Investigation of Adaptive Filtering Techniques for Tracking Vehicles in Videos

Lindsay White
University of California San Diego
9500 Gilman Drive, La Jolla, CA 92093
lkwhite@eng.ucsd.edu

Abstract

Visual object tracking in surveillance videos is an area of active research with many important applications. This paper investigates three popular adaptive-filter-based techniques for tracking vehicles: the Kalman, Unscented Kalman, and Particle Filters. It is hypothesized that non-linear filtering techniques will result in lower average pixel errors between predicted and detected vehicle centroid values, but this is not supported by the results of the experiments. Discussion of the key parameters of the filters and performance comparisons are presented.

1. Introduction

The problem of detecting and tracking objects in videos has long been an area of active research. The detection and tracking of vehicles have the potential to improve driver and road safety [12, 33]. Applications include counting vehicles [44], performing traffic-flow analysis [26, 31], and performing driver behavior analysis [31, 39].

In the context of vehicle tracking, there are two main categories of videos. The first is surveillance-style videos that may be taken from a static traffic camera on a road or at an intersection. The second is a video taken from a camera onboard a moving vehicle. The former type of video is usually used for monitoring and observing, while the latter is typically associated with autonomous vehicle or driver assistance applications.

Generally speaking, there are two interrelated problems that must be solved to perform vehicle tracking in videos. The two problems are those of vehicle detection, or determining the location of every vehicle in each frame, and vehicle tracking, or associating all detections of a single vehicle across frames. This paper focuses primarily on adaptive filtering solutions to the problem of vehicle tracking in surveillance-style traffic videos.

The remainder of the paper is organized as follows: Section 2 reviews related works and related problems. Section 3 provides the mathematical basis required for the experiments, which are described and discussed in Section 4. Section 5 contains concluding remarks and provides suggestions for future research directions.

2. Related Work

2.1. Vehicle Detection

Substantial work has been done on the topic of vehicle detection in images. As such, there are several solutions to identifying the location of vehicles.

For surveillance-style videos with stationary cameras, one popular solution is the use of background subtraction [4, 12, 16]. There are several techniques to implement background subtraction; the general approach is to identify regions of the video that are non-changing between frames and designate these pixels as the background. The regions that do change between frames are considered to be the moving vehicles of interest. One challenge with this approach is the inability to distinguish individual vehicles if they are tightly clustered - the background subtraction method is only able to detect that a region is moving, not how many vehicles are present or where the individual vehicles are located.

Popular solutions to overcome the limitations of background subtraction include use of Optical Flow [2, 4, 39] and machine learning models that are trained on various images of vehicles [7, 32, 39]. Machine learning models are currently among the most popular method for detecting vehicles in images due to their speed and accuracy [6].

2.2. Vehicle Tracking

The Kalman filter is a popular choice for performing vehicle tracking when the vehicular motion can be approximated with a linear model [22, 26, 35]. The Kalman filter is an adaptive filter that performs optimal Minimum Mean-Square-Error (MMSE) estimation on linear systems [20].

To model more complex motion dynamics, non-linear models are necessary. Several extensions to the Kalman filter have been formulated to allow non-linear filtering. Two of the most popular extensions are the Unscented Kalman

Filter [17, 40, 49] and the Particle Filter [21, 30, 32]. These filters are discussed in more detail in Sections 3 and 4. An interesting area of research is on methods to adaptively select between different filter-based tracking methods [37]. Other popular methods of tracking vehicles include kernel-based algorithms [4, 9] and hybrid machine learning-based approaches [7, 42].

2.3. Autonomous Vehicles

A related area of active research is in using machine vision to track vehicles from non-stationary cameras, such as those on autonomous vehicles. There are several complex problems that must be solved to perform tracking when the camera is in motion, but many of the basic concepts and techniques discussed in this paper are applicable to autonomous vehicle machine vision as well [13, 33, 39].

3. Methods

3.1. Algorithm Description

To initialize the algorithm, all vehicle detections in the first frame are considered the start of a track. Each track consists of the current centroid location of the vehicle and a filter, as described below. The proposed algorithm for performing vehicle tracking is as follows [28]:

1. For all current tracks, predict the location of the centroid in the next frame.
2. Advance to the next frame, and perform detection to determine vehicle centroid locations.
3. Associate the detections with the track predictions. Start new tracks and remove lost tracks as needed.
4. Using the associated track detections, perform filter correction.

3.2. Data

There are many datasets available for performing machine vision tasks on vehicles. One popular dataset is the University at Albany DEtection and TRACking (UA-DETRAC) dataset by Lyu *et al*. [24, 25, 45]. This dataset is a collection of real-world images collected from traffic cameras in China. Lyu *et al*. have annotated each frame of the videos with information about all vehicles present. For all vehicles visible in a frame, the annotation information includes a bounding box and centroid for the vehicle location, each vehicle's type and occlusion ratio, and several other pieces of metadata.

This project is less ambitious than vehicle identification. As such, the portions of the dataset that are used are the base images, the ground truth centroid locations, and the ground truth bounding boxes.

