Multiple Models of Extended Kalman Filtering for Predicting Vehicle Location

Cesar Barrios, Henry Himberg, Yuichi Motai, Adel Sadek, School of Engineering, University of Vermont

Abstract—Two frameworks of Extended Kalman Filters (EKF) for predicting vehicle position with the aid of Global Positioning System (GPS) data are proposed to improve existing collision avoidance systems. A better prediction model for vehicle positions provides more accurate collision warnings in situations that current systems can not handle correctly. The Multiple Models Adaptive Estimation System (MMAE) algorithm and the Interacting Multiple Models Estimation (IMME) are applied to the integration of GPS measurements to improve the efficiency and performance. This paper evaluates the multiple-model systems in different scenarios and compares them to each other and to other systems before discussing possible improvements by combining it with other sensors for predicting vehicle location.

I. INTRODUCTION

A vehicle avoidance system by using sensors around the car is one of many ideas behind collision avoidance systems. Engineers have been chipping away at the staggering number of facilities for a long time by designing air bags and seat belts, stronger frames and special interior design to increase the safety of a car. However the only way to save far more lives is to keep cars from smashing into each other in the first place [8]. Previous research have experimented by placing sensors in the front of a vehicle to have the car’s computer maintain a safe distance from the car in front; sensors in the back to be activated only when in reverse to estimate space behind the car; and sensors by the side mirrors to detect objects in the blind spots and prevent collisions when lanes merge or cars change lanes [8].

Another study investigated a method to calculate the suggested safe distance to follow a car. It took into account safety zones around the car in case of maneuvering to avoid a collision. Results showed that the safe distance depended on the car and driver’s awareness, and illustrated how hard it is to get a good system to work well and not give warnings too often [2].

Manuscript received March 20, 2006.

C. Barrios is with the University of Vermont, Burlington, VT 05405 USA (phone: 802-878-9410; e-mail: cbarrios@uvm.edu).
H. Himberg is with the University of Vermont, Burlington, VT 05405 USA (e-mail: hhimberg@uvm.edu).
Y. Motai is with the University of Vermont, Burlington, VT 05405 USA (e-mail: ymotai@uvm.edu).
A. Sadek is with the University of Vermont, Burlington, VT 05405 USA (e-mail: asadeki@uvm.edu).

The next step for many of these warning systems is to implement automatic braking capabilities so that they do not need to rely on driver’s capabilities. In [7], researchers experimented with a dynamic model for brake control using a solenoid-valve-controlled hydraulic brake actuator system. They came up with a proposed brake control law that can provide the collision warning and collision avoidance vehicles with an optimized compromise between safety and comfort [7].

Systems like the ones described above are limited to line of sight for the sensors to detect other vehicles. Their accuracy is also inconsistent as speed and direction varies. The best way to prevent vehicle collisions is to know where vehicles are at all times, where they are heading, and where they will be in the future. Having this knowledge would allow systems to calculate if vehicles’ paths might intersect in the near future and warn a driver of a possible collision if it were a passive system, or apply the brakes automatically in case of an active system.

These types of complex and dynamic collision avoidance systems take into consideration the location of other vehicles nearby, even if not in line of sight. Researches like the one at the Kansai University of Japan [1] or the one by Miller and Huang [6] investigate the option of implementing inter-vehicle communication to be able to, through some judgment algorithm, identify if the trajectory of the vehicles will interest and possibly collide using Global Positioning System (GPS) data collected from the different vehicles. The methods used to estimate the intersection of the paths are somewhat simple and do not give accurate results in scenarios like curves where the estimated future position of the vehicles will not be a straight path.

It is clear that to have better collision avoidance systems we need a more accurate way to estimate the trajectory of the vehicles in all different scenarios. This is where the Kalman Filter (KF) comes into play. The KF has a long history of accurately predicting future states and has been applied to many different fields and this is why it has been chosen for this research.

The contribution of this paper is to investigate two Multiple Models Estimation algorithms applied to the integration of Global Positioning System (GPS) measurements. The different implementations of the Multiple Models Adaptive Estimation (MMAE) and the Interactive Multiple Models Estimation (IMME) algorithms that are designed to improve the efficiency and performance of the
algorithms and improve their performances are described. This paper first develops the initial implementation of the MMAE and IMME algorithms. The algorithms are then tested with real data obtained from GPS log files and compared against each other and against some simple estimations, such as the ones already being used in the previous studies mentioned earlier.

II. METHODOLOGIES

This research is based in the use of a GPS receiver to estimate projected path for a vehicle. Even though GPS receivers have some error in the calculated current position as is explained later, it is estimated to not cause major false estimations in this research.

Similar systems implement the use of other types of sensors to be able to get an accurate estimation, but this research looks into the possibility of using a cheap but accurate GPS receiver to do the same task and the results will provide the answer.

