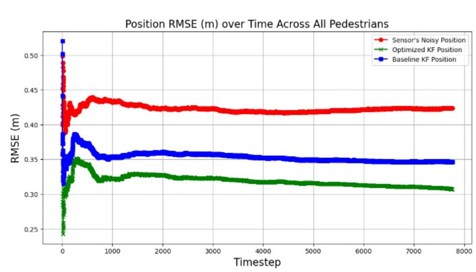



Figure 4: Baseline / Optimized Kalman Filter and Sensor's Noisy Position Stepwise RMSE

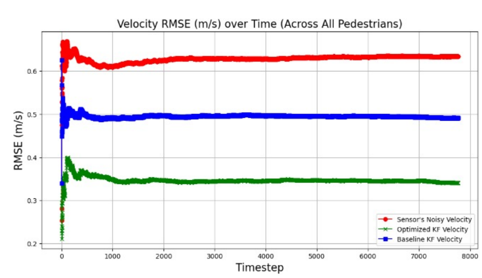



Figure 5: Baseline / Optimized Kalman Filter and Sensor's Noisy Velocity Stepwise RMSE

| Q Uncertainty Tunable Parameters | Change in ED_Sum after (-10%) | Change in ED_Sum after (+10%) |
|---|---|---|
| X Position | -0.00202 | 0.00220 |
| Y Position | -0.00412 | 0.00374 |
| X Velocity | -0.00672 | 0.00601 |
| Y Velocity | -0.00685 | 0.00613 |
| **Conclusion** | | |
| As the goal is to reduce ED_sum, the stepwise regression will follow this order of iteration: | | |

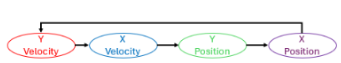



Figure 6: Sensitivity Analysis Results

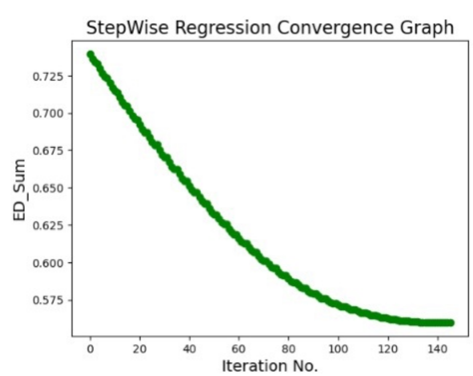



Figure 7: Stepwise Regression Graph

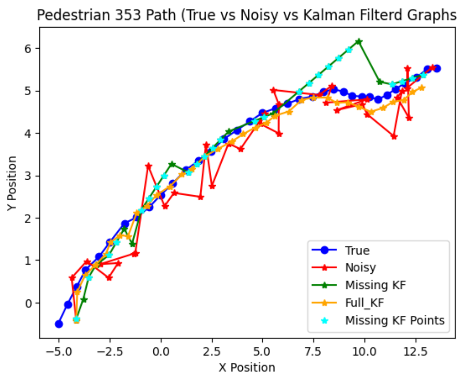



Figure 8: Sample of KF with missing observations. 80% of the 35 observations are missing.

| % of obs missing | % Difference in Position RMSE |
|---|---|
| 20 | 1.49 |
| 30 | 2.52 |
| 50 | 5.94 |
| 75 | 15.26 |
| 90 | 32.22 |

Table 1: Evaluation of full dataset based on the percentage of missing observations. The difference in RMSE is between KF inference on full data vs KF inference on missing data.

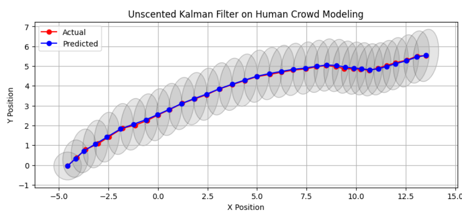



Figure 9: Predicted Path of Agent 353 for Unscented Kalman Filter

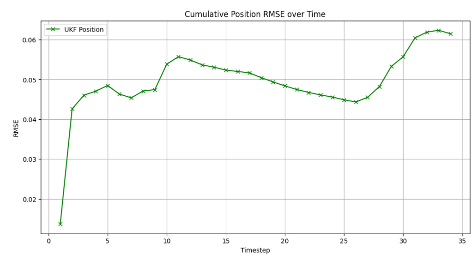



Figure 10: RMSE of Unscented Kalman Filter from Figure 9

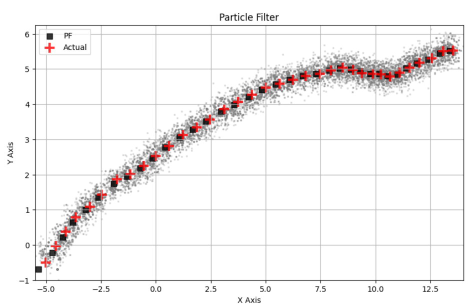



Figure 11: Predicted Path of Agent 353 for Particle Filter

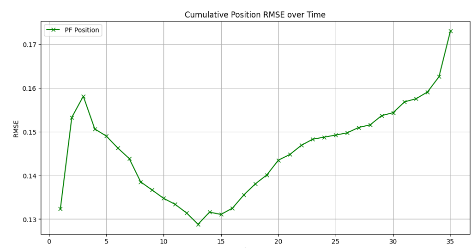



Figure 12: RMSE of Particle Filter from Figure 11

## References

Fritz, M. and Berger, P. D. (2015). Stepwise regression. https://www.sciencedirect.com/topics/computer-science/stepwise-regression. Accessed: 26 April 2024.

Labbe, R. (2014). Kalman and bayesian filters in python. *Chap*, 7(246). 4.

Pellegrini, S., Ess, A., Schindler, K., and Van Gool, L. (2009). You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, pages 261–268. IEEE.

Van Der Merwe, R. (2004). *Sigma-point Kalman Filters for Probabilistic Inference in Dynamic State-space Models*. Phd dissertation, Oregon Health & Science University.

Vipond, T. (2023). What is sensitivity analysis? https://corporatefinanceinstitute.com/resources/financial-modeling/what-is-sensitivity-analysis/. Accessed: 26 April 2024.

Wigren, A., Murray, L., and Lindsten, F. (2018). Improving the particle filter in high dimensions using conjugate artificial process noise. *IFAC-PapersOnLine*, 51(15):670–675.