Pedestrian Motion Estimation Using EKF-, iEKF-, and UKF-based IMM Estimators

Torstein Eliassen
Dept. of Electrical Engineering
Stanford university
torstein.eliassen@stanford.edu

Brian Dobkowski
Dept. of Mechanical Engineering
Stanford University
bdobkows@stanford.edu

Bradley Collicott
Dept. of Aeronautics & Astronautics
Stanford University
collicott@stanford.edu

***Abstract*—In many target tracking applications, it is impractical to predict the movement of an observed target using a single motion model. For the application of pedestrian motion tracking, an Interacting Multiple-Model filter (IMM) is developed for estimating 2D motion from the perspective of a static observer. The filtering system employs the constant velocity, constant turning, and constant acceleration motion models with tunable switching probabilities to capture the multimodal nature of human motion. UKF-IMM, EKF-IMM, and iEKF-IMM filters are implemented on both synthetic and real pedestrian data, and measures are taken to increase numerical stability of the algorithms.**[1]

I. Introduction

Many dynamical systems are multimodal, meaning that their behavior depends on what mode the system is currently in. In particular, even though systems have piecewise continuous behavior, changing of mode poses challenges for filtering. The typical Kalman filter uses only a single dynamics model and thus, together with many of its extensions, it is not suited for filtering the mentioned hybrid systems. The goal of this project is to show how a multi-modal filtering approach can be more more suited for this purpose. Through a pedestrian dataset, an inherently unpredictable multimodal system and limited measurements, the IMM will be implemented, modified, tuned and tested.

II. Literature review

A. Dynamical and Multi-Modal System Estimation

The Interacting Multiple Model (IMM) was first introduced in [1] as a filtering method for multimodal systems. Similar to the Generalized Pseudo Bayesian 2 (GBP2) [2], the IMM filtering method assumes M different states/models the system can be in. However to reduce the number of filters to maintain, the IMM reduces the mixture before prediction and is therefore more computationally efficient than its counterpart, being ”nearly linear” vs. ”quadratic” in the number of models [3]. The authors of [3] provide a survey of the most important IMM methods, explain the hybrid state estimation problem, and compare the IMM to other multi-modal (MM) models, showing that despite the computational savings, the algorithm can obtain high performance. Because of this, they claim it may be well suited for airborne systems, for example, where computation is limited (as the IMM still requires more computation than using a single track filter).

The IMM has been integrated in several other filters to allow for multimodal distributions without compounding computation. As an example, [4] presents a new method for multiple model particle filtering using the IMM. The paper claims to handle nonlinearities and non-gaussian noise. In addition, their method uses a fixed number of particles for each mode, something they claim mitigates certain drawbacks with other MM particle filters.

The authors in [5] survey additional approaches in which different filter models, including UKF, Cubature Kalman Filters, and Particle Filters, are fused using an IMM algorithm. They also discuss algorithmic improvements, including ways to optimize the mode transition matrix [6] and ways to improve computational efficiency. Neural networks have also made their way into multimodal filtering. In [7], for example, RNNs were used to classify modes and augment the maneuver detection response in the IMM algorithm.

B. Applications

The IMM may be applied to any system in which (1) the state evolves according to a discrete difference or continuous differential model and (2) has a finite number of possible models that the system switches between according to a set of transition probabilities. Several applications from literature are discussed here.

Jo et al. [8] use an EKF-based IMM to develop a positioning model that uses low-grade GPS fused with in-vehicle sensors. Both kinematic and dynamic motion models are included in the navigation filter to estimate the position and heading angle of the vehicle. The transition between modes is modeled through a stationary probability distribution – the static parameters of this distribution contrast with IMM filter architectures which estimate the Markov switching parameters. The authors demonstrate that the IMM architecture outperforms single-model EKF while maintaining robustness to GPS faults via sensor fusion.

Schneider and Gavrila [9] apply the EKF-based IMM to pedestrian path prediction, using the standard Constant Velocity (CV), Constant Acceleration (CA), and Constant Turn (CT) motion models. Stereo camera images are used to provide measurements 2D pixel coordinate measurements

[1] Code is available at https://github.com/gravlaks/IMM_project

of pedestrians in the image frame and a scalar measurement of stereo disparity. They compare performance on several common pedestrian actions with each action having a unique *sojourn time* representing the characteristic time scale of the action. The authors conclude that the IMM filter improves performance, but the advantage may be moot in this particular situation due to the high measurement frequency and accuracy. The approach taken in this paper grew primarily out of Schneider and Gavrila's study.

Jiang et al. [10] also use the CV and CA motion models applied to moving obstacle motion prediction, but also introduce the Current Statistical (CS) model to the EKF IMM filter. The CS motion model is introduced to capture nonlinearities over short time periods by predicting the future acceleration via a probability distribution centered at the current acceleration estimate. The scheme is implemented in physical robots with monocular camera and laser scanning sensors to estimate 2D position, velocity, and acceleration.

The IMM filter may also be extended to cooperative multi-robot and learning-based applications. Pierpaoli et al. [11] use an IMM architecture to estimate the trajectory of an "expert system" for use in imitation learning. The use of multiple model hypotheses allows for unlabelled observations to be used in training the neural network policy for controlling the team of robots. This method yields policies that enable similar results to expert systems on a pursuit task.

This project builds upon the pedestrian motion estimation work in [9] by improving upon existing EKF-based techniques and evaluating UKF and iEKF filter backbones in hopes to alleviate losses due to linearization error that have been present in prior work. The constant velocity (CV), constant acceleration (CA), and constant turn (CT) models are used with range and bearing-vector measurements to estimate 2D pedestrian trajectories.

III. Problem Formulation

Assuming that the trajectory of the pedestrian is governed by a continuous-time differential equation.

$$\dot{x}(t) = f(x, \nu, s_k) \tag{1}$$

where f is the dynamics model, $x(t) \in \mathbb{R}^7$ consists of 2D position, velocity, acceleration and turn rate. s_k is the current mode of the pedestrian, ν is white noise fed to the system.