To evaluate the need for the extensive mathematical computations a Kalman Filter (KF) framework requires, some Simple Estimations (SE) will be define which do not take into account any previous data. These estimation models will not require much processor load, which would be excellent for a real-time system. But, even though the SE might be fast they might not be as accurate as using a Kalman Filter, and considering we are already using a GPS device which has some error, we do not want to loose more accuracy in the other steps.

The conventional Kalman filtering algorithm requires the definition of a dynamic and stochastic model. The dynamic model describes how the errors that are modeled develop over time, whereas the stochastic model describes the noise of the new measurements and the stochastic properties of the process being modeled.

There are several algorithms that exist to adapt the stochastic information on-line. These are termed adaptive Kalman filtering algorithms due to their ability to automatically adapt the filter in real time to correspond to the temporal variation of the errors involved.

One such algorithm is termed Multiple Models Adaptive Estimation (MMAE). The MMAE algorithm runs several Kalman filters in parallel, each operating using different dynamic or stochastic models. The MMAE algorithm is used to select either a single ‘best’ Kalman filter solution, or the algorithm can be used to combine the output from all the Kalman filters in a single solution. The obvious limitation of such an approach is the large computational burden imposed by running multiple Kalman filters. However, with improved processor technology, such an approach can now be considered even for real-time applications [13].

Another such algorithm is the Interacting Multiple Model (IMM) which, even though it works in a similar manner as the MMAE by running multiple Kalman filters in parallel, it is more mathematically involved and takes into account the probability of the next KF selection, making it more accurate than the MMAE in many scenarios.

A. GPS Devices

The Global Positioning System (GPS) is a satellite-based navigation system made up of a network of 24 satellites placed into orbit by the U.S. Department of Defense. Despite the cost of building a system like this, there are no subscription fees or setup charges to use GPS even though.

GPS satellites circle the earth twice a day in a very precise orbit and transmit signal information to earth. GPS receivers take this information and use triangulation to calculate the user's exact location.

Factors that can degrade the GPS signal and thus affect accuracy include the following [see table X for the amount of error each of them contributes] [get presentation reference name]:

- Ionosphere and troposphere delays — The satellite signal slows as it passes through the atmosphere. The GPS system uses a built-in model that calculates an average amount of delay to partially correct for this type of error.
- Signal multipath — This occurs when the GPS signal is reflected off objects such as tall buildings or large rock surfaces before it reaches the receiver. This increases the travel time of the signal, thereby causing errors.
- Receiver clock errors — A receiver's built-in clock is not as accurate as the atomic clocks onboard the GPS satellites. Therefore, it may have very slight timing errors.
- Orbital errors — Also known as ephemeris errors, these are inaccuracies of the satellite's reported location.
- Number of satellites visible — The more satellites a GPS receiver can "see," the better the accuracy. Buildings, terrain, electronic interference, or sometimes even dense foliage can block signal reception, causing position errors or possibly no position reading at all. GPS units typically will not work indoors, underwater or underground.
- Satellite geometry/shading — This refers to the relative position of the satellites at any given time. Ideal satellite geometry exists when the satellites are located at wide angles relative to each other. Poor geometry results
when the satellites are located in a line or in a tight grouping.

- Intentional degradation of the satellite signal —
  Selective Availability (SA) is an intentional degradation
  of the signal once imposed by the U.S. Department of
  Defense. SA was intended to prevent military
  adversaries from using the highly accurate GPS signals.
  The government turned off SA in May 2000, which
  significantly improved the accuracy of civilian GPS
  receivers.

<table>
<thead>
<tr>
<th>TABLE X</th>
<th>Sources of GPS Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURCE</td>
<td>ERROR CONTRIBUTION</td>
</tr>
<tr>
<td>Ionospheric delays</td>
<td>10 m</td>
</tr>
<tr>
<td>Tropospheric delays</td>
<td>1 m</td>
</tr>
<tr>
<td>PRN Code Noise</td>
<td>1 m</td>
</tr>
<tr>
<td>SV Clock</td>
<td>1 m</td>
</tr>
<tr>
<td>SV Ephemeris Data</td>
<td>1 m</td>
</tr>
<tr>
<td>Pseudo-Range Noise</td>
<td>1 m</td>
</tr>
<tr>
<td>Receiver Noise</td>
<td>1 m</td>
</tr>
<tr>
<td>Multi-Path</td>
<td>0.5 m</td>
</tr>
<tr>
<td>TYPICAL ERROR WITH BASIC GPS</td>
<td>15 m</td>
</tr>
</tbody>
</table>

There are many types of GPS devices, some are more accurate than others and are used in different scenarios. For example, the Carrier Phase Tracking devices only have an error up to 5 cm but are only available up to 30 km away from a reference station, take a long time to acquire a signal initially, and are very expensive. For this research we used the basic GPS device that has a poor accuracy, and a newer GPS that has some error correction techniques that provide a lot smaller error compared to the basic GPS.

