Predicting Vehicle Trajectory

Cesar Barrios, Yuichi Motai

Trajectory Estimations Using Smartphones

Publication details
Cesar Barrios, Yuichi Motai
Published online on: 01 Mar 2017

How to cite: Cesar Barrios, Yuichi Motai. 01 Mar 2017, Trajectory Estimations Using Smartphones from: Predicting Vehicle Trajectory CRC Press Accessed on: 21 Jan 2019

PLEASE SCROLL DOWN FOR DOCUMENT
3

Trajectory Estimations Using Smartphones*

3.1 Introduction

The overall function of ITS is to improve decision-making, often in real time, improving the operation of the entire transport system. This can go from systems with intelligent route planning implemented to avoid some specific type of traffic in certain areas [1], to keeping track of the position of the vehicle for infrastructure assessment [2], to systems designed to aid with the prevention of collisions between the vehicles [3,4]. For this study, the research focuses on evaluating the use of smartphones as an intermediate step to accelerate the implementation of vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I), which could be used to prevent collisions.

There are two main types of collision avoidance systems: self-sufficient and interactive systems. Self-sufficient systems are those that can obtain enough information from their own sensors, such as those in References 5 through 7, where they placed sensors around the vehicle to maintain a safe following distance or to detect vehicles in the surroundings. Interactive systems are those that, as the name implies, interact with the infrastructure and/or other vehicles, such as researched in References 8 through 10, where their systems send spatial information to nearby vehicles to estimate the probability of a future collision. While self-sufficient systems are limited to line-of-sight detection, the interactive systems account for scenarios farther ahead or even around corners or intersections by predicting and communicating the future estimated trajectories.

The V2V and V2I areas are being well researched these days [11–17], as the government is carefully evaluating the implementation of new technologies to make our roads safer (Figure 3.1). In an article published February 3, 2014 by the United States Department of Transportation (USDOT) [18], the National Highway Safety Administration announced its decision to begin taking the next steps toward implementing V2V technology in all new cars and trucks, after years of research and unprecedented coordination between industry and across government.

* This chapter is a revised version of the author’s paper in IEEE Transactions on Industrial Electronics. DOI: 10.1109/TIE.2015.2478415, approved by IEEE Intellectual Property Rights.
When the steps toward implementing V2V technology are defined, car and truck manufacturers will be mandated to enable this in their new vehicles. The challenge faced is that, because V2V relies on other vehicles nearby also supporting V2V technology, there will be a gap of many years when the V2V/V2I will not be able to show its true potential in improving road safety. In an article published by *Forbes* on March 14, 2013 [19], they calculated that the age of the average vehicle on the road is at a record high of 10.8 years, which means there are vehicles on the roads that are 20 years old. Keeping this in mind, it is a long time to wait to ensure full V2V/V2I reliability.

The scientific contribution of this research includes the evaluation of using the smartphone to predict future trajectories for a possible implementation as a temporary hook into the V2V/V2I infrastructure in older vehicles. Allowing drivers of older vehicles the possibility of taking advantage of this new technology would not only benefit them, but it would also benefit the rest of the V2V/V2I-enabled vehicles, as the number of vehicles participating in the system would be much greater. Smartphones are already being used in the transportation field, and one example is the mobile application DriveWell, created by Cambridge Mobile Telematics [20], where the smartphone’s built-in sensors are used to provide a driver safety scoring and tips on how to improve it.

Since this research wants to evaluate the use of a smartphone’s built-in sensors for a setup that could be used in a V2V/V2I system, it will focus on the prediction of a vehicle’s future trajectory, and compare the results with the use of more robust sensors mounted on a vehicle to predict the

**FIGURE 3.1**
Illustration for V2V and V2I from Reference 18.
same future trajectory. Given that multiple sensors will be used, some type of sensor fusion will be needed to use the different measurements in the prediction.

3.2 Sensor Fusion Techniques

The MSDF techniques are used in many diverse fields, although most of the literature addresses the fields of military target tracking or autonomous robotics [21]. The MSDF is required to combine and process data, which has been traditionally performed by some form of Kalman [22] or Bayesian filters. Furthermore, there can be two ways of setting up an MSDF system: centralized or decentralized. While a centralized approach suffices for most common scenarios where the sensors are synchronous, a decentralized approach is more convenient when the sensors should be treated independently [23–28], as with asynchronous sensors.

In Reference 29, the authors discuss one solution they have developed: the OATFA, which evolved from their earlier research on an ASTF [30]. They base their technique in the IMM algorithm, but replaced the conventional Kalman filters with their OATFA (which contains several Kalman filters of its own). The OATFA treats each sensor separately, passing the output from each to a dedicated Kalman filter, departing from the idea that the best way to fuse data is to deliver it all to a central fusion engine. The results from the IMM–OATFA show position estimation errors half of those of what the conventional IMM produces. However, as pointed out by the same authors in Reference 31, all measurement data must be processed before the fusion algorithm is executed. With a similar approach as the technique described above, the authors of Reference 32 create asynchronous holds, where, from a sequence of inputs sampled at a slow sampling rate, it generates a continuous signal that may be discretized at a high sampling rate. Despite the benefits of making the asynchronous system into a synchronous one by using these methods, restrictions arise where, if for some reason, a sensor is delayed in providing its data or is offline for a few cycles. The whole system needs to wait, as it is designed to work with certain data at specific rates (Figure 3.5).