3.3. Vehicle Detection

For time and scope reasons, the ground truth centroids from the dataset are used as the detected locations in each frame. One benefit of this method is that errors in the detection do not influence the performance analysis of the filtering mechanism. This makes comparison between different filters straightforward.

3.4. Vehicle Motion Models

To perform vehicle tracking, a prediction must be made for the location of the vehicle in the next video frame. The predictions will be based on motion models [36], which seek to capture the dynamics of vehicle motion in equations. There are two motion models that will be studied in this paper: Constant Velocity, and Constant Turn Rate and Velocity.

3.4.1 Constant Velocity Model

The Constant Velocity (CV) motion model is based on the assumption that the centroid of the vehicle moves with independent motion at independent but constant velocities in the X and Y directions [3, 38].

The recursive kinematics equation that governs the relationship between the current position, x_n, and the previous position, x_{n-1}, is given in Equation 1, where v is the constant velocity and Δt is the time step between the two positions.

$$x_n = x_{n-1} + v_x \Delta t \tag{1}$$

This equation can be extended to two independent dimensions and rewritten in matrix form. The state model vector, s, and state transition matrix, F, are given in Equation 2. x and y are the X and Y locations of the centroid in pixels, and v_x and v_y are the velocities in the X and Y directions in pixels per frame, respectively.

$$s = \begin{bmatrix} x \\ v_x \\ y \\ v_y \end{bmatrix}, \quad F = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

The measurement matrix assumes that the only measurements available are the X and Y locations, and the velocity measurements are unavailable. The measurement matrix is given by Equation 3.

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3}$$

3.4.2 Constant Turn Rate and Velocity Model

The Constant Turn Rate and Velocity (CTRV) model, also known in the literature as the Coordinated Turn model, is based on the assumption that the vehicle is moving with a

constant speed and making a smooth turn in the X-Y plane at a constant rate [5, 38, 47, 48]. The state model vector, s, is given in Equation 4, and the update equations, F, are given in Equation 5. x, y, v_x, and v_y are defined as above. ω is the constant rate of turn in radians per frame.

$$s = \begin{bmatrix} x & v_x & y & v_y & \omega \end{bmatrix}^T \tag{4}$$

$$F = \begin{bmatrix} x_n + \frac{v_{x,n}}{\omega} sin(\omega\Delta t) + \frac{v_{y,n}}{\omega}(1 - cos(\omega\Delta t)) \\ v_{x,n} cos(\omega\Delta t) - v_{y,n} sin(\omega\Delta t) \\ y_n + \frac{v_{x,n}}{\omega}(1 - cos(\omega\Delta t)) + \frac{v_{y,n}}{\omega} sin(\omega\Delta t) \\ v_{x,n} sin(\omega\Delta t) + v_{y,n} cos(\omega\Delta t) \\ \omega \end{bmatrix} \tag{5}$$

Note that F is not a matrix, as it was in the case of the CV model. In this model, F is a group of non-linear equations to describe the transformation of the input states, s_n, to the states at the next time step, s_{n+1}.

As in the CV model, this model assumes that the only measurements available are the X and Y locations. The measurement matrix is given by Equation 6.

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{6}$$

3.5. Vehicle Prediction

3.5.1 Kalman Filter

Use of the Kalman filter is typically separated into two stages: a prediction stage followed by an update stage [20, 22, 35]. In the prediction stage, the current state and the state transition matrix are used to generate an estimate for the next state. In the update stage, the potentially noisy or incomplete measurement of the next state is combined with the estimate and the statistics of the system to provide an updated estimate of the next state.

The Kalman filter prediction equation, given in Equation 7, relates the state estimate at time $n+1$, s_{n+1}, to the state at time n through the state transition matrix, F, and the process noise, v_1. The estimate of the future state, $\hat{s}_{n+1}$, is the same equation without the additive noise term. The Kalman filter measurement equation, given in Equation 8, relates the observation at time n, y_n, to the state at time n through the observation matrix, C_n, and the measurement noise, v_2.

$$s_{n+1} = F s_n + v_{1,n} \tag{7}$$

$$y_n = C_n s_n + v_{2,n} \tag{8}$$

The two noise terms, v_1 and v_2, cannot be exactly predicted, but their covariance statistics can be utilized to combine the estimate and the measurement to generate a better estimate of the next state. The prediction and update equations are given in Equations 9 and 10, respectively. In these equations, Y_{n-1} is the set of all observations to time $n-1$, $\hat{s}_{n|Y_{n-1}}$ is the *a priori* state estimate at time n given all observations up to time $n-1$, $K_{n,n-1}$ is the *a priori* covariance estimate, $\hat{y}_{n|Y_{n-1}}$ is the estimate for the observation at time n, α_n is the innovation, R_n is the innovation covariance, G_n is the Kalman gain, $\hat{s}_{n|Y_{n-1}}$ is the *a posteriori* state estimate, and K_n is the *a posteriori* covariance matrix. $Q_{1,n}$ and $Q_{2,n}$ are the covariance matrices of the process noise, $v_{1,n}$, and the measurement noise $v_{2,n}$, respectively.