1) Basic GPS Receiver
   The basic GPS receiver was one of the first devices introduced to the general public. Its accuracy was not very good as it could have up to 15 meters in error, but errors like this do not cause any problems when being used only for navigation systems that are used mainly as an aid to the driver.

   This research was started with this basic GPS receiver (Delorme Earthmate USB), and even though it did not look like it had errors of 15m a more accurate GPS receiver was acquired.

2) GPS Receiver with WAAS Error
   The Wide Area Augmentation System (WAAS) is an extremely accurate navigation system developed for civil aviation by the Federal Aviation Administration (FAA) in conjunction with the United States Department of Transportation (DOT). Its accuracy is less than 3 meters 95% of the time, and it provides integrity information equivalent to or better than receiver autonomous integrity monitoring (RAIM). This is achieved through 25 ground stations throughout the US and Alaska which measure the difference between their surveyed location and the GPS signal. These ground stations send the measured difference to a master relay station which sends the corrections to two geostationary satellites at the same longitudes as the East and West coasts. Those satellites beam the correction signal back to Earth, where WAAS-enabled GPS receivers apply the correction to their computed GPS position. Before WAAS, the U.S. National Airspace System (NAS) did not have the ability to provide horizontal and vertical navigation for precision approaches for all users at all locations, as ground-based systems are quite expensive. WAAS provides service for all classes of aircraft in all flight operations, including en route navigation, airport departures, and airport arrivals, including all-weather precision approaches throughout the NAS. Europe and Asia are conducting parallel efforts by way of the European Geostationary Navigation Overlay System (EGNOS) and the Japanese Multi-Functional Satellite Augmentation System (MSAS), respectively. [wikipedia.com]

   The WAAS GPS receiver also used in this research is a USB Holux WAAS/EGNOS GPS receiver with a SiRF starIII chip that has < 2.2m of error.

B. Estimation Models
   Because a vehicle can move in very different ways, to be able to estimate or predict its trajectory we need to define different models. Each model will be good for one specific set of conditions, so several models need to be defined to be able to cover most, if not all, possible scenarios a vehicle can be found in. Three models have been identified that seem to cover all vehicle’s behaviors: a vehicle traveling at constant velocity, or with constant acceleration, or with constant jerk (change in acceleration). These models, whether by themselves or a combination of them, should be able to cover a vehicle’s movement accurately. The different models provide a mathematical set of equations that can be used to estimate the vehicle’s future location after a set amount of time. But, even though three models have been identified, there are many ways they can be implemented. Two obvious ways these models can be implemented is without taking into account any previous data, which we will refer to as Simple Estimations (SE), and including previous data in the models to be able to obtain more accurate estimations, as in the case of Kalman Filters (KF).

1) Simple Estimations (SE)

   **Constant Velocity Model (CVM)**
\[ x_{k+1} = x_k + (x_k - x_{k-1}) \Delta k \]
\[ y_{k+1} = y_k + (y_k - y_{k-1}) \Delta k \]  
\hspace{1cm} (0.1)

**Constant Acceleration Model (CAM)**

\[ x_{k+1} = (Vx_k - Vx_{k-1}) \Delta k + x_k \]
\[ y_{k+1} = (Vy_k - Vy_{k-1}) \Delta k + y_k \]  
\hspace{1cm} (18)

**Constant Jerk Model (CJM)**

\[ x_{k+1} = (ax_k - ax_{k-1}) \Delta k^2 + Vx_k \Delta k + x_k \]
\[ y_{k+1} = (ay_k - ay_{k-1}) \Delta k^2 + Vy_k \Delta k + y_k \]  
\hspace{1cm} (19)

2) **Extended Kalman Filters (EKF)**

The Kalman Filter (KF) is a two-step probabilistic estimation process that is very popular in the robotics world as a tool to predict the next position of the robot in a linear system. Kalman filters are based on linear algebra and the hidden Markov model. The underlying dynamical system is modeled as a Markov chain built on linear operators perturbed by Gaussian noise. The state of the system is represented as a vector of real numbers. At each discrete time increment, a linear operator is applied to the state to generate the new state, with some noise mixed in, and optionally some information from the controls on the system if they are known. Then, another linear operator mixed with more noise generates the visible outputs from the hidden state.

The Kalman filter is a recursive estimator. This means that only the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state.

The Extended Kalman Filter (EKF) is similar to the KF but it can be used in non-linear systems because it linearizes the transformations via the Taylor Expansions. In the EKF the state transition and observation models need not be linear functions of the state but may instead be (differentiable) functions.

The EFK is a two step process: correct and predict:

a) **Correction (using measurement data)**

- Compute a gain factor (*Kalman Gain*) that minimizes the error covariance.

