MARVEL: Multiple Antenna based Relative Vehicle Localizer

Dong Li[†], Tarun Bansal[†], Zhixue Lu[†] and Prasun Sinha
{lido, bansal, luz, prasun}@cse.ohio-state.edu
Deptt. of Computer Science, Ohio State University
Columbus, Ohio
[†]Co-primary Authors

Abstract

Access to relative location of nearby vehicles on the local roads or on the freeways is useful for providing critical alerts to the drivers, thereby enhancing their driving experience as well as reducing the chances of accidents. The problem of determining the relative location of two vehicles can be broken into two smaller subproblems: (i) Relative lane localization, where a vehicle determines if the other vehicle is in left lane, same lane or right lane with respect to it, and (ii) Relative front-rear localization where it needs to be determined which of the two vehicles is ahead of the other on the road. In this paper, we propose a novel antenna diversity based solution, MARVEL, that solves the two problems of determining the relative location of two vehicles. MARVEL has two components: (i) a smartphone; and (ii) four wireless radios. Unlike exisiting technologies, MARVEL can also determine relative location of vehicles that are not in the immediate neighborhood, thereby providing the driver with more time to react. Further, due to minimal hardware requirements, the deployment cost of MARVEL is low and it can be easily installed on newer as well as existing vehicles. Using results from our real driving tests, we show that MARVEL is able to determine the relative lane location of two vehicles with 96.8% accuracy. Through trace-driven simulations, we also show that by aggregating information across different vehicles, MARVEL is able to increase the localization accuracy to 98%.

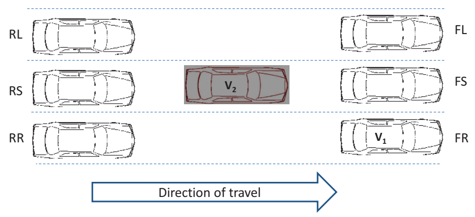


Figure 1: Vehicle V_1 can be in six different relative positions with respect to V_2: Front-Left, Front-Same, Front-Right, Rear-Left, Rear-Same and Rear-Right. Vehicle V_1 is said to be in *front* of V_2 if *front* of V_1 is ahead of that of V_2 along the direction of travel.

1 Introduction

The use of GPS equipped smartphones has been increasing rapidly. But the GPS devices on the phones do not have sufficient accuracy to localize the vehicles up to the lane-level. Availability of traffic information at the micro-level or lane-level granularity is useful for multiple applications that not only reduces the chances of accidents but also enhances the driving experience. Some applications are as follows: (i) Alerting the drivers of upcoming obstacles or potholes that are in the same lane as the vehicle, and further guiding the driver to move to the appropriate lane to avoid them; (ii) Alerting the drivers if there is a vehicle in the blind zone or if this vehicle is tailgating another vehicle, thereby reducing the chances of collision; (iii) Alerting the driver that the vehicle ahead is slowing down if the two vehicles are in the same lane; (iv) Detecting the lane-level location of slow moving vulnerable vehicles; and, (v) Determining the differences in speeds of different lanes, to assist in traffic planning.

The problem of *Vehicular Localization* has two variants: (i) *Relative Localization*, where for two vehicles, the objective is to determine the location of a vehicle with respect to the other vehicle; and, (ii) *Absolute Localization*, where for a single vehicle, the objective is to determine its absolute location. In this paper we develop a novel solution called MARVEL (Multiple Antenna based Relative Vehicle Localizer) that provides *relative localization* of on-road vehicles. Specifically, for two given vehicles, V_1 and V_2, MARVEL uses antenna diversity to enable both of them to determine the relative position of the other vehicle among the six different possible regions.

Although, GPS technology is widely used for vehicular localization, various factors such as signal multipath, unknown delays due to ionosphere and tropo-

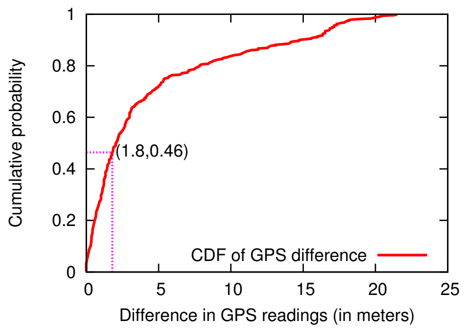


Figure 2: Comparison of GPS trace of two smartphones located in the same car: Cumulative distribution function of GPS error (in meters).

sphere, error in the clocks of GPS devices, and inaccuracies in the locations of satellites [12] reduce its accuracy. Device manufacturers such as Garmin report the average GPS accuracy to be 3 meters [12] even for devices equipped with newer WAAS (Wide Area Augmentation System) and DGPS (Differential GPS) technology. The GPS trace collected by us for two phones (Motorola Atrix and iPhone 4) mounted next to each other (horizontal distance $\leq$ 1cm) on the windshield of the same car showed similar inaccuracies. Figure 2 shows the cumulative probability distribution of distance between the locations of the vehicle along the width of the road as reported by the two smartphones. The width of the lanes as recommended by AASHTO [3] on local and arterial roads is 3.60 meters. So, if the error is more than 1.80 meters across the direction of travel, GPS will miscompute the *relative lane location* (left, right or same lane) of the other vehicle, which happens in 54% of the cases. Similarly, GPS may also incorrectly compute the *relative front vs. rear location* of the other vehicle if the two vehicles are close to each other. In urban canyons, due to multipath effects, the accuracy of GPS is expected to decrease even further. *In this paper, we propose an antenna diversity based novel solution for both relative lane localization and relative front-rear localization*.