To evaluate whether smartphones can properly estimate future trajectories to be considered as an option to fill in the V2V/V2I implementation gap, a system to estimate a future position of a vehicle will be set up to determine where the vehicle will be 3 s later, which is based on the average human reaction time of 1.5 s to stop a vehicle [33]. Looking at 3 s ahead of time was chosen as a reference point that is double the reaction time of an average human being. In reality, this number will probably vary in relation to the speed and the type of the vehicle since a faster or heavier vehicle will need more time to slow down, but it is taken as a reference point.
3.3 Position Estimation with Kalman Filters

For this research, the core method chosen to estimate a future position of a vehicle is the use of KF. The KF [34] was first proposed in the 1960s and it has been the most commonly used technique in target tracking and robot navigation since. The basic KF has been presented as a form of Bayesian filter [35], which is an optimal estimator for linear Gaussian systems. From a series of noisy measurements, the KF is capable of estimating the state of the system in a two-step process: correct and then predict [36–38].

The elements of this state vector \((x)\) are position, velocity, and acceleration of the vehicle. The position \((x_v)\) and velocity \((v_v)\) components of the state estimate have an \(x\) and \(y\) component to them (east-west and north-south directions), and the acceleration \((a_v)\) has an \(n\) and \(t\) component to it (normal and tangential acceleration). So, the full state vector matrix will be \(X = (x_v, x'_v, v_v, v'_v, a_n, a_t)\).

The estimated error covariance \((P)\) for the state estimate is based on the relationships between each of the elements to the others. The error covariance matrix is a dataset that specifies the estimated accuracy in the observation errors between all possible pairs of vertical levels.

Together with \(P\), the Jacobian matrix of the measurement model \((H)\), the measurement noise covariance \((R)\), and with the measurement noise \((\sigma_m)\) are used to calculate the Kalman gain \((K)\). Once the \(K\) is calculated, the system looks at the measured data \((Z)\) to identify the error of the predicted position and uses it to adjust \(P\).

The KF has a long history of accurately predicting future states of a moving target, and has been applied to many different fields [39–46], including transportation, which is why it was selected for this research. Because one KF estimates the future position of a vehicle using one spatial movement model, there is a need to set up several KFs to account for the different spatial states in which the vehicle can be found. Having multiple KF running in parallel at the same time requires a framework that can obtain one weighted answer.

This research opted for the use of IMM, which can calculate the probability of success of each KF model at every filter execution, providing a combined solution for the vehicle behavior [47–50]. These probabilities are calculated according to a Markov model for the transition between maneuver states, as detailed in Reference 51. To implement the Markov model, it is assumed that at each execution time, there is a probability \(p_{ij}\) that the vehicle will make the transition from model state \(i\) to state \(j\).

In Johnson and Krishnamurthy’s paper [52], they describe the IMM as a recursive suboptimal algorithm that consists of four core steps, interacting with the KF steps as illustrated in Figure 3.2.

The four-step IMM process starts with the calculation of the mixing probabilities, which uses the transition matrix and the previous iteration mode probabilities \(\mu_{k-1}(i)\) to compute the normalized mixing probabilities.
Trajectory Estimations Using Smartphones

The mixing probabilities are recomputed each time the filter iterates before the mixing step.

The second step uses the mixing probabilities, which are used to compute new initial conditions for each of the \( n \) filters. The initial state vectors are formed as the weighted average of all of the filter state vectors from the previous iteration \( x_{k-1}^j \). The error covariance corresponding to each of the new state vectors is computed as the weighted average of the previous iteration’s error covariance conditioned with the spread of the means \( P_{k-1}^j \).

The third step calculates mode-matched filtering; using the \( \hat{x}_{k-1}^j \) and \( P_{k-1}^j \), the bank of \( n \) Kalman filters produce outputs \( \hat{x}_k^j \), the covariance matrix \( P_k^j \), and the probability density function \( f_n(z_k) \) for each filter (\( n \)).

The fourth set of calculations begins once the new initial conditions are computed; the filtering step generates a new state vector, error covariance, and likelihood function for each of the filter models. The probability update step then computes the individual filter probability (\( \mu_k(j) \)) as the normalized product of the likelihood function and the corresponding mixing probability normalization factor.

The estimate and covariance combination is used for output purposes only \( \hat{x}_k = \sum_{j=1}^n \mu_k^j \cdot \hat{x}_k^j \); it is not part of the algorithm recursions.