- Correct state estimation by adding the product of the Kalman gain and the prediction error to the prediction.

b) **Prediction (from the state variables)**

- Predict the next state from the current state using the system model. Assume the model is perfect (no process noise)

- Predict the error covariance of the next state prediction.

- Correct the error covariance estimation using the Kalman gain.

**Correct Step**

(a) Calculate the Kalman Gain

\[
S = HP_{k}H^{T} + VRV^{T}
\]
\[
K_{k} = \frac{P_{k}H^{T}}{S}
\]

(b) Correct the a priori state estimate

\[
x_{k}^{(a)} = K_{k}(z_{k}^{(a)} - h(x_{k}^{(a)},0))
\]

(c) Correct the a posteriori error covariance matrix estimate

\[
P_{k}^{(a)} = (I - K_{k}H)P_{k}^{(a)}
\]

**Prediction Step**

(a) Predict the state

\[
x_{k}^{(p)} = f(x_{k-1},0)
\]

(b) Predict the error covariance matrix

\[
P_{k}^{(p)} = AP_{k-1}A^{T} + WQW^{T}
\]

Fig. 3. Extended Kalman Filter

Notation:

- \( x \) state estimate
- \( z \) measurement data
- \( A \) Jacobian of the system model with respect to state
- \( W \) Jacobian of the system model with respect to process noise
- \( V \) Jacobian of measurement model with respect to measurement noise
- \( H \) Jacobian of the measurement model
- \( Q \) process noise covariance
- \( R \) measurement noise covariance
- \( K \) Kalman Gain
- \( P \) estimated error covariance
- \( \sigma_{p}^{2} \) prediction noise
- \( \sigma_{m}^{2} \) measurement noise

For our system the state vector for this system consists of four parameters, each one with an x and y component. Even though not all four of them are used in the models identified in this research such as: constant velocity, constant acceleration, constant jerk, all four parameters will be present in the models for an easier implementation.
\[
\begin{bmatrix} x_v \\ v_v \\ a_v \\ j_v \end{bmatrix} = \begin{bmatrix} \text{Position – of – vehicle} \\ \text{Velocity – of – vehicle} \\ \text{Acceleration – of – vehicle} \\ \text{Jerk – of – vehicle} \end{bmatrix}
\]  

(1)

The estimated \( P \) is used together with the Jacobian matrix \( H \) and the measurement noise covariance \( (R) \) together with the Jacobian matrix \( V \) to calculate the Kalman Gain.

\[
\begin{bmatrix} x, x_v, x, v_v, x, a_v, x, j_v \\ v, x_v, v, v_v, v, a_v, v, j_v \\ a, x_v, a, x_v, a, a_v, a, j_v \\ j, x_v, j, v_v, j, a_v, j, j_v \end{bmatrix}
\]

(2)

\[
\begin{bmatrix} x_v + v \\ v_v + v \\ a_v + v \\ j_v + v \end{bmatrix}
\]

(3)

\[
H = \left[ \frac{\partial h(x,v)}{\partial x} \right]_{t=0}^{t=(k-1)}
\]

(4)

\[
R = \sigma^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\]

(5)

\[
V = \left[ \frac{\partial h(x,v)}{\partial v} \right]_{t=0}^{t=(k-1)}
\]

(6)

Once the Kalman Gain \( (K) \) is calculated the system brings in the measured data \( (Z) \) to correct the predicted position and also the covariance error. Since this system can only measure location and speed from the GPS the other two parameters are set to zero for the system to calculate from prior data.

\[
Z_k = \begin{bmatrix} x_v \\ v_v \\ a_v \\ j_v \end{bmatrix}
\]

(7)

After correction of the previously predicted values the system is ready to predict the next position by using the state vector equations. The filter also estimates the error covariance of the estimated location by using the Jacobian matrix \( A \) and the Jacobian matrix \( W \) together with the Process noise covariance \( (Q) \) as follows.

\[
A = \left[ \frac{\partial f(x,w)}{\partial x} \right]_{t=0}^{t=(k-1)}
\]

(8)

\[
W = \left[ \frac{\partial f(x,w)}{\partial w} \right]_{t=0}^{t=(k-1)}
\]

(9)

\[
Q = \sigma^2 \cdot I
\]

(10)

To obtain an accurate estimation of the vehicle’s position three adaptive prediction (AP) algorithms are defined to account for the different possible scenarios. The state equations will be very different between the different models. The following three models account for most, if not all, the possible situations a vehicle could be found in.