$$Predict: \begin{cases} \hat{s}_{n|Y_{n-1}} = F\hat{s}_{n-1|Y_{n-1}} \\ K_{n,n-1} = FK_{n-1}F^H + Q_{1,n-1} \end{cases} \tag{9}$$

$$Update: \begin{cases} \hat{y}_{n|Y_{n-1}} = C\hat{s}_{n|Y_{n-1}} \\ \alpha_n = y_n - \hat{y}_{n|Y_{n-1}} \\ R_n = CK_{n,n-1}C^H + Q_{2,n} \\ G_n = FK_{n,n-1}C^H R_n^{-1} \\ \hat{s}_{n|Y_n} = \hat{s}_{n|Y_{n-1}} + G_n\alpha_n \\ K_n = (I - FG_nC)K_{n,n-1} \end{cases} \tag{10}$$

3.5.2 Unscented Kalman Filter

The Unscented Kalman Filter (UKF) is an extension to the Kalman filter that allows the use of nonlinear state transition equations, first proposed by Julier and Uhlmann [17–19]. The basic intuition is to generate a set of "sigma" points that have a known Gaussian distribution. At each time step, the sigma points are processed through the nonlinear transition equation and a weighted sum of the transformed points is used to estimate statistics about the state, which is the variable of interest. The equations for the prediction and update steps, given in Equations 11 and 12, are very similar to the standard Kalman filter equations.

$Predict:$

$$\begin{cases} \hat{s}^{(i)}_{n|Y_{n-1}} = \hat{s} = F(s^{(i)}_{n-1|Y_{n-1}}) \\ \hat{s}_{n|Y_{n-1}} = \mu_n = \sum_{i=0}^{p} W^{(i)}\hat{s} \\ K_{n,n-1} = \sum_{i=0}^{p} W^{(i)}(\hat{s} - \mu_n)(\hat{s} - \mu_n)^T + Q_{1,n-1} \end{cases} \tag{11}$$

$Update:$

$$\begin{cases} \hat{y}^{(i)}_{n|Y_{n-1}} = \hat{y} = C(\hat{s}) \\ \hat{y}_{n|Y_{n-1}} = \beta_n = \sum_{i=0}^{p} W^{(i)}\hat{y} \\ \alpha_n = y_n - \hat{y}_{n|Y_{n-1}} \\ R_n = \sum_{i=0}^{p} W^{(i)}(\hat{y} - \beta_n)(\hat{y} - \beta_n)^T + Q_{2,n} \\ G_n = \sum_{i=0}^{p} W^{(i)}(\hat{s} - \mu_n)(\hat{y} - \beta_n)^T \\ W_n = G_n R_n^{-1} \\ \hat{s}_{n|Y_n} = \mu_n + W_n\alpha_n \\ K_n = K_{n,n-1} - W_n R_n W_n^T \end{cases} \tag{12}$$

Y_{n-1}, $\hat{s}_{n|Y_{n-1}}$, $K_{n,n-1}$, $\hat{y}_{n|Y_{n-1}}$, α_n, R_n, G_n, $\hat{s}_{n|Y_{n-1}}$, K_n, $Q_{1,n}$, and $Q_{2,n}$ are defined as above. W_n is the weight

vector used for combining the transformed points. There are p points and weights, with the individual weights denoted $W^{(i)}$. The individual transformed points are denoted $\hat{s}^{(i)}_{n|Y_{n-1}}$ and $\hat{y}^{(i)}_{n|Y_{n-1}}$ for the transformed state and observation estimates, respectively. F and C are now the transformation and observation functions, which may be vectors or equations depending on the model.

3.5.3 Particle Filter

The Particle Filter (PF) is another extension to the Kalman filter to allow the use of non-linear state transition equations. PFs originate in the field of physics, and come to the field of state estimation through an original "bootstrap filter" proposed by Gordon *et al*. [11]. The version described and implemented here is known as the Sampling-Importance-Resampling (SIR) filter [1, 8, 10].

Similarly to the UKF, the PF relies on sample points with known distributions and associated weights. However, the distributions are no longer constrained to be Gaussian. The sample points, also known as particles, are used to approximate the *a priori* and *a posteriori* probability density functions (pdfs) of the state conditioned on the observations. The state itself, s_n, is estimated as a weighted average of the particles.

Consider a set of p particles at the n^{th} iteration, denoted $\{x^i_n\}^p_{i=1}$. The goal of the PF is to approximate the *a posteriori* pdf of the particles given the observations through a discrete set of samples drawn from another pdf, termed the importance or proposal density. The importance density is often chosen to be the conditional pdf of the current particle given the previous particle. In this case, the i^{th} particle weight of the n^{th} iteration is approximated in Equation 13. The *a posteriori* pdf estimate is given in Equation 14.

$$w^i_n = w^i_{n-1} p(y_n|x^i_n) \tag{13}$$

$$p(s_n|y_n) = \sum_{i=1}^{p} w^i_n \delta(s_n - x^i_n) \tag{14}$$

After a number of iterations, the vast majority of particle weights tend towards being negligibly small while a small number remain substantial. This problem is called degeneracy, and leads to large amounts of processing power being used for particles that will contribute very little to the final estimate. To avoid this problem, the particles are resampled with replacement. In other words, the particles with the smallest weights are likely to be replaced by copies of the particles with larger weights, and the larger the weight of a given particle the more likely that particle is to appear multiple times.