3.4 Position Estimation Framework Using GPS and Accelerometer Sensors

For the evaluation of the use of smartphones to predict a future position of a vehicle, only those sensors common across all devices were selected, which, in this case are the GPS and accelerometer sensors. With the measurements obtained from these sensors, the future position estimation is obtained.
In this setup, the GPS sensor provides the location \((s_x, s_y)\), the velocity \((v)\), and the angle of direction \((\beta)\) using north as the zero. Then the accelerometer provides normal acceleration \((a_n)\) and tangential acceleration \((a_t)\). The different models to be used in this setup are defined below, which represents the different spatial states the vehicle can be found in Equations 3.1 through 3.3:

**Constant Location Model (CL)**

\[
s(k) = s(k-1) + \sigma_{p_s}
\]
\[
v(k) = 0
\]
\[
a(k) = 0
\]

**Constant Velocity Model (CV)**

\[
s(k) = s(k-1) + v(k-1) \cdot \Delta k + \sigma_{p_s}
\]
\[
v(k) = v(k-1) + \sigma_{p_v}
\]
\[
a(k) = 0
\]

**Constant Acceleration Model (CA)**

\[
s(k) = s(k-1) + v(k-1) \cdot \Delta k + \frac{1}{2} a(k-1) \cdot \Delta k^2 + \sigma_{p_s}
\]
\[
v(k) = v(k-1) + a(k-1) \cdot \Delta k + \sigma_{p_v}
\]
\[
a(k) = a(k-1) + \sigma_{p_a}
\]

The above three KF models are used as part of an IMM setup to merge each of the KF predictions and obtain one single predicted position, as described in Section 3.5. When not all sensor measurements are available to properly populate the three KF models, the DR approach described in Section 3.6 is used to handle the asynchronous data.

### 3.5 Multi-Sensor Data Fusion Setup

This research initially looks at predicting future positions by running the IMM system at 1 Hz with all vehicle-mounted (VM) sensors online in every iteration, and then running it at 10 Hz where some sensors are offline and measurements are missing in many of the iterations. A measurement could be missing, either due to the sensor not being able to take the measurement (system running at a faster frequency than the sensor, malfunction or no satellites in view for the GPS) or due to the processing CPU not being able to read/write fast enough. When a measurement is absent and the value is needed for the models, the missing
values are calculated from the measurements obtained by the remaining sensors based on previous real measurements, not estimations, when available.

In this research, because the system is running at the rate of the accelerometer (10 Hz), the missing measurements come from the GPS and ScanTool VM sensors, which run at only 1 Hz. Instead of using previously estimated values for the missing measurements, this setup uses the real measurements obtained from the accelerometer and uses the equations below to derive the missing measurements:

\[
\begin{align*}
    v(k) &= v(k-1) + a(k) \cdot \Delta k \\
    s(k) &= s(k-1) + v(k-1) \cdot \Delta k + \frac{1}{2} a(k) \cdot \Delta k^2
\end{align*}
\] (3.4)

Using this DR approach, as outlined in Figure 3.3, predictions are more accurate than waiting until all measurements are available again, or predicting these measurements a second time, using only previously estimated values. Only when all measurements are missing, which will be very unlikely, the system will use all the previously estimated values to feed the KF models and obtain the new position estimation.

### 3.6 Car and Smartphone Sensor Setup for a V2V/V2I System

As described toward the end of Section 3.2, this research will estimate the future position of the vehicle 3 s away using the framework described in
Section 3.4, which is something that could be shared with other vehicles (V2V) or with the infrastructure (V2I) as part of some collision avoidance system. As the communication standards are defined for the V2V and V2I systems, smartphones will fall into one of two different categories as briefly illustrated in Figure 3.4: (a) already supporting the technology required for the communication, or (b) needing an external device, illustrated as a black box, and connected via USB/Bluetooth/Wi-Fi to manage the communication aspect. This is briefly illustrated in Figure 3.4. Since smartphones and V2V/V2I are designed independently of each other, the use of an external device (black box) is more likely to be the case to enable smartphones to participate in a V2V/V2I system.

The VM sensors specifically set up for this specific task will be used, like manufacturers will implement in their vehicles. Smartphone (SP) sensors will also need to be evaluated when used for position estimation to determine if they yield similar results.

To properly evaluate if smartphones can be used in a V2V/V2I system, this research plans to set a baseline by running the VM measurements through the position estimation framework defined in Section 3.4, and calculate the position errors in the estimations by comparing them to the actual GPS data. Once the baseline is established and a determination of what are the amounts of prediction errors obtained, the individual SP measurements will be fed into the same position estimation framework and the error will be calculated in the position estimations. This research can then proceed to compare the results between the different sensors used and evaluate whether the smartphones’ built-in sensors yield similar prediction errors or not.

The setup on the VM sensors consists of a Garmin 16HVS GPS receiver running at 1 Hz and a Crossbow 3-axis accelerometer running at 10 Hz. An AutoEnginuity ODBII ScanTool (which obtains the velocity from the vehicle’s internal system at 1 Hz) is also available, but it will not be used in this evaluation because the smartphones this research is using do not have a way of connecting into the ODBII system. The data from the sensors used are postprocessed from the different log files recorded on an earlier date, and matched based on time stamps. Since these VM sensors were mounted on a van from the University of Connecticut, they will be labeled as UConn data throughout this research.