We choose to model pedestrian dynamics through three different modes: Constant Velocity, Constant Acceleration and Constant Turn rate Let $p(t), v(t), a(t)$ be respectively the position, velocity and acceleration of the pedestrian. We give the continuous-time motion models below, for which we derive discretized models.

A. Motion models

1) Constant velocity: The CV model assumes only white-noise acceleration.

$$\begin{aligned} \dot{p}(t) &= v(t) \\ \dot{v}(t) &= \nu \end{aligned} \tag{2}$$

, $\nu \sim \mathcal{N}(\mathbf{0}, \sigma_q \boldsymbol{I},)$

2) Constant Acceleration: The CA model assumes only white-noise jerk

$$\begin{aligned} \dot{p}(t) &= v(t) \\ \dot{v}(t) &= a(t) \\ \dot{a}(t) &= \nu \end{aligned} \tag{3}$$

, $\nu \sim \mathcal{N}(\mathbf{0}, \sigma_a \boldsymbol{I},)$

3) Constant Turning Rate: The CT model assumes both white-noise turn-rate acceleration and white-noise acceleration.

$$\begin{aligned} \dot{p}(t) &= v(t) \\ \dot{v_x}(t) &= -\omega v_y(t) + \nu_{a,x} \\ \dot{v_y}(t) &= \omega v_x(t) + \nu_{a,y} \\ \dot{\omega}(t) &= \nu_\omega \end{aligned} \tag{4}$$

, $\nu_\omega \sim \mathcal{N}(0, \sigma_\omega,)$, $\nu_{a,x} \sim \mathcal{N}(0, \sigma_a,)$, $\nu_{a,y} \sim \mathcal{N}(0, \sigma_a,)$.

If ω is known, the constant turn model would be linear; however, knowing the turning rate is unrealistic. To appropriately capture the nonlinear dynamics of the CT model, the angular rate is included in the state and estimated along with the linear velocity and acceleration.

B. Derivation of the discrete dynamics models

Deriving the discrete dynamics models is done through exact linearization of stochastic LTIs. Our motion models are linear (CV, CA) or will be approximated as linear (CT). They can thus all be expressed in the following form:

$$\dot{x} = Ax + B\nu \tag{5}$$

where $\nu \sim \mathcal{N}(\mu_v, \tilde{Q}_k)$.

We want a discrete system in the form:

$$x_k = Fx_{k-1} + Gw_k \tag{6}$$

Let T be the timestep. From linear system theory, we have that:

$$F = e^{AT},\ w_k = \int_{t_{k-1}}^{t_k} e^{AT} B \nu_k \tag{7}$$

[12] shows:

$$Q_k = E[w_k w_k^T] = \int_{t_{k-1}}^{t_k} e^{(t_k - \tau)A} B \tilde{Q}_k B^T e^{(t_k - \tau)A^T} d\tau \tag{8}$$

, $\omega_k \sim \mathcal{N}(\mu_w, Q_k)$ In situations where the integral is not trivial, it can be evaluated by solving the following equation [13]:

$$exp(\begin{bmatrix} -A & B\tilde{Q}_k B^T \\ 0 & A^T \end{bmatrix}) = \begin{bmatrix} x & V_2 \\ 0 & V_1 \end{bmatrix},\ Q = V_1^T V_2 \tag{9}$$

The derivations of discretization of each motion model are given in the appendices.

C. Measurement Models

Range ρ and bearing angle θ measurements will be provided to the filter.

$$\rho = \|p\|_2 + \nu_\rho \qquad \theta = \frac{p}{\|p\|_2} + \nu_\theta \tag{10}$$

The measurement noise is modeled as zero-mean Gaussian white noise with known covariance $\nu \sim \mathcal{N}(0, R)$. Unless otherwise specified, the variances of the measurements are $0.1m^2$ and $0.01rad^2$ for range and bearing, respectively.

D. Mode Transition

We assume that the target modes can be modeled using a Hidden Markov Model, as in [14]. Thus, the mode evolves according to some transition probabilities π^{ij}. Let p denote the mode probability vector, then the system's mode evolution can be modeled by

$$p_t = \pi^T p_{t-1} \tag{11}$$

E. Interacting Multiple Models

The IMM algorithm models each of these modes individually and blends their output to provide the most realistic state estimate based on the true operational mode of the pedestrian. The algorithm provides an automated way to adaptively estimate which dynamics fit the pedestrian the best at any given moment.

In an IMM, there are $s = 1, ..., M$ system modes (also called sojourns), and these modes change according to a Markov process with transition probabilities π^{ij}. Each mode's dynamics are modeled separately and used in distinct filter models. The algorithm uses 11 to evolve the modes.

The IMM has 4 steps, explained and derived in [14]. Step one calculates the mixing probabilities, which are the probabilities of the system's mode at time $t-1$, conditioned on the current mode. In step two, these mixing probabilities, and the M-Gaussian mixture posterior from time $t{-}1$, are used to approximate single Gaussian priors for each filter model. In step three, each filter model is run independently using its mode-matched prior, and as a byproduct a mode-conditional likelihood is calculated. In step four, the mode probabilities p_t are updated using the mode-conditional likelihood and the predicted probability from Equation 11.

The result is a Gaussian mixture model posterior - by using M filter models in parallel, we can receive a multimodal distribution by fusing each model's Gaussian posterior according to its weighted probability given the measurement. Also, we can approximate this Gaussian mixture with a single Gaussian through moment matching, which likely has higher accuracy than any one model, if we desire a unimodal posterior distribution. A key feature of the IMM is that it reduces the Gaussian mixture *before* the filtering step, meaning it must only maintain one filter per mode.

IV. Experimentation and Metrics

The primary experiment will be building a suite of filter models and fusing their estimates with the IMM algorithm, and comparing these results to the single filter output. The expectation is that the IMM will provide better state estimates, but experiments will be done to quantify the improvement in performance and examine the effect of using different filter algorithms and dynamics models. The computation time of the IMM will also be examined and compared to that of unimodal filters to quantify the cost of using the algorithm.

Our main success criteria will be mean square error (MSE) of the ground truth and estimated trajectories

$$\mathcal{L}_{MSE} = \frac{1}{N}\sum_{k=0}^{N}\|p_k - \hat{p}_k\|_2^2 \tag{12}$$

where $\hat{(\cdot)}$ represents an estimated quantity. The filter is also examined in terms of uncertainty quantification by comparing the filter covariance to the true trajectory estimation error.