The algorithm for the PF is given as follows:

1. Select p particles from the importance density and assign a weight to each one.
2. Normalize all the weights.
3. Resample the particles with replacement.
4. Reset all weights to be $1/p$.

3.6. Vehicle Tracking

When multiple vehicles are present in a given frame, it is necessary to associate the detections in the current frame with detections from previous frames. The method of association is based on Munkres' assignment algorithm [34].

Each track has an associated filter to predict the likely centroid location in the following frame. All filters generate their predictions, then detection is run on the new frame. A cost matrix is generated by calculating the distance in pixels between each predicted centroid and each detected centroid. Each row contains the distances between all detections and a given track prediction.

To use the matrix, the smallest value in each row is subtracted from all other values in the row. This results in a matrix with at least one zero per row. If all columns and all rows have exactly one zero, the locations of the zeros denote the pairings between tracks and detections. The case of multiple zeros per row or column is handled by simply choosing either zero, as choosing either is the same cost. The case of no zeros per column is solved by subtracting the smallest of the new values of the column (after the initial row subtraction) from all other values of the column, which will result in at least one zero, then proceeding as normal.

An extension of Munkres' algorithm is utilized to account for the options to not assign a detection to an existing track (the desired behavior when a new vehicle appears), or to not associate a given track with any of the detections (the desired behavior when a vehicle leaves the frame) [27]. In this version, the original cost matrix is extended to include a cost of not performing an assignment for each detection (when the detection is too far from all predictions) and for each track (when the track prediction is too far from all detections). This extension allows the algorithm to identify when new tracks for new vehicles should be created, and when tracks for vehicles that have left the frame should be removed.

4. Experiments

4.1. Experimental Setup

All experiments were performed on a computer with the following hardware setup: Intel Core i7@2.60Ghz with 6 cores, Intel UHD Graphics 630, and 16GB@2.67Ghz DDR4 RAM.

The main software tool used was MATLAB R2020b by The MathWorks, Inc [29]. The key toolboxes were the Im-

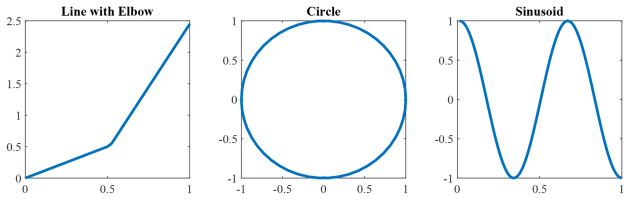

Figure 1. Shapes for Insight Experiments. From left to right, the shapes are a line with an elbow, a circle, and a sinusoid.

age Processing Toolbox v11.2, the Computer Vision Toolbox v9.3, and the Sensor Fusion and Tracking Toolbox v2.0.

The code used to perform the experiments discussed below in Sections 4.2, 4.3, and 4.4 is available at `https://github.com/KI6LZN/251b_project`.

4.2. Insight Experiments

4.2.1 Kalman Filter

The first set of experiments run on the Kalman filter are experiments with synthetic data to determine how the parameters of the filter affect its overall performance. There are three sets of synthetic data being utilized for these experiments: two lines with different slope values forming an elbow, a circle, and a cosine. The plots for these shapes are shown in Fig. 1.

Recalling the Kalman filter's prediction and update equations, Equations 9 and 10, there are four parameters that are not recursively defined and therefore must be selected or tuned: F, C, Q_1, and Q_2. As the Kalman filter requires a linear state transition function, the motion model utilized is the Constant Velocity (CV) model, described in Section 3.4.1. F and C are given from this model. Q_1 and Q_2 are the covariance matrices of the process noise and the measurement noise, respectively.

Because the CV motion model makes the assumption that the two dimensions, x and y, are independent in terms of both position and velocity, the two covariance matrices are both taken to be diagonal. The velocities are assumed to be constants, and there is assumed to be no covariance between velocity and position. Because the two directions are assumed to be identically distributed, the only values that need to be optimized are the values for the position covariance in both the process and measurement noise matrices, and the velocity covariance in the process noise matrix.

The first set of experiments is to determine the role that the three values that control the matrices have on the overall performance of the system as it tracks the three different shapes with no noise added. As would be expected, the linear Kalman filter performed very well when tracking the line with the elbow. The position measurement error primarily impacts how quickly the filter is able to adjust to the new slope after the elbow.

One interesting observation is that the particular values in the process error covariance matrix are not very important - the ratio between the location and velocity covariance values governs the behavior of the filter far more than the actual values. The location covariance value being larger results in the filter taking longer to align to the measurement values.