### Constant Velocity Model (CVM)

\[
x_v(k) = x_v(k-1) + v_v(k-1) + w_v(k-1) + v_v(k) \cdot \Delta k
\]

(11)

\[
v_v(k) = v_v(k-1) + w_i
\]

(11)

\[
a_v(k) = 0
\]

(11)

\[
j_v(k) = 0
\]

(11)

### Constant Acceleration Model (CAM)

\[
x_v(k) = x_v(k-1) + v_v(k-1) + w_v(k-1) + v_v(k) \cdot \Delta k + w_v(k)
\]

(12)

\[
v_v(k) = v_v(k-1) + a_v(k) \cdot \Delta k + w_i
\]

(12)

\[
a_v(k) = a_v(k-1) + w_e
\]

(12)

\[
j_v(k) = a_v(k-1) + w_e
\]

(12)

### Constant Jerk Model (CJM)

\[
x_v(k) = x_v(k-1) + v_v(k-1) + a_v(k) \cdot \Delta k + w_v(k-1) + v_v(k) \cdot \Delta k + w_v(k)
\]

(13)

\[
v_v(k) = v_v(k-1) + a_v(k) \cdot \Delta k + w_i + j_v(k) \cdot \Delta k
\]

(13)

\[
a_v(k) = a_v(k-1) + w_e
\]

(13)

\[
j_v(k) = (1/2)(a_v(k-1) - a_v(k-2)) + w_e
\]

(13)

### C. Multiple Models

As the dynamic state of vehicles is highly variable over time, the model selected has to meet the conditions of very different situations. However, a solution based on the implementation of a unique model that fulfills the consistency requirements of scenarios with high dynamic changes, provokes unrealistic noise considerations when mild maneuvers are performed, diminishing the filter efficiency and impoverishing the final solution. Therefore, two different interactive multi-model filters have been developed and implemented to identify which one is better for this type of scenario.
The Multiple Model Adaptive Estimation (MMAE) algorithm is used to select either a single ‘best’ Kalman filter solution, or the algorithm can be used to combine the output from all the Kalman filters in a single solution. It uses only the previous evaluation of the individual filters used to identify which one should be used in the calculation of the next estimated location.

The Interacting Multiple Model (IMM) algorithm calculates the probability of occurrence for each of the individual filters and uses that information to identify which of the filters will be more predominant. This algorithm continues calculating the probability for each of the steps throughout the whole run; therefore, the IMM should be more accurate than the MMAE.

The obvious limitation of such approaches is the large computational burden imposed by running multiple Kalman filters. However, with improved processor technology, such an approach can now be considered even for real-time applications.

1) Multiple Models Adaptive Estimation (MMAE)

The classic MMAE uses a bank of m Kalman filters running simultaneously, each tuned to a different data set. The principle of the MMAE algorithm is described by figure 4 which shows that the new measurements, \( z_k \), are used in a bank of N Kalman filters. Each filter is configured to use either different stochastic matrices, or different mathematical models. The updated state estimates, \( x_k \), for the N Kalman filters are computed using the extended Kalman filter algorithm. The states from each filter are then combined by computing weight factors, and summing the weighted outputs. There are many different ways in which the weight factors can be computed. The one chosen for this system was the Dynamic Multiple Model method since not one filter will be the correct one at all times. This algorithm is described next.

The weight factors are computed using the recursive formula in equation (14), for N Kalman filters, where \( p_n(k) \) is the probability that the nth model is correct. The probability density function, \( f_n(z_k) \), is computed for each filter based on the innovation, \( v^T \cdot S^{-1} \cdot v \), and its corresponding covariance, \( S_k \), using the formula in equation (16).

\[
\begin{align*}
p_n(k) &= \frac{f_n(z_k) \cdot p_n(k-1)}{\sum_{j=1}^{N} f_j(z_k) \cdot p_j(k-1)} \quad (14) \\
S_k &= H \cdot P \cdot H^T \\
f_n(z_k) &= \frac{1}{\sqrt{(2\pi)^m |S_k|}} e^{-\frac{1}{2} (z_k - \mu)^T S^{-1} (z_k - \mu)} \quad (16)
\end{align*}
\]

The expression for the covariance in equation (15) reflects the filter’s estimate of the measurement residuals, not the actual residuals. This becomes clear when one examines the update expressions for “\( P \)” in the Kalman filter: “\( P \)” does not depend on the measurement residual. The effect of this is that the expression may indicate some small residual variance, when in fact at particular points in time the variance is relatively large. This is indeed exactly the case when one is simultaneously considering multiple models for a process—one of the models, or some combination of them, is “right” and actually has small residuals, while others are “wrong” and will suffer from large residuals. Thus when one is computing the likelihood of a residual for the purpose of comparing model performance, one must consider the likelihood of the actual measurement at each time step, given the expected performance of each model (14). This likelihood and probability variables allow the MMAE to determine which one of the filters defined should be used in the estimation of the next location, providing an accurate estimation.

2) Interacting Multiple Models

The basic idea of IMM is to simultaneously use several filters and mix their outputs to obtain a better estimation. This method allows coping with the uncertainty on the target motion by running a set of possible displacement modes at the same time. Even if the target is supposed to possibly be in each displacement mode, the probability that it is in each of them is considered and updated during execution of the IMM.