A. Data and Ground Truth

The IMM will be evaluated using simulated data generated using the CV, CA, and CT motion models to validate its ability to effectively estimate the target mode and state. The performance will be further evaluated on a 2D pedestrian tracking dataset from [15] which provides ground truth position and velocity for individual pedestrians in a crowd. These trajectories adequately capture the motion of pedestrians interacting with one another as well as with vehicles.

V. Contributions and Extensions

A. State Expansion

The IMM is developed for systems of the same state. In order to use the CV, CA and CT model together in the IMM, we use the same state vector in all: $x = [x, y, \dot{x}, \dot{y}, \ddot{x}, \ddot{y}, \omega]$.

The CV model is simple as it estimates acceleration and turn rate to be zero. The CT model is traditionally given without acceleration in the state, and the CA model is given without the turn rate in the state as it would be redundant information. However, when reducing the Gaussian mixture and mixing the models, we have to find a formula for finding the turn rate in the CA model and vice versa, the acceleration in the CT model.

We have for the CA model that

$$\begin{aligned}
\theta &= arctan(\dot{y}/\dot{x}) \\
\omega &= \begin{bmatrix} \frac{d\theta}{dx} & \frac{d\theta}{dy} \end{bmatrix} \begin{bmatrix} \frac{d\dot{x}}{dt} \\ \frac{d\dot{y}}{dt} \end{bmatrix} \\
&= \begin{bmatrix} \frac{\dot{-y}}{\dot{x}^2+\dot{y}^2} & \frac{\dot{x}}{\dot{x}^2+\dot{y}^2} \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} \\
&= \frac{-\dot{y}\ddot{x} + \dot{x}\ddot{y}}{\dot{x}^2 + \dot{y}^2}
\end{aligned} \tag{13}$$

and for the CT model that:

$$\begin{aligned}
\ddot{x} &= -\omega\dot{y} \\
\ddot{y} &= \omega\dot{x}
\end{aligned} \tag{14}$$

In order to achieve the correct covariance estimations, we also have to include these results when calculating the Jacobians for the respective models (when using an EKF).

CA:

$$\begin{aligned} \frac{\partial \omega}{\partial \ddot{x}} &= \frac{-\dot{y}}{\dot{x}^2+\dot{y}^2}, \frac{\partial \omega}{\partial \ddot{y}} = \frac{\dot{x}}{\dot{x}^2+\dot{y}^2} \\ \frac{\partial \omega}{\partial \dot{x}} &= \frac{\ddot{y}}{\dot{x}^2+\dot{y}^2} - 2\dot{x}\frac{\ddot{y}\dot{x}-\ddot{x}\dot{y}}{(\dot{x}^2+\dot{y}^2)^2} \\ \frac{\partial \omega}{\partial \dot{y}} &= \frac{-\dot{x}}{\dot{x}^2+\dot{y}^2} + 2\dot{y}\frac{\ddot{x}\dot{y}-\ddot{y}\dot{x}}{(\dot{x}^2+\dot{y}^2)^2} \end{aligned} \tag{15}$$

CT:

$$\begin{aligned} \frac{\partial \ddot{x}}{\partial \omega} = -\dot{y}, \frac{\partial \ddot{x}}{\partial \dot{y}} &= -\omega \\ \frac{\partial \ddot{y}}{\partial \omega} = \dot{x}, \frac{\partial \ddot{y}}{\partial \dot{x}} &= \omega \end{aligned} \tag{16}$$

B. Numerical instability measures

The IMM may be prone to numerical instability. In the original Kalman update, there is a subtraction that may (due to round off errors) lead to a non positive semidefinite matrix). The Joseph form is therefore instead used for the covariance update:

$$\Sigma_{t|t} = (I - KC)\Sigma_{t|t-1}(I - KC) + KRK^T \tag{17}$$

where $\Sigma_{t|t}$ is the posterior covariance, $\Sigma_{t|t-1}$ is the prediction covariance, I is the identity matrix, K is the Kalman gain, C is the measurement jacobian and R is the measurement noise covariance. This update is computationally more expensive, but is less sensitive to round off errors [16]. Given $\Sigma_{t|t-1}$ positive definite, it preserves positive definitess by performing the sum of a positive definite matrix and a semipositive definite matrix. The Joseph form is also implemented in the UKF following the derivations from [17].

Additionally, in a multimodal filter, the likelihoods of a given model can become close to zero. This is a common occurence when using artificial data. When renormalizing weights before the mixing steps, a situation may occur where the weights are so small that they sum to zero numerically. This situation is prevented by simply having a threshold on the mode log likelihood.

C. Merging of states heuristics

In the standard IMM algorithm, the M posteriors, which are all gaussian mixtures, are reduced to Gaussians through moment matching. This has the unfortunate effect that, when mixing the CV turn rate or acceleration with the CA/CT turn rate / acceleration, the states will be drawn towards 0 for any nonzero CV likelihood as the CV model by design predicts 0 turn rate and acceleration.

In order to avoid this bias towards 0 for the turn rate and accelerations, we employ the heuristic of only mixing CT and CA accelerations/turn rates. No longer will the mixed gaussian and the reduced gaussian have the same moments, but this will give a more realistic turn rate and acceleration for the CT/CA models.

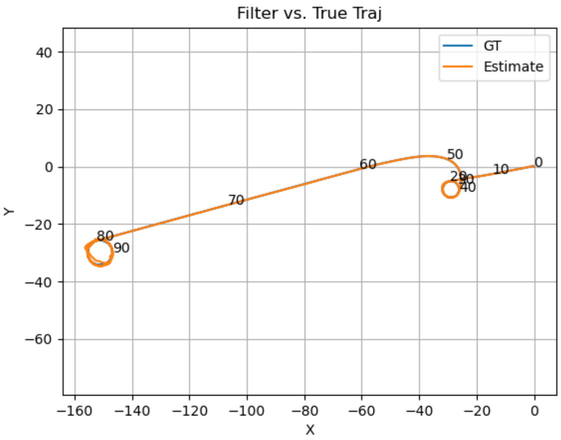


Fig. 1: Simulated CV, CT, CA Trajectory

D. Multiple filter models

In addition to having separate motion models, we implement the IMM with multiple filters as well, namely the EKF, iEKF and the UKF. The iEKF, by taking a MAP approach to the update step, should improve errors due to linearization at a small computational expense. The UKF, by linearizing via the Unscented Transform [18], may handle nonlinearities even better. We expect these filters to provide some performance improvement due to the nonlinear CT and measurement models.