For tracking the circle, changing the measurement error changes the shape and diameter of the approximated circle, moving to a spiral shape at the extremes. The filter does a decent job at tracking the sinusoid shape, though the predictions overshoot the extrema. The measurement error value is particularly important for the sinusoidal track - when the measurement error is small, the predictions are very close to the true value, but when the measurement error value is large, the predictions greatly overshoot the extrema and lag behind the curve of the sinusoid. In both cases, similar observations regarding the importance of the ratios within the covariance matrix compared with the actual values are noted.

The second set of experiments repeats the first set of experiments with the modification that the measurements are corrupted by relatively-large magnitude Gaussian white noise. In the case of the line and the sinusoid, the average per-point error is smaller between the true value and the filter predictions, compared with the average per-point error between the true value and the noisy data point. This shows the value of using the filter instead of relying on the noisy data points. Tracking the circle is less successful, though in all three cases the main results from above still hold - the ratio of the values in the process covariance matrix are more important than the actual values. In contrast with the first set of experiments, the relative size of the measurement error compared with the values in the process matrix now play a role in the performance of the filter. The actual value of the measurement error itself is important as well: too small and the filter follows the errors quite closely, too large and the filter is unable to adjust to curves. This second case is particularly apparent in the sinusoid and circle cases.

4.2.2 Unscented Kalman Filter

The UKF can utilize either the Constant Velocity (CV) or the Constant Turn Rate and Velocity (CTRV) motion models, described in Section 3.4. Insight experiments are performed with both motion models, which allows further understanding of the the role the motion models play within the filter. As before, the tracks for the insight experiments are shown in Fig. 1, and there are two sets of experiments performed with noiseless and noisy data samples.

Using the CV model, the parameters that can be tuned for the UKF are the measurement and process covariance matrices, as before, and the distribution of the sigma points.

The results from the UKF are very similar to the results from the Kalman filter. The distribution of the sigma points have very little effect on the overall performance of the filter; this is to be expected as the state transition function is linear and therefore does not benefit as greatly from the unscented transformation compared with nonlinear functions. This holds true in both the noise-free and noisy experiments for all three tracks.

For the CTRV motion model, two additional parameters over the CV case can be tuned in the form of the initial turn rate, ω, and the covariance of ω. In the case of the noise-free experiments, the line and sinusoid tracks perform nearly the same as the CV model. The circle performs nearly seven times better in terms of average point error. For all three tracks, the ratio between the location and velocity covariances is more important than their values.

When tracking the line, the value of ω and its covariance value are two of the most important parameters. The measurement error determines the number of points required to recover from the error introduced at the elbow. Surprisingly, the distribution of the sigma points does not have a large influence on the performance of the filter. Similar observations are found in the case of the sinusoid.

As expected, the CTRV model performs much better when tracking a circle compared with the CV model. The distribution of the sigma points plays a slightly larger role in the overall performance of the system, compared with the other two tracks, but still a very small role overall. As with the other two tracks, ω and its covariance are the two important individual parameters. The measurement error plays a much smaller role than it did in the CV model.

The second set of experiments is again performed on the same synthetic data with added Gaussian noise. For all three tracks in the noisy case, the results of changing the various parameters are similar: the position and velocity covariance values are now more important than their ratio, and the sigma points play a larger role than they did in the noiseless case, but ω and its covariance remain the main parameters that impact overall performance. The noisy-data CTRV line is the only case in which the position covariance value being larger than the velocity covariance value resulted in better performance, and is also the only case in which the measurement error value is especially important to the overall performance of the system. In all three cases, the per-point error is lower between the filter predictions and the true value compared with the noisy data point and the true value. In comparison with the CV cases, the line tracking performed worse, the sinusoid tracking performed slightly better, and the circle tracking performed much better in terms of average per-point error - these are all intuitively satisfying results.

4.2.3 Particle Filter

Like the UKF, the PF can utilize either the Constant Velocity (CV) or Constant Turn Rate and Velocity (CTRV) motion models, and insight experiments using the three tracks from Fig. 1 are performed with both motion models. To ensure repeatability and enable comparison between different parameter choices, a seeded random number generator is used. Compared with the Kalman filter, the additional parameters that can be tuned for the PF are the number of particles and the resampling method.

In both the noise-free and noisy cases with the CV and CTRV motion models, one clear trend is that the number of particles is very important. When more particles are used, the estimates are much closer to the true values of the function, but the time required to process the tracks is dramatically increased. To give a scale comparison, the results summary in Table 1 shows the time taken to run the Kalman filter and UKF in seconds, while the time required for the PF is noted in minutes. Unlike the previous filters, the specific values within the correlation matrices are more important than their ratio. Interestingly, though, the trend of the velocity covariance being larger than the position covariance continues for the CV case but not the CTRV case.

With the CTRV motion model, the value of ω is less important than the total number of particles. When using a sufficiently large number of particles, the value of ω again becomes a dominant factor in the behavior of the filter.