![FIGURE 3.4](image)

(a) SP supporting V2V/V2I communication technology and (b) SP needing external device (black box) for communication.
For the smartphones used in this evaluation, some were selected from different manufacturers and at different price ranges, to identify if there is some limitation on which ones can accurately predict the future trajectory of a vehicle. Also, smartphones are used with different operating systems as well to improve the evaluation experiment and take that into account as well. They were mounted securely on the vehicle to ensure the accelerometer readings truly reflect the dynamics of the vehicle. Because several smartphones were running at the same time, they were mounted in the trunk where they would still have a clear view of the sky, as shown in Figure 3.5, but a more common implementation would be to mount only one of them on the dashboard. Figure 3.5 shows six smartphones, but one of them did not record any GPS data, so it had to be removed from this experiment. The smartphones used in the evaluation of accurately predicting future trajectories are listed in Table 3.1.

All smartphones listed above have a built-in accelerometer sensor that can take measurements at 10 Hz, but no details were found on their model or sensitivity. These smartphones also have a built-in GPS sensor, and only very basic information was found about them. iPhone 5 has an A-GPS/GLONASS sensor, while the other four smartphones do not have support for Global

<table>
<thead>
<tr>
<th>TABLE 3.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartphone Specs</td>
</tr>
<tr>
<td>Manufacturer</td>
</tr>
<tr>
<td>Alcatel</td>
</tr>
<tr>
<td>HTC</td>
</tr>
<tr>
<td>LG</td>
</tr>
<tr>
<td>Apple</td>
</tr>
<tr>
<td>Apple</td>
</tr>
</tbody>
</table>

Note: Details about these smartphones obtained from http://www.gsmarena.com.

<sup>a</sup> Subsidized price; this model could not be purchased without a contract; the real price could be two or three times more.
Navigation Satellite System by the Russians (GLONASS). Also, both Apple smartphones can take measurements from the GPS sensor at 10 Hz, while the other three smartphones can only take measurements at 1 Hz.

Some smartphones also have a three-axis gyro sensor and a compass, which could be used as well to better estimate a position of a vehicle, but to match more closely the sensors mounted on the vehicle, and have a more equal comparison, they were not used in this experiment.

The measurements from the internal sensors of iPhone smartphones are recorded by running the SensorLog v1.4 application written by B. Thomas. The sensors’ measurements on the Android smartphones are recorded using the Data Recording v1.0.2.0 application written by T. Wolf. The data used are labeled by smartphone manufacturer name, except when there are more than one device per manufacturer, in which case the data were labeled by model name.

To properly exercise the position estimation framework described in Section 3.4, the route shown in Figure 3.6a for this evaluation was selected, which contains several curves (smooth and sharp) and straight paths, driven at different speeds in the larger and smaller roads. There were also some traffic lights on the way, and even a U-turn, providing also some stop-and-go scenarios. The route driven, shown in Figure 3.6a, is approximately 44 km long and takes about 45 min to drive all of it.

It is important to keep in mind that, despite having all SP devices secured on the same area of the vehicle as shown in Figure 3.5, their sensors’ sensitivity and tolerances are not the same, as visually shown in Figures 3.7 and 3.8.

Figure 3.7 shows the GPS readings of all SP and VM sensors, and, despite most of them being close to each other, they are not identical, and, therefore, will introduce some more error into the system when using the VM GPS as

![FIGURE 3.6](image_url)

Map of the recorded route near the University of Connecticut: (a) whole trajectory and (b) selected curves.
FIGURE 3.7
GPS measurements.

FIGURE 3.8
Accelerometer normal and tangential measurements.
a baseline. And, similarly, Figure 3.8 shows the accelerometer measurements for each of the devices used in this research, where it can be observed that some devices have a large variation of measurements compared to others.

3.7 Evaluation Criteria

To evaluate whether smartphones can properly predict a future trajectory and be considered as a possible solution to fill in the V2V/V2I implementation gap, the position estimation error between the VM sensors (UConn) and the SP sensors was selected using the same KF models and IMM framework. To start with, the position estimation error between both setups will be evaluated for the whole trajectory recorded. Also, since position estimation errors tend to increase during non-straight paths, this research will also divide the trajectory recorded into smooth and sharp curves. To determine whether a set of consecutive points in the trajectory is a curve or a straight line, the change in the heading (angle) between the current heading and the heading 2 s before was looked at; if more than 5°, then it was defined as a curve. And, to determine if the curve is a sharp one, the change has to be greater than 20°, otherwise it was defined as a smooth curve. For the selected curve shown in Figure 3.6b, the results of categorizing the trajectory based on heading is shown in Figure 3.9.

To calculate the position estimation error in each step, this research will compare the estimated position to the actual position measured by the GPS

![Figure 3.9](image-url)  
Classification of trajectory segments.
3 s later. This will allow a dataset of calculated errors to be built for the whole trajectory, which then can be divided into the route sections described in the previous paragraph. This research will look at average prediction errors and RMS prediction errors using Equation 3.5 to try to evaluate whether the sensors built into smartphones can properly fill in the V2V/V2I implementation gap:

\[
\text{RMS} = \sqrt{\frac{\sum_{k}^{K} (err_k - err_{k-1})^2}{(K-k)}}
\]  

(3.5)

For this experiment, a tolerance of 10% from the position estimation errors obtained from the VM sensors will be used; therefore, if a smartphone yields more than 10% higher estimation error than the VM sensors, then it will be concluded that such a smartphone cannot properly predict a future trajectory, and hence could not be considered as a possible temporary solution to fill in the V2V/V2I implementation gap.