VI. Results and Discussion

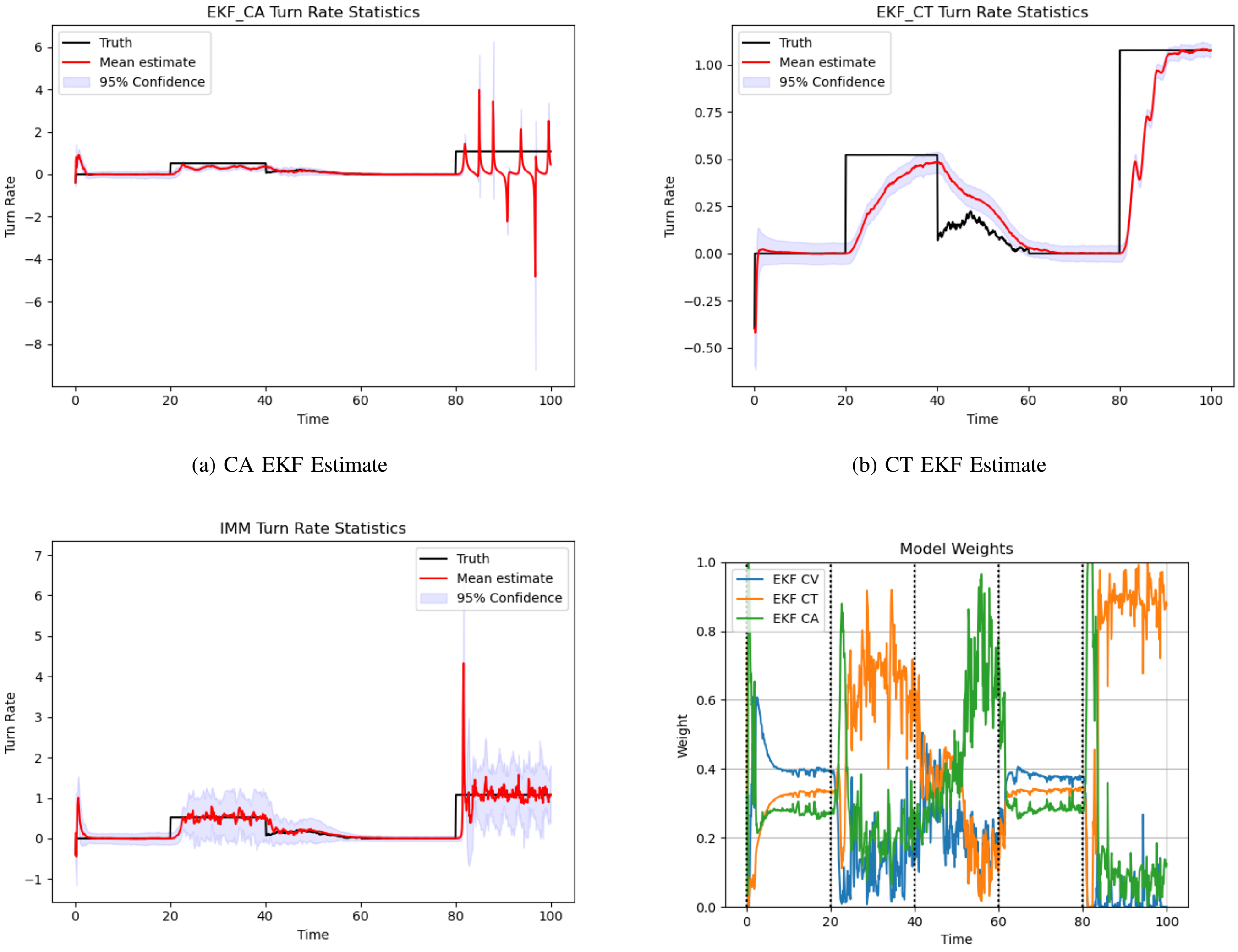
A. Simulated Data

To validate the filters and the IMM algorithm, trajectories (similar to that in Figure 1) were simulated using the same dynamics models used for propagation within the filter models, but with different noise. By doing so, it was proven that the IMM algorithm correctly identified the target's mode and switched its model probabilities to suit that mode.

Performance in the simulation phase can be seen in Figure 2. This trajectory cycles from CV-CT-CA-CV-CT, as shown by the dashed black lines on the model weights plot. It can be seen that the IMM favors the correct model during its sojourn time. The added bonus of including a CA model is that it can pick up quick changes between modes, which would be useful when a pedestrian starts or stops motion, or begins to turn. It makes sense that during constant velocity sojourns, the probabilities do not vastly favor the CV model, since the filters can estimate zero acceleration with all models.

B. Pedestrian Dataset

A 2D pedestrian tracking dataset was used to validate the performance of the IMM. The 318 top-view pedestrian tracks from a crowd-vehicle interaction dataset [15] were used as the ground truth trajectories. All trajectories are provides in units of meters at a rate of $30hz$ – times are presented in seconds unless otherwise specified.



(a) CA EKF Estimate

(b) CT EKF Estimate

(c) EKF IMM Estimate

(d) Model Probabilities

Fig. 2: Turn Rate Estimation of CA, CT, and IMM Filters on Simulated Trajectory.

The trajectories were estimated with simulated range and bearing measurements using EKF, iEKF, and UKF filters with varying combinations of motion models. The aggregate statistics for MSE position tracking and computation time are shown in Table I.

Trajectories for a representative trajectory from the pedestrian dataset are shown for each filter in Fig. 4. The IMM mode weights for the CV and CA models are shown in Fig. 5.