There are four traditional methods for performing resampling of the particles: systematic, multinomial, stratified, and residual [23]. These are all different ways to perform the statistical replacement of lower-weight particles with higher-weight particles. An observation about the choice of resampling method is that the optimal method is more dependent on the shape of the track being predicted more than by the addition of the corrupting noise. However, the choice of resampling method does not necessarily hold between motion models.

4.3. Parameter Tuning

After performing the insight experiments, the parameters for the various filter must be tuned for use with the three UA-DETRAC sequences of interest. Sequence 39031 is a traffic camera looking at a straight section of road with cars moving straight toward or straight away from the camera. Sequence 40701 is a traffic camera looking at an intersection where cars are making a left turn from the main street, moving away from the camera. Sequence 40851 is a traffic camera looking at an intersection where cars are making a left turn from the cross street, moving towards the camera. All filters are tuned on each sequence individually.

To perform the parameter tuning, a function iterates through various options for the three parameters, and performs tracking on all vehicles in the frames. The metric

Filter	Motion Model	Sequence	Min Mean Error	Max Mean Error	Overall Mean Error	Time
Kalman	CV	39031	0.0605px	13.1244px	1.0696px	17 sec
		40701	0.3433px	2.2509px	0.9097px	25 sec
		40851	0.2246px	4.1441px	0.9458px	12 sec
UKF	CV	39031	0.0605px	13.1244px	1.0696px	83 sec
		40701	0.3433px	2.2509px	0.9094px	95 sec
		40851	0.2246px	4.1441px	0.9458px	58 sec
UKF	CTRV	39031	0.0861px	13.1244px	1.0733px	156 sec
		40701	0.3080px	2.2410px	0.9168px	153 sec
		40851	0.0950px	4.2593px	0.9457px	104 sec
PF	CV	39031	0.0266px	13.1296px	1.0807px	47 min
		40701	0.2653px	2.1615px	0.9186px	65 min
		40851	0.2088px	4.1114px	0.9530px	31 min
PF	CTRV	39031	0.0235px	14.0288px	1.1095px	35 min
		40701	0.3377px	2.1943px	0.9196px	47 min
		40851	0.2392px	4.1148px	0.9516px	23 min

Table 1. Summary of results.

by which the parameters are compared is the mean error in pixels, which is calculated as the mean distance between the estimated centroid location and the ground truth centroid location, averaged over all visible tracks. The parameters that result in the lowest mean error are taken as the optimal values.

Due to time and resource limitations, the number of particles used in the PF is limited to 10,000. This is the number of particles per filter, and there is one filter per vehicle detection. Larger quantities of particles will result in better estimates for the centroid locations.

4.4. Results

Following parameter tuning, the filters are run on the same three image sequences with the optimized parameters. The mean pixel errors across all frames, the single-frame minimum mean error value, the single-frame maximum mean error values, and the rounded time duration for the filters are shown in Table 1. Each frame is a color image of size 960-by-540 pixels (px).

For the three sequences listed in the table and described above, all filters tested perform very well. The average error between the predicted centroid and the ground truth centroid is on the order of one pixel for all three sequences. The differences between the different filters and the different motion models are relatively small. As mentioned above, the number of particles being utilized in each PF is limited due to time and resource constraints. This results in suboptimal estimates for the centroid locations, but the error values shown in the chart are still quite close to the values produced by the Kalman and Unscented Kalman filters.

As expected, the UKF with the CV motion model is very similar to the Kalman filter with the CV motion model. The UKF performs slightly better on average in sequence 40701, but overall the two filters are both performing very well across different videos. The UKF requires more processing time to arrive at the same result as the Kalman filter, so in the case of a linear motion model the Kalman filter is preferred. The PF requires substantially more processing time than the UKF, and does not yield accuracy improvements in the case of these videos with the number of particles being processed.

One interesting observation is that there is generally a peak in the centroid error when a vehicle begins to leave the frame. This is due to the filter attempting to predict the center location of the vehicle, while the ground truth centroid is only able to give information about the center of the visible portion of the vehicle. Large spikes in mean error tend to occur when there are multiple vehicles leaving the frame and few other vehicles present in the frame.

5. Conclusions

This paper has presented a comparison between three popular adaptive-filter-based solutions to the problem of tracking vehicles in stationary videos. It was hypothesized that the vehicle motion would be better predicted by a nonlinear filtering technique, but the experimental results show that the linear Constant Velocity motion model with the linear Kalman filter does an excellent job at describing and predicting the tracks. All filters with both motion models perform very well. On average the error between the ground truth centroid location and the predicted centroid location is

on the order of 1 pixel, which is remarkable in an image that is 960-by-540px.

The computation requirements for the Particle Filter (PF) and Unscented Kalman Filter (UKF) are substantially greater than the requirements for the Kalman filter. Though there are slight gains to be made in some cases by using a nonlinear motion model and nonlinear filter, the gains are sufficiently small that they are outweighed by practical concerns for computation time and resources, particularly in real-time applications.