3.8 Experimental Evaluation

3.8.1 Dataset Characteristics

The characteristics of the complete recorded dataset are shown in Table 3.2, where the mean and standard deviation for the position difference between each second, velocity, normal acceleration, and tangential acceleration are displayed.

Looking at Table 3.2, it is quickly noticed that the values between the distance and velocity columns are very similar, as expected, because this research is measuring the position change every 1 s. Also, as mentioned in Section 3.6, the UConn data were obtained on an earlier date, so it can be seen that, overall, the University of Connecticut van was driven a little bit faster than the vehicle with the smartphones. Also, because all five smartphones were in the same vehicle, their sensor measurements should have been very similar, which is not the case in several places. For example, for sharp curves, the two iPhones seemed to be moving at a much higher speed than the other three devices, while during straight paths they seemed to be moving a little slower than the rest. The tangential acceleration for all devices seems to be fairly consistent across all devices, while the normal acceleration is not as consistent, especially when smooth and sharp curves were observed, which could imply that some sensors are more sensitive than others.
### 3.8.2 Position Estimation Setup

To set up the IMM, it is necessary to calculate the transition probability matrix, so the GPS position measurements for the whole trajectory shown in Figure 3.6a were used. From this full GPS log that contained multiple scenarios, it was determined which transition occurs frame by frame by comparing the actual measurements from the GPS to the smoothed measurements. The smoothing of the data was done with a rolling window using a combination of median smoothing, split the sequence, and Hann’s sequence, which removed any abrupt changes from the data. The type of spatial change, such as no change, a constant change, and so on, determines each transition. Similarly, by calculating the covariance of the differences in

<table>
<thead>
<tr>
<th>Device</th>
<th>Distance (m)</th>
<th>Velocity (m/s)</th>
<th>Normal Acceleration (m/s²)</th>
<th>Tangential Acceleration (m/s²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whole trajectory</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UConn</td>
<td>18.14 ± 8.99</td>
<td>17.95 ± 8.45</td>
<td>−0.17 ± 0.58</td>
<td>0.61 ± 0.60</td>
</tr>
<tr>
<td>Alcatel</td>
<td>16.31 ± 8.40</td>
<td>16.30 ± 7.95</td>
<td>−0.12 ± 0.73</td>
<td>0.66 ± 0.95</td>
</tr>
<tr>
<td>HTC</td>
<td>16.21 ± 8.24</td>
<td>16.29 ± 8.01</td>
<td>−0.09 ± 0.56</td>
<td>0.64 ± 0.68</td>
</tr>
<tr>
<td>LG</td>
<td>16.79 ± 9.13</td>
<td>16.45 ± 7.83</td>
<td>−0.18 ± 0.58</td>
<td>0.61 ± 0.64</td>
</tr>
<tr>
<td>iPhone 3GS</td>
<td>16.59 ± 9.37</td>
<td>16.21 ± 7.96</td>
<td>−0.22 ± 0.74</td>
<td>0.51 ± 0.88</td>
</tr>
<tr>
<td>iPhone 5</td>
<td>16.25 ± 9.09</td>
<td>16.24 ± 8.18</td>
<td>−0.11 ± 0.79</td>
<td>0.62 ± 0.82</td>
</tr>
<tr>
<td>UConn</td>
<td>19.43 ± 8.29</td>
<td>19.02 ± 8.14</td>
<td>−0.16 ± 0.48</td>
<td>0.55 ± 0.57</td>
</tr>
<tr>
<td>Straight paths</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alcatel</td>
<td>18.76 ± 6.92</td>
<td>18.72 ± 6.70</td>
<td>−0.12 ± 0.72</td>
<td>0.64 ± 0.95</td>
</tr>
<tr>
<td>HTC</td>
<td>18.40 ± 6.56</td>
<td>18.38 ± 6.51</td>
<td>−0.10 ± 0.54</td>
<td>0.62 ± 0.57</td>
</tr>
<tr>
<td>LG</td>
<td>18.98 ± 7.74</td>
<td>18.42 ± 6.65</td>
<td>−0.18 ± 0.56</td>
<td>0.59 ± 0.50</td>
</tr>
<tr>
<td>iPhone 3GS</td>
<td>16.96 ± 9.04</td>
<td>16.63 ± 7.77</td>
<td>−0.20 ± 0.74</td>
<td>0.50 ± 0.88</td>
</tr>
<tr>
<td>iPhone 5</td>
<td>16.23 ± 9.24</td>
<td>16.23 ± 8.29</td>
<td>−0.10 ± 0.78</td>
<td>0.60 ± 0.73</td>
</tr>
<tr>
<td>UConn</td>
<td>16.19 ± 9.26</td>
<td>15.81 ± 8.03</td>
<td>−0.29 ± 0.89</td>
<td>0.74 ± 0.63</td>
</tr>
<tr>
<td>Smooth curves</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alcatel</td>
<td>14.66 ± 7.90</td>
<td>14.58 ± 7.36</td>
<td>−0.11 ± 0.75</td>
<td>0.73 ± 0.98</td>
</tr>
<tr>
<td>HTC</td>
<td>12.76 ± 8.32</td>
<td>12.92 ± 7.99</td>
<td>−0.07 ± 0.64</td>
<td>0.78 ± 0.91</td>
</tr>
<tr>
<td>LG</td>
<td>14.01 ± 9.28</td>
<td>13.53 ± 7.65</td>
<td>−0.21 ± 0.64</td>
<td>0.68 ± 0.82</td>
</tr>
<tr>
<td>iPhone 3GS</td>
<td>12.47 ± 11.31</td>
<td>11.99 ± 8.49</td>
<td>−0.34 ± 0.70</td>
<td>0.58 ± 0.87</td>
</tr>
<tr>
<td>iPhone 5</td>
<td>17.99 ± 6.64</td>
<td>17.86 ± 5.83</td>
<td>−0.16 ± 0.89</td>
<td>0.82 ± 1.29</td>
</tr>
<tr>
<td>Sharp curves</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UConn</td>
<td>8.52 ± 9.17</td>
<td>9.91 ± 7.65</td>
<td>−0.10 ± 0.89</td>
<td>1.01 ± 0.68</td>
</tr>
<tr>
<td>Alcatel</td>
<td>7.13 ± 9.60</td>
<td>7.49 ± 8.35</td>
<td>−0.16 ± 0.79</td>
<td>0.63 ± 0.88</td>
</tr>
<tr>
<td>HTC</td>
<td>5.29 ± 9.05</td>
<td>5.97 ± 8.68</td>
<td>−0.07 ± 0.64</td>
<td>0.59 ± 0.94</td>
</tr>
<tr>
<td>LG</td>
<td>5.48 ± 8.86</td>
<td>7.09 ± 8.01</td>
<td>−0.16 ± 0.66</td>
<td>0.59 ± 1.03</td>
</tr>
<tr>
<td>iPhone 3GS</td>
<td>16.30 ± 12.13</td>
<td>13.50 ± 9.49</td>
<td>−0.15 ± 1.05</td>
<td>0.70 ± 0.89</td>
</tr>
<tr>
<td>iPhone 5</td>
<td>12.00 ± 8.54</td>
<td>12.18 ± 8.54</td>
<td>−0.50 ± 1.04</td>
<td>0.62 ± 1.58</td>
</tr>
</tbody>
</table>