From the tabulated results, we see that the best performer in each base filter category is an IMM. For the EKF and iEKF filters, the CV/CA model outperforms all others in terms position error and standard deviation, but suffers a significant penalty in computation time. We observe a small performance increase with the iEKF compared to the EKF on the unimodal filters CV and CA, possibly due to the nonlinearity in the measurement model. This comes with a small runtime cost. In the UKF category, we see that the largest benefit is gained on the IMM with the non-linear CT motion model, again with a 4-5x penalty in runtime. A qualitative analysis of the representative trajectories in Fig. 4 shows that the IMMs outperform the best single filter model for the EKF base filter. The CV/CA models together are able to track the trajectory very closely, while the CV/CT model is slightly less effective, but shows better performance than the CT EKF alone. The Model weights shown in Fig. 5 show that the IMM makes use of the CA model only during the portions of the trajectory where the pedestrian is ”maneuvering”. This behavior demonstrates the ability of the IMM to capitalize on using multiple motion models in its filtering scheme by using the simple CV model most of the time and increasing weight for the CA model during more complex motion patterns.

VII. Conclusion

An IMM filtering architecture was implemented with EKF, iEKF, and UKF filters for the purpose of pedestrian tracking.

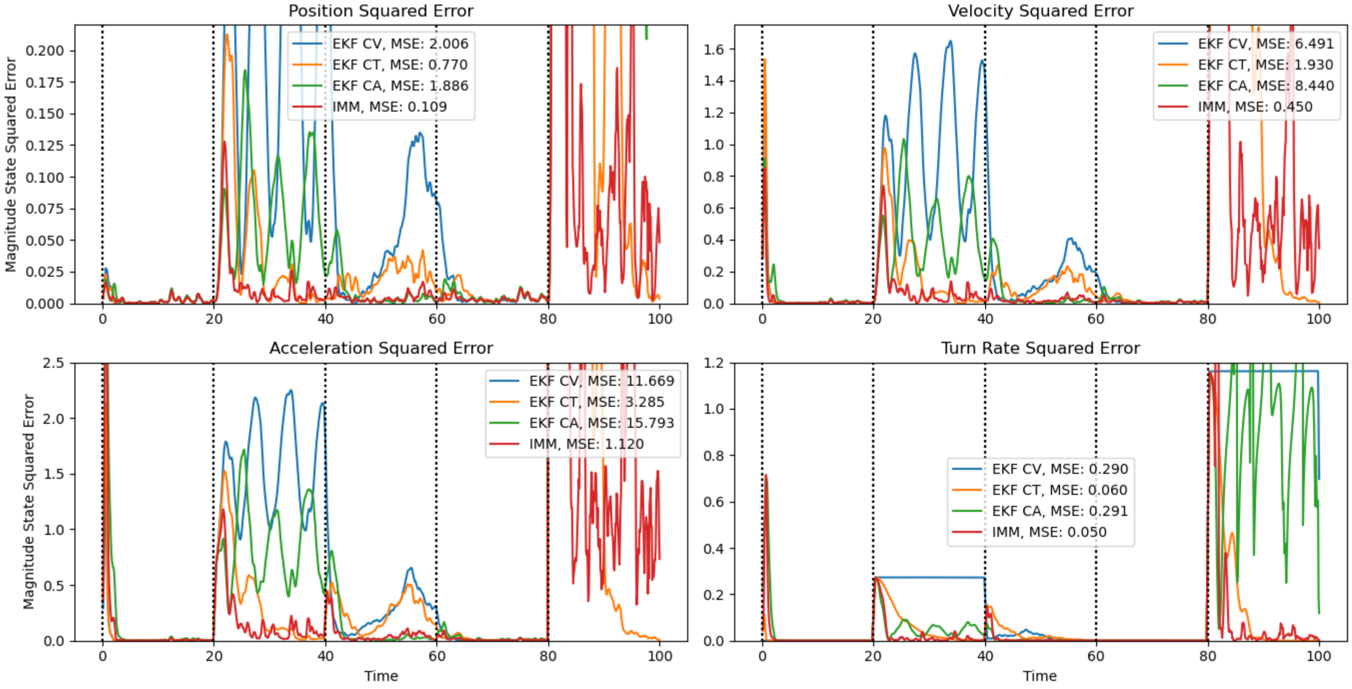


Fig. 3: IMM and Individual Filter Errors for Simulated Trajectory

TABLE I: Position Error and Timing for Pedestrian Dataset

Filter	Mean MSE	1-σ MSE	Mean Iter.	1-σ Iter.
CV/CT EKF IMM	3.199 m	2.521 m	1.467 ms	0.044 ms
CV/CA EKF IMM	**2.333 m**	**0.930 m**	2.270 ms	0.205 ms
CV EKF	3.409 m	2.459 m	0.285 ms	0.034 ms
CA EKF	3.443 m	2.251 m	0.326 ms	0.079 ms
CT EKF	3.310 m	2.569 m	**0.273 ms**	**0.025 ms**
CV/CT iEKF IMM	3.199 m	2.521 m	1.734 ms	0.049 ms
CV/CA iEKF IMM	**2.333 m**	**0.930 m**	2.203 ms	0.083 ms
CV iEKF	3.297 m	2.539 m	**0.300 ms**	**0.017 ms**
CA iEKF	3.436 m	2.186 m	0.412 ms	0.033 m
CT iEKF	3.310 m	2.569 m	0.443 ms	0.025 ms
CV/CT UKF IMM	**2.535 m**	**0.955 m**	4.134 ms	0.317 ms
CV/CA UKF IMM	2.738 m	0.932 m	4.458 ms	0.221 ms
CV UKF	3.213 m	2.521 m	1.333 ms	0.499 ms
CA UKF	3.444 m	2.251 m	1.485 ms	0.068 m
CT UKF	3.295 m	2.543 m	**1.332 ms**	**0.059 ms**