The IMM filter calculates the probability of success of each model at every filter execution scan, supplying a realistic combined solution for the vehicle behavior. These probabilities are calculated according to a Markov model for the transition between maneuver states, as detailed in [26]. To implement the Markov model, it is assumed that at each scan time there is a probability \( P_{ij} \) that the vehicle will make a transition from model state \( i \) to state \( j \). These
In Leigh A. Johnston and Vikram Krishnamurthy’s paper on “An Improvement to the Interacting Multiple Model (IMM) Algorithm” [1212] they describe the IMM as a recursive suboptimal algorithm that consists of four core steps:

- **Step 1)** Weight calculation;
- **Step 2)** Filter input calculation;
- **Step 3)** Kalman filtering;
- **Step 4)** Output combination.

**Initialization:**
As in any recursive system, the IMM algorithm first needs to be initialized before it can start its four step recursion.

\[ \hat{\lambda}_0(j) = \pi_j \]  
\[ \lambda_0(i) = \pi_i \]  
\[ \forall i \]  
\[ \forall j \]  
\[ (0.2) \]

**Recursion:**

- **Step 1)** Weight matrix calculation

**Step 2)** Filter input calculation

\[ \hat{x}_k^j = \frac{P_{k-1}^j x_k^j}{P_{k-1}^j} \]
\[ (0.3) \]

**Step 3)** Kalman filtering

The bank of s Kalman filters produce outputs \( \hat{x}_k^j \) and \( P_k^j \) where \( j = 1, \ldots, s \), according to the equations for the EKF in section B, part 2.

**Step 4)** Output combination

The combined state estimate \( \hat{x}_k \) and mixing weights are given by

\[ \hat{x}_k = \sum_{j=1}^{s} \hat{\lambda}_k(j) \hat{x}_k^j \hat{\lambda}_k(j) \]  
\[ (0.6) \]

The IMM mode switching process is assumed to be a Markov chain with known mode transition probabilities. The mode transition probabilities are actually selected as a design parameter when the filter is designed. The mode transition matrix \( \beta \) is \( N \times N \) matrix that determines the weighting during the mixing probability calculation. The off-diagonal values in \( \beta \) affect the dynamic performance of the filter. Lower values result in less overshoot but more RMS error; higher values have the opposite effect.

\[ \beta = \begin{bmatrix} 0.98 & 0.01 & 0.01 \\ 0.01 & 0.98 & 0.01 \\ 0.01 & 0.01 & 0.98 \end{bmatrix} \]  
\[ (17) \]

**Step 1)** Weight matrix calculation

The probability mixing calculation uses the transition matrix (17) and the previous iteration model probabilities (22) to compute the normalized mixing probabilities (18). The mixing probabilities are re-computed each time the filter iterates before the mixing step.

\[ \mu_{i,j} = \frac{1}{c} \cdot P_{i,k-1} \cdot \beta_{i,j} \]  
\[ (18) \]
\[ c_i = \sum_{j=0}^{M-1} \beta_{i,j} \cdot p_{j,k-1} \] (19)

• **Step 2) Filter input calculation**

The mixing probabilities are used to compute new initial conditions for each of the \( N \) filters. The initial state vectors are formed as the weighted average all the filter state vectors from the previous iteration (20). The error covariance corresponding to each of the new state vectors is computed as the weighted average of the previous iteration error covariance’s conditioned with the spread of the means (21).

\[
\hat{x}_{i,k-1}^0 = \sum_{j=0}^{M-1} \hat{x}_{j,k-1} \cdot \mu_{i,j} 
\] (20)

\[
P_{i,k}^0 = \sum_{j=0}^{M-1} H_{i,j} \left[ \hat{p}_{j,k-1} + \left( \hat{x}_{j,k-1} - \hat{x}_{i,k-1}^0 \right) \left( \hat{x}_{j,k-1} - \hat{x}_{i,k-1}^0 \right)^T \right] 
\] (21)

• **Step 3) Kalman filtering**

The bank of \( s \) Kalman filters produce outputs \( \hat{x}_k^j \) and \( P_k^j \) where \( j = 1, ..., s \), according to the equations for the EKF in section B, part 2.

• **Step 4) Output combination**

Once the new initial conditions are computed, the filter step generates a new state vector, error covariance and likelihood function (13) for each of the filter models. The probability update step then computes the individual filter probability as the normalized product of the likelihood function and the corresponding mixing probability normalization factor (19).

\[
p_{i,k} = \frac{1}{c} \cdot \Lambda_{i,k} \cdot c_i 
\] (22)

\[
c = \sum_{i=0}^{M-1} \Lambda_{i,k} \cdot c_i 
\] (23)

The output averaging step combines the \( N \) filter state vectors using the individual probabilities to form the output state vector (15). The error covariance of the output state estimate can also be computed (16)

MISSING LAST EQUATION THAT CALCULATES “x”

=================================================================================================

MISSING LAST EQUATION THAT CALCULATES “x”

=================================================================================================
III. EXPERIMENTAL RESULTS

The experimental setting for testing the different models described in section II needs a log file of GPS data that contains different scenarios, specially those currently causing problems in existing systems. Figure 5 shows the trajectory recorded. It has many turns and contains various changes in speed and direction. This data is perfect to use, and even though it will probably be very hard for the system to give accurate estimations in some sections, it will provide a real life situation where accidents could happen. There is also a short highway section to test the system at higher speeds with less turns also.