Note: Values represent median ± standard deviation of all sensor measurements collected by each device (∼25,000 data points).
the measurements to each other, the measurement noise covariance matrix (R) was obtained. And last, by calculating the covariance of the differences in the measurements compared to their respective x and y components, the process covariance noise (Q) for each KF was obtained. From this type of information, calculating the frequency the vehicle changes from one state to another, the transition probability matrix is derived.

Next, each of the devices was run through the same IMM system using the KF models described in Section 3.4, and for each new measurement obtained from any of the sensors, the system predicts the position of where the vehicle is going to be 3 s later in time.

### 3.8.3 Evaluation of Position Estimation Error by System Rate

This part of the experiment is to make sure the system defined in Section 3.5, which runs at the rate of its fastest sensor, yields better results than running the system at the rate of its slowest sensor. First, the VM sensor’s dataset (UConn) is run through the IMM system at the rate of 1 Hz when all sensors are online at each iteration. Then, the same dataset is run through the same IMM system but running at 10 Hz, so now missing values from offline sensors are calculated based on measurements from online sensors using a DR approach. For this part of the experiment, all of the VM sensors available were used, including the AutoEnginuity ODBII ScanTool.

Accuracy in predicting where the vehicle will be 3 s later in time is the factor to observe to be able to evaluate if running the IMM system at the rate of its fastest sensor yields smaller errors. Calculating the RMS prediction error by comparing each predicted position with each actual GPS measurement 3 s late in time, Table 3.3 is compiled.

Taking a quick look at Table 3.3, it can be observed that the RMS prediction errors for the system running at 10 Hz are smaller than when the system is run at 1 Hz, especially during curves. Even during straight paths, some improvements can be observed because the system is detecting a change in the vehicle’s speed faster, and can accordingly recalculate its prediction.

<table>
<thead>
<tr>
<th>TABLE 3.3</th>
<th>RMS Prediction Error (3 s Ahead)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UConn at 1 Hz</td>
</tr>
<tr>
<td>Whole trajectory</td>
<td>1.68 ± 3.21</td>
</tr>
<tr>
<td>Straight paths</td>
<td>1.27 ± 2.66</td>
</tr>
<tr>
<td>Smooth curves</td>
<td>2.60 ± 3.73</td>
</tr>
<tr>
<td>Sharp curves</td>
<td>3.43 ± 4.98</td>
</tr>
</tbody>
</table>

*Note: Contains values representing median prediction error in meters ± standard deviation.*
3.8.4 Evaluation of Position Estimation Error by Device

Now that it is shown that running this research’s system at the rate at the fastest sensor yields better predictions of future positions, all smartphones are also run through this setup and their corresponding prediction errors are recorded in Table 3.4. This table displays the RMS distance between the predicted and actual positions. This prediction error can only be calculated when the time stamps between the predicted position and GPS reading match. It is assumed that the GPS reading is correct and it calculates the distance vector to the predicted position. Then the mean and standard deviation was calculated of all the calculated RMS error vectors for the whole trajectory and also for the defined segment types.