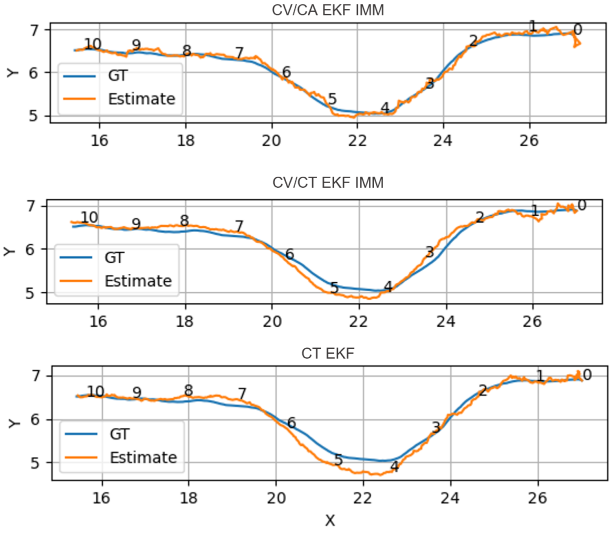


Fig. 4: Pedestrian Tracking Trajectories

The IMM was improved via extensions for added numerical stability and a model mixing heuristic to appropriately account for turning rate and acceleration estimates across models. The IMM model weighting was validated on simulated data generated using CV, CA, and CT motion models.

Further, the IMM was shown to outperform single-model estimators on a real-world pedestrian trajectory dataset. Notably, the EKF IMM was shown to outperform all single-model variants of the iEKF and UKF filters. The EKF IMM, however, suffers a computation time penalty on par with the single-model UKF. Substituting the UKF as the base filter in the IMM garnered a significant improvement in CT motion model, demonstrating the gains of using a derivative-free approach with a non-linear motion model.

Overall, the IMM provides up to 33% performance improvement over single-model filters in terms of mean squared position error on a 2D pedestrian dataset at the expense of 5x

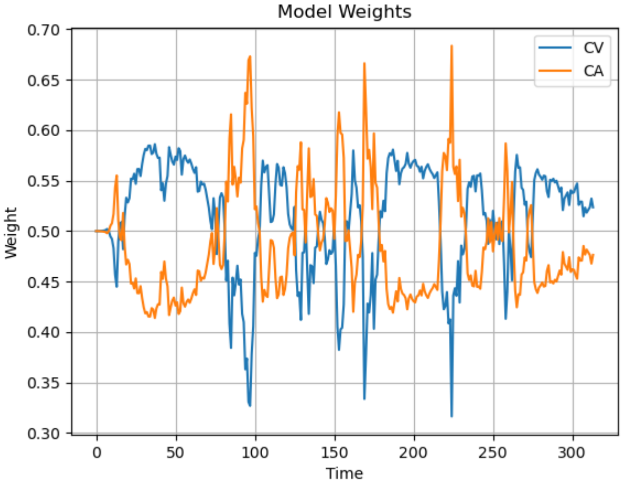


Fig. 5: IMM CV and CA Probabilities for Ped. Traj.

computation time compared to the standard EKF. Future effort could be directed at parallelizing the internal operations of the IMM such as the M filter updates. Also, one could introduce more complex motion models, and evaluate the IMM on 3D trajectories for both vehicles and pedestrians.

References

[1] H. Blom, "An efficient filter for abruptly changing systems," pp. 656 – 658, 12 1984.

[2] S. Challa, M. R. Morelande, D. Mušicki, and R. J. Evans, *Fundamentals of object tracking*. Cambridge University Press, 2011.

[3] E. Mazor, A. Averbuch, Y. bar shalom, and J. Dayan, "Interacting multiple model methods in target tracking: A survey," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 34, pp. 103 – 123, 02 1998.

[4] Y. Boers, "Interacting multiple model particle filter," *IEE Proceedings - Radar, Sonar and Navigation*, vol. 150, pp. 344–349(5), October 2003.

[5] A. Akca and M. Önder Efe, "Multiple model kalman and particle filters and applications: A survey," *IFAC-PapersOnLine*, vol. 52, no. 3, pp. 73–78, 2019. 15th IFAC Symposium on Large Scale Complex Systems LSS 2019.

[6] L. Sun, C. Shen, *et al.*, "An improved interacting multiple model algorithm used in aircraft tracking," *Mathematical Problems in Engineering*, vol. 2014, 2014.

[7] L. Deng, D. Li, and R. Li, "Improved imm algorithm based on rnns," in *Journal of Physics: Conference Series*, vol. 1518, p. 012055, IOP Publishing, 2020.

[8] K. Jo, K. Chu, and M. Sunwoo, "Interacting multiple model filter-based sensor fusion of gps with in-vehicle sensors for real-time vehicle positioning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 329–343, 2012.

[9] N. Schneider and D. M. Gavrila, "Pedestrian path prediction with recursive bayesian filters: A comparative study," in *Pattern Recognition* (J. Weickert, M. Hein, and B. Schiele, eds.), (Berlin, Heidelberg), pp. 174–183, Springer Berlin Heidelberg, 2013.

[10] R. Jiang, X. Tian, L. Xie, and Y. Chen, "A robot collision avoidance scheme based on the moving obstacle motion prediction," in *2008 ISECS International Colloquium on Computing, Communication, Control, and Management*, vol. 2, pp. 341–345, 2008.

[11] P. Pierpaoli, H. Ravichandar, N. Waytowich, A. Li, D. Asher, and M. Egerstedt, "Inferring and learning multi-robot policies by observing an expert," 2019.

[12] E. Brekke, *Fundamentals of Sensor Fusion: Target tracking, navigation and SLAM*. 2020.

[13] C. Van Loan, "Computing integrals involving the matrix exponential," *IEEE Transactions on Automatic Control*, vol. 23, no. 3, pp. 395–404, 1978.

[14] Y. Bar-Shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*. Wiley, 2004.

[15] D. Yang, L. Li, K. Redmill, and U. Ozguner, "Top-view trajectories: A pedestrian dataset of vehicle-crowd interaction from controlled experiments and crowded campus," pp. 899–904, 06 2019.

[16] *Computational Aspects of Estimation*, ch. 7, pp. 301–318. John Wiley & Sons, Ltd.