Future studies on this topic may include exploring other motion models, such as the Constant Acceleration [43, 49] or the Curvilinear [5, 36] motion models. These motion models may yield better performance than the two motion models presented in this paper by considering different representations of the vehicle dynamics that govern how the vehicles move between frames. Further classes of adaptive filters that could be considered include Recursive Least-Squares (RLS) estimators [46] and adaptive correlation filters [15, 32]. Two particularly interesting areas of research lie in adaptively selecting the best filter and filter characteristics without manual parameter tuning for each dataset [37, 41], and in combining different adaptive filters to take advantage of their respective strengths [14]. Another interesting investigative paper may include tracking of vehicles via RADAR measurements, which are inherently nonlinear and may better illustrate the advantages and disadvantages of each filtering method.

References

[1] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[2] Sepehr Aslani and Homayoun Mahdavi-Nasab. Optical flow based moving object detection and tracking for traffic surveillance. *International Journal of Electrical and Computer Engineering*, 7:1252–1256, 2013.

[3] Nathanael L. Baisa. Derivation of a constant velocity motion model for visual tracking. *CoRR*, abs/2005.00844, 2020.

[4] S. R. Balaji and S. Karthikeyan. A survey on moving object tracking using image processing. In *2017 11th International Conference on Intelligent Systems and Control (ISCO)*, pages 469–474, 2017.

[5] R.A. Best and J.P. Norton. A new model and efficient tracker for a target with curvilinear motion. *IEEE Transactions on Aerospace and Electronic Systems*, 33(3):1030–1037, 1997.

[6] Hüseyin Seçkin Dikbayir and Halil İbrahim Bülbül. Deep learning based vehicle detection from aerial images. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 956–960, 2020.

[7] Mengyuan Dong, Xiong Zhang, and Changjun Chen. Object tracking via multi-layer convolutional features with adaptive correlation weighting. In *2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pages 144–148, 2019.

[8] Jos Elfring, Elena Torta, and René van de Molengraft. Particle filters: A hands-on tutorial. *Sensors*, 21(2), 2021.

[9] A. Elgammal, R. Duraiswami, D. Harwood, and L.S. Davis. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, 90(7):1151–1163, 2002.

[10] Simon Godsill. Particle filtering: the first 25 years and beyond. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7760–7764, 2019.

[11] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F Radar and Signal Processing*, 140(2):107–113, 1993.

[12] Raad Ahmed Hadi, Ghazali Sulong, and Loay Edwar George. Vehicle detection and tracking techniques: A concise review. *Signal & Image Processing: An International Journal*, 5(1):1–12, Feb 2014.

[13] Seung-Jun Han and Jeongdan Choi. Real-time precision vehicle localization using numerical maps. *ETRI Journal*, 36(6):968–978, 2014.

[14] B. He, L. Yang, K. Yang, Y. Wang, N. Yu, and C. Lu. Localization and map building based on particle filter and unscented kalman filter for an auv. In *2009 4th IEEE Conference on Industrial Electronics and Applications*, pages 3926–3930, 2009.

[15] Jie Hu, Huaxiong Zhang, Jie Feng, Hai Huang, and Hanjie Ma. A scale adaptive kalman filter method based on quaternion correlation in object tracking. In *2012 Third International Conference on Networking and Distributed Computing*, pages 170–174, 2012.

[16] Saba Joudaki, Mohd Shahrizal Bin Sunar, and Hoshang Kolivand. Background subtraction methods in video streams: A review. In *2015 4th International Conference on Interactive Digital Media (ICIDM)*, pages 1–6, 2015.

[17] Simon J. Julier and Jeffrey K. Uhlmann. A New Extension of the Kalman Filter to Nonlinear Systems. In Proc. of AeroSense: The 11th Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls, 1997.

[18] Simon J. Julier and Jeffrey K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

[19] Simon J. Julier, Jeffrey K. Uhlmann, and Hugh F. Durrant-Whyte. A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 45(3):477–482, 2000.

[20] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[21] W. L. Khong, W. Y. Kow, F. Wong, I. Saad, and K. T. K. Teo. Enhancement of particle filter approach for vehicle tracking via adaptive resampling algorithm. In *2011 Third International Conference on Computational Intelligence, Communication Systems and Networks*, pages 259–263, 2011.

[22] Yun Jip Kim, Yun Koo Chung, and Byung Gil Lee. Vessel tracking vision system using a combination of kaiman filter, bayesian classification, and adaptive tracking algorithm. In *16th International Conference on Advanced Communication*

Technology, pages 196–201, 2014.

[23] Tiancheng Li, Miodrag Bolic, and Petar M. Djuric. Resampling methods for particle filtering: Classification, implementation, and strategies. *IEEE Signal Processing Magazine*, 32(3):70–86, 2015.

[24] Siwei Lyu, Ming-Ching Chang, Dawei Du, Wenbo Li, Yi Wei, Marco Del Coco, Pierluigi Carcagnì, Arne Schumann, Bharti Munjal, Doo-Hyun Choi, et al. UA-DETRAC 2018: Report of AVSS2018 & IWT4S challenge on advanced traffic monitoring. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018.