As expected, the prediction error was less during straight paths, and it increased during curves. Based on the values recorded in Table 3.4, the prediction errors can double during curves. Also, the prediction errors for smooth curves were better than during the sharp curves, which makes sense because, in a smooth curve, the vehicle is changing its heading less abruptly than in a sharp curve, allowing the system more time to recalculate and correct its next prediction.

This research also observed that the prediction error was not the same between all devices, and sometimes a device that has a small prediction error in one segment type may not be as good as that on a different segment type, making it hard to draw conclusions from Table 3.4. In spite of these results, if one looks at the percent deviation of prediction errors compared to the UConn results, it can be narrowed down to the HTC and LG cellphones having the smaller prediction errors overall and meeting the tolerance of no more than 10% more error than obtained with the UConn sensors.

Figure 3.10 is another way of representing the prediction errors for each of the devices in the different segment types. The boxplots display the median value as the solid line dividing the box into two, and then the upper and lower half of the boxes represent the interquartiles, which together represent 50% of the calculated prediction errors. The upper whisker indicates that 75% of the errors fall below it, and the lower whisker indicates the 25%

<table>
<thead>
<tr>
<th>TABLE 3.4</th>
<th>RMS Prediction Error (3 s Ahead)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UConn</td>
</tr>
<tr>
<td>Whole trajectory</td>
<td>1.29 ± 2.64</td>
</tr>
<tr>
<td>Straight paths</td>
<td>0.92 ± 1.65</td>
</tr>
<tr>
<td>Smooth curves</td>
<td>2.28 ± 4.87</td>
</tr>
<tr>
<td>Sharp curves</td>
<td>2.51 ± 4.36</td>
</tr>
</tbody>
</table>

*Note:* Values representing median prediction error in meters ± standard deviation by all devices for trajectory shown in Figure 3.6 (∼25,000 data points).
marker. With this in mind, it can be seen that for the straight paths, except for the iPhone 3GS, the boxes are very short, which means that the prediction errors have a high level of agreement. One can also see small boxes in the smooth and sharp curves for the HTC and LG, so it can be observed that their predictions are fairly consistent most of the time, unlike the boxplot for the iPhone 3GS where it is a very large box indicating a very low level of agreement between the predictions. Also, the lower the boxes to the x-axis, the smaller the prediction errors; so a small box close to the x-axis, like the HTC in sharp curves, or the UConn in straight lines, represents a very accurate prediction system. Again, looking at the boxplots for the five smartphones, one can visually pick the HTC and LG to be fairly good, then maybe Alcatel and iPhone 5, though it looks like iPhone 5 is not as reliable as Alcatel during sharp curves.

When looking at iPhone 3GS results, both in Table 3.4 and Figure 3.10, it can be observed that this device has prediction errors much larger than other devices. It seems this device has a problem, losing its signal quite often, introducing more errors to what was assumed to be the “true” position. Looking more into this topic, we have found several Apple discussion forums (https://discussions.apple.com) where users have reported very inaccurate GPS locations when using iPhone 3GS running iOS 5.
Predicting Vehicle Trajectory

To look at a subset of the whole route shown in Figure 3.6a, a couple of curves were selected as shown in Figure 3.6b, and the results represented in a similar way, but only for this small subset of the dataset. This selected segment of the route represents only 0.8 km (36% straight path, 44% smooth curve, and 20% sharp curve), which takes around 10 s to go through.

When looking at Table 3.5, the first difference that might be observed when comparing it to the results for the whole route shown in Table 3.4 is that the average prediction error for the whole trajectory of the selected subset is different. In this case, straight paths are a small percentage of the whole selected subset while smooth curves are the most abundant. For this very specific set of curves, the UConn data are better than any of the smartphones in all trajectory types. The LG device yields the smallest prediction errors of all the smartphones, and still within the selected 10% tolerance when compared to the UConn results. The next best devices seem to be HTC and iPhone 5 smartphones, where, despite having prediction errors over the 10% tolerance, their prediction errors are around 20% worse than the UConn results.

Another difference one can observe in Table 3.5 is that, unlike the results in Table 3.3, the HTC device did not seem to perform as well in this selected set of curves than when evaluated over the whole route. Even when looking at the results for smooth and sharp curves, the HTC results were always worse than the UConn prediction errors, which is not the case when looking at the data in Table 3.4. In Table 3.5, it seems that most smartphones performed worse than the UConn setup in this set of curves. Since this is consistent across all smartphones, it can be concluded that there was something on the curves that affected the prediction, like bumps or maybe unleveled pavement, especially over the sharp curve section.

A visual representation of the prediction errors during the small subset of curves previously mentioned is shown in Figure 3.11, where each predicted trajectory is compared to the actual GPS position measured 3 s later. One can observe that for some devices there is a smooth trajectory of predicted positions, like for the UConn and LG, closely followed by the HTC, but it can also be observed that some other devices are constantly correcting its predicted position drastically, causing all those spikes during the curves.