[17] F. De Vivo, A. Brandl, M. Battipede, and P. Gili, "Joseph covariance formula adaptation to square-root sigma-point kalman filters," *Nonlinear dynamics*, vol. 88, no. 3, pp. 1969–1986, 2017.

[18] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *Signal processing, sensor fusion, and target recognition VI*, vol. 3068, pp. 182–193, International Society for Optics and Photonics, 1997.

Appendix A
Derivation of CV Discrete Time Process Noise Matrix

The continuous time model for constant velocity is

$$\dot{p}_k = v_k,\ \ \dot{v}_k = \tilde{\nu}_k,\ \ \dot{a}_k = 0$$

Where $\tilde{\nu}_k \sim \mathcal{N}(0, \tilde{Q}_k)$. The discrete system and process noise mapping matrices can be written assuming the process noise enters via the velocity term:

$$F = \begin{bmatrix} \boldsymbol{I}_{2\times2} & T\boldsymbol{I}_{2\times2} & \mathbf{0}_{2\times2} \\ \mathbf{0}_{2\times2} & \boldsymbol{I}_{2\times2} & \mathbf{0}_{2\times2} \\ \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} \end{bmatrix} \qquad B = \begin{bmatrix} \mathbf{0}_{2\times2} \\ \boldsymbol{I}_{2\times2} \\ \mathbf{0}_{2\times2} \end{bmatrix}$$

Where the discrete time process noise is $w_k \sim \mathcal{N}(0, Q_k)$. To map the continuous time process noise ν to the discrete time process noise w, we use the following equation:

$$Q = \int_{t_{k-1}}^{t_k} F_k B_k \tilde{Q} B_k^{\intercal} F_k^{\intercal} d\tau$$

$$F_k B_k = \begin{bmatrix} \boldsymbol{I}_{2\times2} & T\boldsymbol{I}_{2\times2} & \mathbf{0}_{2\times2} \\ \mathbf{0}_{2\times2} & \boldsymbol{I}_{2\times2} & \mathbf{0}_{2\times2} \\ \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} \end{bmatrix} \begin{bmatrix} \mathbf{0}_{2\times2} \\ \boldsymbol{I}_{2\times2} \\ \mathbf{0}_{2\times2} \end{bmatrix} = \begin{bmatrix} T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \\ \mathbf{0}_{2\times2} \end{bmatrix}$$

The continuous process noise covariance is given by $\tilde{Q} = diag([\sigma_v^2 \ \ \sigma_v^2])$

$$F_k B_k \tilde{Q} = \begin{bmatrix} T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \\ \mathbf{0}_{2\times2} \end{bmatrix} \sigma_\nu^2 \boldsymbol{I}_{2\times2} = \sigma_\nu^2 \begin{bmatrix} T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \\ \mathbf{0}_{2\times2} \end{bmatrix}$$

$$F_k B_k \tilde{Q} B_k^{\intercal} F_k^{\intercal} = \sigma_v^2 \begin{bmatrix} T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \\ \mathbf{0}_{2\times2} \end{bmatrix} \begin{bmatrix} T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \\ \mathbf{0}_{2\times2} \end{bmatrix}^T$$

$$= \sigma_v^2 \begin{bmatrix} T^2\boldsymbol{I}_{2\times2} & T\boldsymbol{I}_{2\times2} & \mathbf{0}_{2\times2} \\ T\boldsymbol{I}_{2\times2} & \boldsymbol{I}_{2\times2} & \mathbf{0}_{2\times2} \\ \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} \end{bmatrix}$$

Thus we can solve for the process noise matrix by evaluating the integral from above.

$$Q = \sigma_v^2 \begin{bmatrix} T^3\boldsymbol{I}_{2\times2}/3 & T^2\boldsymbol{I}_{2\times2}/2 & \mathbf{0}_{2\times2} \\ T^2\boldsymbol{I}_{2\times2}/2 & T\boldsymbol{I}_{2\times2} & \mathbf{0}_{2\times2} \\ \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} \end{bmatrix}$$

This result is consistent with Equation (4.64) in [12].

APPENDIX B
CONSTANT ACCELERATION

The CA continuous time dynamics are:

$$\dot{p}_k = v_k,\ \dot{v}_k = a_k,\ \dot{a}_k = 0$$

Where $\tilde{\nu}_k \sim \mathcal{N}(0, \tilde{Q}_k)$. The discrete system and process noise mapping matrices can be written assuming the process noise enters via the acceleration term:

$$F = \begin{bmatrix} \boldsymbol{I}_{2\times2} & T\boldsymbol{I}_{2\times2} & T^2\boldsymbol{I}_{2\times2}/2 \\ \boldsymbol{0}_{2\times2} & \boldsymbol{I}_{2\times2} & T\boldsymbol{I}_{2\times2} \\ \boldsymbol{0}_{2\times2} & \boldsymbol{0}_{2\times2} & \boldsymbol{I}_{2\times2} \end{bmatrix} \qquad B = \begin{bmatrix} \boldsymbol{0}_{2\times2} \\ \boldsymbol{0}_{2\times2} \\ \boldsymbol{I}_{2\times2} \end{bmatrix}$$

Where the discrete time process noise is $w_k \sim \mathcal{N}(0, Q_k)$. To map the continuous time process noise ν to the discrete time process noise w, we use the following equation:

$$Q = \int_{t_{k-1}}^{t_k} F_k B_k \tilde{Q} B_k^{\intercal} F_k^{\intercal} d\tau$$

$$F_k B_k = \begin{bmatrix} \boldsymbol{I}_{2\times2} & T\boldsymbol{I}_{2\times2} & T^2\boldsymbol{I}_{2\times2}/2 \\ \boldsymbol{0}_{2\times2} & \boldsymbol{I}_{2\times2} & T\boldsymbol{I}_{2\times2} \\ \boldsymbol{0}_{2\times2} & \boldsymbol{0}_{2\times2} & \boldsymbol{I}_{2\times2} \end{bmatrix} \begin{bmatrix} \boldsymbol{0}_{2\times2} \\ \boldsymbol{0}_{2\times2} \\ \boldsymbol{I}_{2\times2} \end{bmatrix} = \begin{bmatrix} T^2\boldsymbol{I}_{2\times2}/2 \\ T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \end{bmatrix}$$