The trajectory shown in figure 5 will be divided into different scenarios: scenario 1 (back streets), scenario 2 (roads), scenario 3 (highways). Scenario 1 consists mainly of slow speeds but curvy streets, Scenario 2 consists in medium speeds with only some turns, and Scenario 3 is mostly high speeds on a highway. These scenarios were defined so it would be easier to evaluate, with more detail, the functionality of the estimation systems.

The language used for the reading of the GPS logs and implementation of the different filters was Visual Basic. No code was re-used except for the matrix manipulations piece. This language was chosen because there was source code available to get data directly from the GPS, instead of only log files, to be able to test the system in real-time. It was also chosen because the navigation software used in Figure 5, Mappoint 2004, can be imbedded in Visual Basic, allowing the software to display the estimated future location on the map also. Being able to look at the estimated points on an actual map makes it easier to visually inspect and present the system.

A. Accuracy of the GPS Devices

As described in the methodology section, GPS receivers have some error in the calculation of the current position of a vehicle. Since for this research there were two types of GPS receivers available, a good start is to evaluate them both and see how much better one is compared to the other. Their comparison will provide a useful way of evaluating which one is better for this type of applications.

1) Basic GPS Receiver

[TBD]

2) GPS Receiver with WAAS Error Correction

[TBD]

3) Comparison between 1 and 2

[TBD]

B. Analysis of the Estimation Filters

Because the Kalman Filters are so mathematically involved, requiring a lot of processing power, it is good to measure if their results are better than the SE and by how much. In any system it is always good to evaluate all of its parts to make sure they are all worth it, especially if the idea is to be commercially available.

1) Simple Estimations (SE)

To evaluate the Simple Estimations each of the models had to be coded. Setting up the three simple estimation models was very straight forward because the mathematical equations were simple. Running the SE on the same GPS log files recorded for this research provided some results where the error between the estimated location and the actual location were recorded in table II.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>SE 1-sec</th>
<th>SE 2-sec</th>
<th>SE 3-sec</th>
<th>Whole Route</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario 1</td>
<td>6.28</td>
<td>6.46</td>
<td>4.15</td>
<td>6.07</td>
</tr>
<tr>
<td>Scenario 2</td>
<td>10.22</td>
<td>12.38</td>
<td>6.98</td>
<td>10.87</td>
</tr>
<tr>
<td>Scenario 3</td>
<td>19.52</td>
<td>23.96</td>
<td>13.52</td>
<td>20.96</td>
</tr>
<tr>
<td>Scenario 1</td>
<td>13.76</td>
<td>13.10</td>
<td>8.59</td>
<td>12.68</td>
</tr>
<tr>
<td>Scenario 2</td>
<td>25.32</td>
<td>30.66</td>
<td>16.35</td>
<td>26.80</td>
</tr>
<tr>
<td>Scenario 3</td>
<td>63.91</td>
<td>73.27</td>
<td>43.90</td>
<td>65.89</td>
</tr>
<tr>
<td>Scenario 1</td>
<td>23.30</td>
<td>20.56</td>
<td>15.43</td>
<td>20.77</td>
</tr>
<tr>
<td>Scenario 2</td>
<td>46.12</td>
<td>47.36</td>
<td>30.62</td>
<td>44.55</td>
</tr>
<tr>
<td>Scenario 3</td>
<td>144.12</td>
<td>127.96</td>
<td>99.50</td>
<td>129.42</td>
</tr>
</tbody>
</table>

Units are in feet.

Table II shows how some models work better in the different scenarios. For the most part Scenario 3 will provide the most accurate results be cause it contains mostly constant
speeds with very little turns, allowing all models to estimate the next location with more precision than in the other scenarios.

2) Extended Kalman Filters (EKF)
To be able to evaluate the three different Kalman Filter Models, they had to be coded, tested and tuned individually to get as accurate estimations as possible. It is a given that one model will not be very accurate all the time on a real time GPS log, so one fake GPS log was created for each of the three models to exercise only one model at the time. By doing this it is possible to tune each of the filters individually knowing that the estimation should be as close as possible at all times.

Once the filters have been tuned they were individually run through the different scenarios, as it was done with the Simple Estimations, and the results were recorded in table I.