### TABLE 3.5

<table>
<thead>
<tr>
<th></th>
<th>UConn</th>
<th>Alcatel</th>
<th>HTC</th>
<th>LG</th>
<th>iPhone 3GS</th>
<th>iPhone 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whole trajectory</td>
<td>1.46 ± 0.97</td>
<td>3.22 ± 3.64</td>
<td>1.81 ± 1.87</td>
<td>1.56 ± 1.65</td>
<td>6.34 ± 6.95</td>
<td>1.79 ± 2.18</td>
</tr>
<tr>
<td>Straight paths</td>
<td>0.69 ± 0.53</td>
<td>2.96 ± 4.60</td>
<td>1.42 ± 2.11</td>
<td>0.77 ± 0.58</td>
<td>4.50 ± 5.76</td>
<td>0.95 ± 0.74</td>
</tr>
<tr>
<td>Smooth curves</td>
<td>1.69 ± 0.54</td>
<td>3.13 ± 2.18</td>
<td>1.75 ± 1.43</td>
<td>1.84 ± 2.13</td>
<td>11.57 ± 6792</td>
<td>1.57 ± 0.85</td>
</tr>
<tr>
<td>Sharp curves</td>
<td>1.98 ± 0.97</td>
<td>3.52 ± 3.81</td>
<td>2.67 ± 2.03</td>
<td>2.19 ± 1.26</td>
<td>14.73 ± 6.24</td>
<td>2.69 ± 3.12</td>
</tr>
</tbody>
</table>

**Note:** Values represent median prediction error in RMS meters ± standard deviation by all devices for the selected curve (∼550 data points.)
One positive thing of looking at the predicted position errors as shown in Figure 3.11 is that, despite the conclusions obtained from Table 3.5 that iPhone 5 was predicting a future position better than several of the other devices, this research would not think this is a reliable device after looking at Figure 3.11. Therefore, even though Figure 3.11 cannot be used by itself to draw some final conclusions, it is a very useful addition to Table 3.5.

An important lesson learned in this research was that the accelerometer sensor could provide misleading information based on road slopes. Table 3.6 shows a data subset where velocity was increasing but tangential acceleration was decreasing, implying that the vehicle was slowing down.

### TABLE 3.6

<table>
<thead>
<tr>
<th>Time Stamp</th>
<th>Altitude (m)</th>
<th>Velocity (m/s)</th>
<th>Tangential Acceleration (m/s²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>33:17</td>
<td>117.93</td>
<td>18.06</td>
<td>−0.508</td>
</tr>
<tr>
<td>33:18</td>
<td>118.03</td>
<td>18.30</td>
<td>−0.593</td>
</tr>
<tr>
<td>33:19</td>
<td>118.12</td>
<td>18.53</td>
<td>−1.061</td>
</tr>
<tr>
<td>33:20</td>
<td>118.27</td>
<td>18.77</td>
<td>−1.168</td>
</tr>
<tr>
<td>33:21</td>
<td>118.80</td>
<td>19.07</td>
<td>−1.858</td>
</tr>
</tbody>
</table>

**Note:** Misleading accelerometer measurements due to altitude changes.
at the data, it can be observed that the vehicle was actually speeding up, and
the decrease on the tangential acceleration was actually because the road was
going up (altitude increase) and not because the vehicle was slowing down.

3.9 Conclusions

The built-in sensors of some smartphones were evaluated to predict the
future trajectory of a vehicle and their prediction errors compared to those
obtained by using VM sensors. Results shown in Tables 3.4 and 3.5 indicate
that smartphones yield similar or better prediction errors and could, there-
fore, be used in older vehicles to participate in a V2V/V2I system. Though
smartphone price seemed to play a small role, the HTC smartphone is one
of the cheaper ones used in this experiment, and it performed quite well
in some scenarios. The more expensive LG device yielded more reliable
results in more, so price could be a factor, but then the iPhone 5, being the
most expensive one, did not contribute well to the price factor. If the results
of the trajectory prediction errors for the whole trajectory driven for this
experiment (shown in Table 3.4) are observed, it can be concluded that all
smartphones, except for iPhone 3GS, yield prediction errors up to 10% more
than the UConn errors. Actually, it can be observed that the results from
the smartphones perform, in most scenarios described in Table 3.4, even
better than the baseline set by the VM sensors (UConn), some even almost
50% better. With these results, it can be safely concluded that the trajectory
prediction accuracy of the smartphones’ sensors is sufficiently reliable to be
considered for a temporary hook to enable an older vehicle to participate in
a V2V/V2I system.

Future research will include the use of other sensors found in some of
the smartphones used (gyroscope and magnetic compass) to evaluate if they
improve prediction of a vehicle’s trajectory and how these predicted trajecto-
ries can fit into collision avoidance systems.

References

1. V. Di Lecce, A. Amato, Route planning and user interface for an advanced

2. X. Xu, T. Xia, A. Venkatachalam, D. Huston, The development of a high
speed ultrawideband ground penetrating radar for rebar detection, *Journal of Engineering


Predicting Vehicle Trajectory


Trajectory Estimations Using Smartphones