The continuous process noise covariance is given by $\tilde{Q} = diag([\sigma_a^2 \;\; \sigma_a^2])$

$$F_k B_k \tilde{Q} = \begin{bmatrix} T^2\boldsymbol{I}_{2\times2}/2 \\ T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \end{bmatrix} \sigma_a^2 \boldsymbol{I}_{2\times2} = \sigma_a^2 \begin{bmatrix} T^2\boldsymbol{I}_{2\times2}/2 \\ T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \end{bmatrix}$$

$$F_k B_k \tilde{Q} B_k^{\intercal} F_k^{\intercal} = \sigma_a^2 \begin{bmatrix} T^2\boldsymbol{I}_{2\times2}/2 \\ T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \end{bmatrix} \begin{bmatrix} T^2\boldsymbol{I}_{2\times2}/2 \\ T\boldsymbol{I}_{2\times2} \\ \boldsymbol{I}_{2\times2} \end{bmatrix}^T$$

$$= \sigma_a^2 \begin{bmatrix} T^4\boldsymbol{I}_{2\times2}/4 & T^3\boldsymbol{I}_{2\times2}/2 & T^2\boldsymbol{I}_{2\times2}/2 \\ T^3\boldsymbol{I}_{2\times2}/2 & T^2\boldsymbol{I}_{2\times2} & T\boldsymbol{I}_{2\times2} \\ T^2\boldsymbol{I}_{2\times2}/2 & T\boldsymbol{I}_{2\times2} & \boldsymbol{I}_{2\times2} \end{bmatrix}$$

Thus we can solve for the process noise matrix by evaluating the integral from above.

$$Q = \sigma_a^2 \begin{bmatrix} T^5\boldsymbol{I}_{2\times2}/20 & T^4\boldsymbol{I}_{2\times2}/8 & T^3\boldsymbol{I}_{2\times2}/6 \\ T^4\boldsymbol{I}_{2\times2}/8 & T^3\boldsymbol{I}_{2\times2}/3 & T^2\boldsymbol{I}_{2\times2}/2 \\ T^3\boldsymbol{I}_{2\times2}/6 & T^2\boldsymbol{I}_{2\times2}/2 & T\boldsymbol{I}_{2\times2} \end{bmatrix}$$

APPENDIX C
DERIVATION OF CT DISCRETE TIME PROCESS NOISE MATRIX AND TRANSITION MATRIX

The constant turning rate model admits the following relation between acceleration and velocity:

$$\ddot{p}_x = -\omega\dot{p}_y,\ \ddot{p}_y = \omega\dot{p}_y \tag{18}$$

Had we known the turn rate, this model would be linear, however realistically we do not know the turning rate and must append it to our state vector. Our state vector is : $\boldsymbol{x} = [p_x, p_y, \dot{p}_x, \dot{p}_y, \omega]$, where

$$\dot{\omega} = \tilde{\nu}_k \tag{19}$$

As in [12], we assume white noise acceleration on the velocities and the turn rate.

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -\omega & 0 \\ 0 & 0 & \omega & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} n \tag{20}$$

where n $\sim \mathcal{N}(0, D)$ with D=diag($[\sigma_a^2, \sigma_a^2, \sigma_\omega^2]$)

In order to discretize the system with exact linearization, we will assume the turn rate slowly varying and decouple the turn rate from the system.

Let x_1 be the first four states, $n_1 \sim \mathcal{N}(0, D_1)$

$$G_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, D_1 = \begin{bmatrix} \sigma_a^2 & 0 \\ 0 & \sigma_a^2 \end{bmatrix}, A_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -\omega \\ 0 & 0 & \omega & 0 \end{bmatrix} \tag{21}$$

We start by finding F_d, the discretized F matrix. From equation 9, we get:

$$F_k = \begin{bmatrix} 1 & 0 & \frac{i(e^{iT\omega} - e^{-i^T\omega})}{2\omega} & \frac{e^{iT\omega} - e^{-i^T\omega}}{2\omega} \\ 0 & 1 & \frac{-e^{iT\omega} - e^{-iT\omega}}{2\omega} & \frac{i(e^{iT\omega} - e^{-iT\omega})}{2\omega} \\ 0 & 0 & \frac{e^{iT\omega} + e^{-iT\omega}}{2} & \frac{i(-e^{iT\omega} + e^{-iT\omega})}{2} \\ 0 & 0 & \frac{i(e^{iT\omega} - e^{-iT\omega})}{2} & \frac{e^{iT\omega} + e^{-iT\omega}}{2} \end{bmatrix} \tag{22}$$

Using Euler's formula: $e^{ix} = cos(x) + isin(x)$, this simplifies to :

$$\begin{bmatrix} 1 & 0 & \frac{sinT\omega}{\omega} & \frac{-1+cosT\omega}{\omega} \\ 0 & 1 & \frac{1-cosT\omega}{\omega} & \frac{sinT\omega}{\omega} \\ 0 & 0 & cosT\omega & -sinT\omega \\ 0 & 0 & sinT\omega & cosT\omega \end{bmatrix} \tag{23}$$

We have a singularity in $\omega = 0$. The singularity can be mitigated by taking the limit when $\omega \rightarrow 0$.

$$\begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & cosT\omega & -sinT\omega \\ 0 & 0 & sinT\omega & cosT\omega \end{bmatrix} \tag{24}$$

For ω, we simply have:

$$\omega_k = \omega_{k-1} \tag{25}$$

The calculation of Q is nontrivial and is given in [12]:

$$\begin{bmatrix} \frac{T^3}{3}\sigma_a^2\boldsymbol{I}_{2\times2} & \frac{T^2}{2}\sigma_a^2\boldsymbol{I}_{2\times2} & 0 & \\ \frac{T^2}{2}\sigma_a^2\boldsymbol{I}_{2\times2} & T\sigma_a^2\boldsymbol{I}_{2\times2} & 0 & \\ 0 & & 0 & T\sigma_\omega^2 \end{bmatrix} \tag{26}$$

This model is of course inherently nonlinear as ω appears in the transition matrix. In our implementation, we implement an EKF to handle the nonlinearities.