[25] Siwei Lyu, Ming-Ching Chang, Dawei Du, Longyin Wen, Honggang Qi, Yuezun Li, Yi Wei, Lipeng Ke, Tao Hu, Marco Del Coco, et al. UA-DETRAC 2017: Report of AVSS2017 & IWT4S challenge on advanced traffic monitoring. In *Advanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference on*, pages 1–7. IEEE, 2017.

[26] Vamsi Krishna Madasu and M. Hanmandlu. Estimation of vehicle speed by motion tracking on image sequences. In *2010 IEEE Intelligent Vehicles Symposium*, pages 185–190, 2010.

[27] MathWorks, Inc. assignDetectionsToTracks. [Online] Available: https://www.mathworks.com/help/vision/ref/assigndetectionstotracks.html. (2012, July 17).

[28] MathWorks, Inc. Motion-Based Multiple Object Tracking. [Online] Available: https://www.mathworks.com/help/vision/ug/motion-based-multiple-object-tracking.html. (2012, July 17).

[29] The Mathworks, Inc., Natick, Massachusetts. *MATLAB version 9.9.0.1538559 (R2020b) Update 3*, 2020.

[30] Xue Mei and Haibin Ling. Robust visual tracking and vehicle classification via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2259–2272, 2011.

[31] Brendan Tran Morris and Mohan Manubhai Trivedi. Learning, modeling, and classification of vehicle track patterns from live video. *IEEE Transactions on Intelligent Transportation Systems*, 9(3):425–437, 2008.

[32] Reza J. Mozhdehi, Yevgeniy Reznichenko, Abubakar Siddique, and Henry Medeiros. Deep convolutional particle filter with adaptive correlation maps for visual tracking. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 798–802, 2018.

[33] Amir Mukhtar, Likun Xia, and Tong Boon Tang. Vehicle detection techniques for collision avoidance systems: A review. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2318–2338, 2015.

[34] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, 1957.

[35] Z. Qiu, D. An, D. Yao, D. Zhou, and B. Ran. An adaptive kalman predictor applied to tracking vehicles in the traffic monitoring system. In *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, pages 230–235, 2005.

[36] X. Rong Li and V.P. Jilkov. Survey of maneuvering target tracking. part i. dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1333–1364, 2003.

[37] Maria Scalzo, Gregory Horvath, Eric Jones, Adnan Bubalo, Mark Alford, Ruixin Niu, and P.K. Varshney. Adaptive filtering for single target tracking. *Proceedings of SPIE - The International Society for Optical Engineering*, 7336:11, 04 2009.

[38] Robin Schubert, Eric Richter, and Gerd Wanielik. Comparison and evaluation of advanced motion models for vehicle tracking. In *2008 11th International Conference on Information Fusion*, pages 1–6, 2008.

[39] Sayanan Sivaraman and Mohan Manubhai Trivedi. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1773–1795, 2013.

[40] Jiasheng Song and Guoqing Hu. Multi-feature visual tracking using adaptive unscented kalman filtering. In *2013 Sixth International Symposium on Computational Intelligence and Design*, volume 1, pages 197–200, 2013.

[41] Alvaro Soto. Self adaptive particle filter. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05, page 1398–1403, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.

[42] Jaya Krishna Sunkara, M Santhosh, Suresh Babu Cherukuri, and L. Gopi Krishna. Object tracking techniques and performance measures — a conceptual survey. In *2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, pages 2297–2305, 2017.

[43] Lennart Svensson and Joakim Gunnarsson. A new motion model for tracking of vehicles. *IFAC Proceedings Volumes*, 39(1):1376–1381, 2006. 14th IFAC Symposium on Identification and System Parameter Estimation.

[44] Birgi Tamersoy and J.K. Aggarwal. Counting vehicles in highway surveillance videos. In *2010 20th International Conference on Pattern Recognition*, pages 3631–3635, 2010.

[45] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *Computer Vision and Image Understanding*, 2020.

[46] Hadi Sadoghi Yazdi, Mahmood Fathy, and A. Mojtaba Lotfizad. Vehicle tracking at traffic scene with modified rls. In Aurélio Campilho and Mohamed Kamel, editors, *Image Analysis and Recognition*, pages 623–632, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[47] Xianghui Yuan, Chongzhao Han, Zhansheng Duan, and Ming Lei. Comparison and choice of models in tracking target with coordinated turn motion. In *2005 7th International Conference on Information Fusion*, volume 2, pages 6 pp.–, 2005.

[48] Xianghui Yuan, Feng Lian, and Chongzhao Han. Models and algorithms for tracking target with coordinated turn motion. *Mathematical Problems in Engineering*, 2014:1–10, 2014.

[49] Fan Zhang, Hong Li, Kalilou Kone, and Wei Zhang. Vehicle tracking based on nonlinear motion model. In Zengqi Sun and Zhidong Deng, editors, *Proceedings of 2013 Chinese Intelligent Automation Conference*, pages 403–410, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.