<table>
<thead>
<tr>
<th>EKF</th>
<th>Scenario 1</th>
<th>Scenario 2</th>
<th>Scenario 3</th>
<th>Whole Route</th>
</tr>
</thead>
<tbody>
<tr>
<td>CYM-1sec</td>
<td>6.35</td>
<td>6.43</td>
<td>4.19</td>
<td>6.08</td>
</tr>
<tr>
<td>CAM-1sec</td>
<td>6.29</td>
<td>6.47</td>
<td>4.16</td>
<td>6.08</td>
</tr>
<tr>
<td>CJM-1sec</td>
<td>6.32</td>
<td>6.53</td>
<td>4.18</td>
<td>6.13</td>
</tr>
<tr>
<td>CYM-2sec</td>
<td>13.80</td>
<td>12.98</td>
<td>8.65</td>
<td>12.64</td>
</tr>
<tr>
<td>CAM-2sec</td>
<td>13.80</td>
<td>12.99</td>
<td>8.65</td>
<td>12.65</td>
</tr>
<tr>
<td>CJM-2sec</td>
<td>13.81</td>
<td>13.02</td>
<td>8.65</td>
<td>12.67</td>
</tr>
<tr>
<td>CYM-3sec</td>
<td>23.37</td>
<td>20.15</td>
<td>15.57</td>
<td>20.59</td>
</tr>
<tr>
<td>CAM-3sec</td>
<td>23.37</td>
<td>20.16</td>
<td>15.58</td>
<td>20.60</td>
</tr>
<tr>
<td>CJM-3sec</td>
<td>23.38</td>
<td>20.18</td>
<td>15.58</td>
<td>20.62</td>
</tr>
</tbody>
</table>

Units are in feet.

Running the three filters together showed how, when one was very close to the real value, the other two were not that accurate. In some instances more than one filter was accurate, probably when speed changes or acceleration changes were very small. In other cases none of the three filters was accurate at all, probably because of an abrupt change in direction or even in speed. The system reads data from the GPS every one second, so it is possible, though not common, to have a big change occur during that one second. For the most part one second will not allow the speed to change by a big amount, allowing the filters to estimate the next location accurately.

3) Comparison between 1 and 2
With the data collected from the Simple models and the Kalman filters by running the same set of points, we are able to compare each model at a time and verify if the EKF models are more accurate given their good reputation and extensive mathematical equations.
Figure 7 shows the average error in feet for each of the filters. From this graph it is very clear to see which models are better than the other in the different scenarios, but we need to look at the whole route to be able to decide which set of filters performs better for this scenario. Of course the whole route column in Table I and II will change depending on the route recorded. The best way to select the best set of filters is to look at how they perform in the different scenarios and identify which scenario will be more predominant in the application it is intended for.

For our research we will choose the Kalman filters as they perform better than the Simple Estimation overall. This was expected because of the long history Kalman Filters have and their excellent estimations when implemented in a wide range of systems.

C. Multiple Models

1) Multiple Models Adaptive Estimation (MMAE)

2) Interacting Multiple Models Estimation (IMME)

3) Comparison between 1 and 2
IV. CONCLUSION

The Kalman Filters are a good choice for predicting a future vehicle’s positions. They performed well as the experimental results showed, and with the ability of being able to work together through a MMAE system, they are an excellent choice for a position estimation system compared to other simple systems being used [1].

The MMAE provided some very accurate estimations if the time gap remained small (1 second). The bigger the time gap the greater the inaccuracy. A gap of 1 second is not useful for a collision avoidance system as warning a driver about a possible collision 1 second before it happens would not allow for enough time to do anything to prevent it. The minimum time gap needed would be a 2 or 3 second time gap. As we already saw, looping the MMAE gives better results than just increasing the time gap, and even though the accuracy is reduce, it is still more accurate than the simple method used in some current researches where the velocity is assumed to be constant and a straight path is estimated for the vehicle.

The future work includes a combination of the MMAE framework and Geographical Information System (GIS) which contains the information of actual roads. Devices such as the Crossbow sensor accelerometer together with the AutoEnginuity ScanTool can also be used to rely on more accurate measurements of velocity and acceleration instead of extracting that information from location changes from the GPS. If the estimated value could be correlated to actual roads’ information, and the data is reliable, then any error could be corrected at any point in time to the actual road location, providing a much better estimation of the vehicle’s projected path, even for bigger time gaps.

ACKNOWLEDGMENT

The authors would like to thank IBM Microelectronics, Essex Junction, Vermont for the support and time used in this study.

REFERENCES


[24] Christopher Hilde, Terry Moore, Martin Smith, “Multiple Model Kalman Filtering for GPS and Low-cost INS integration”, Institute of Engineering, Surveying and Space Geodesy, University of Nottingham, 2004


[27] Yaakov Bar-Sholom, X. Rong Li, Thiagalingam Kirubarajan, “Estimation with Applications to Tracking and navigation”, Wiley and Sons, 2001