Various vehicle manufacturers are beginning to roll out new car models with vehicular technologies such as doppler radar and cameras for relative localization. These technologies work even when the hardware is deployed on only one vehicle (and not the other vehicles in the vicinity), but they have several limitations: (i) Both the doppler radar and camera can accurately detect only those vehicles that are in the immediate neighborhood; (ii) Nearby obstacles such as parked vehicles, store fronts and crash barriers reduce the accuracy of the radars [13] as well as cameras [13]; (iii) These technologies cannot be easily retrofitted to existing vehicles, resulting in lower market penetration; and, (iv) There is generally a tradeoff between accuracy and number of radars used [18]. More radars can cover more regions while providing high localization accuracy, however this increases the deployment cost.

MARVEL is minimal in design and consists of two components: (i) A smartphone that could be the same as one carried by the driver; and, (ii) Four off-the-shelf wireless radios mounted on the sides of the vehicle. The smartphone runs our localization algorithm and displays relevant alerts to the driver. The wireless radios on different vehicles send *lane discovery* beacons to each other and report the RSSI (Received Signal Strength Indicator) of the received beacons to the smartphone. MARVEL determines the relative lane location of vehicles by leveraging the differences in RSSI measurements due to spatial antenna diversity. MARVEL is based on the observation that path loss of the links between radios is asymmetrical when the vehicles are in different lanes, and it is symmetrical when the vehicles are in the same lane. MARVEL satisifes the following desired requirements:

- Useful information (such as presence of potholes, applying hard brakes or presence of slow moving vulnerable traffic) from vehicles that are in the same lane as this vehicle, can provide the driver with more time to react. Our system is able to detect relative location of even those vehicles that are not in the immediate neighborhood (thus invisible to camera and radar), thereby providing the driver with more information to reduce the incidence of collisions. This information can also be used for improving the design of autonomous cruise control mechanisms.

- It works in a wide variety of environmental conditions such as bad weather (rain, snow etc.), various light conditions (dark, sun glare etc.), as well as in presence of urban noise (e.g. parked cars and store fronts on roadside, crash barriers, sound walls etc.)

- It is low cost due to ubiquity of smartphones and the low cost of wireless radios.

- It can be easily deployed on older as well as newer vehicles.

- Our driving results performed using different vehicles and under varying speed and traffic conditions show that MARVEL computes the relative location with an average accuracy of 96%. By aggregating information across multiple vehicles, MARVEL is able to increase the localization accuracy to 98%.

This paper is organized as follows: In Section 2, we discuss related work in this area. Section 3 describes MARVEL in detail. In the next two sections, we describe the results from our real driving experiments. In Section 6, we describe two algorithms that further improve the localization accuracy by aggregating information across different vehicles. In Section 7, we describe the results from our ns-3 and SUMO based simulations. In Section 8, we discuss different methods that can make MARVEL more realistic and useful. Finally, in the last section we conclude the paper.

2 Related Work

The topics of vehicle detection and lane recognition have been studied in the literature of autonomous vehicles for decades. Radar [2,20], laser [22], and acoustic [7] based sensors and cameras are common devices in such studies. Radar, laser and acoustic sensors detect the distance of objects by measuring the round trip time of signals. Those sensors are commonly used for vehicle detection and blind spot detection. For instance, [2] embeds 24-Ghz radar sensors into the back side bumper to monitor the blind spots of the host vehicle, and triggers a vision warning signal if some vehicles are detected. However, those sensors are usually limited in range to line of sight, difficult to install especially on existing vehicles and exhibit a tradeoff between accuracy and cost. Cameras have broader application in vehicle detection [14] and lane recognition [8, 23] due to their low costs. The existing solutions have used camera for recognizing vehicles, obstacles and lane marks through image recognition techniques. However, using camera for vehicle detection and lane recognition is highly susceptible to errors due to various factors such as: (i) Bad light conditions (e.g., night time, sun glare, headlight glare, shadows from nearby buildings); (ii) Improper weather conditions (e.g. snow, rain); and, (iii) Surrounding noise (e.g., faded lane marks, vehicles parked on roadside, roadside crash barriers, trees, store fronts etc.).

On the other hand, the computational power of smartphones is being utilized in various applications such as assisted driving and road infrastructure monitoring [17]. For example, [24] has recently proposed ways to determine if the user of the smartphone is the driver or a passenger in the vehicle which could be used to adjust the behavior of the phone based on the owner's type. The idea of using wireless sensors to detect and track vehicles has been studied in [19, 21]. These works mainly focus on detecting and tracking the movements of vehicles in a wireless sensor network deployed in a given area. In contrast, the objective of this paper is to *determine relative vehicle locations* using *multiple wireless radios installed on vehicles*.

3 System Design

In this section, we will first describe the objectives of our solution. Then, we give an overview of our system design followed by a list of challenges. Finally, we describe MARVEL in detail.

3.1 Problem Statement

The objective of our system is to determine the relative position of two vehicles. Specifically, given two vehicles (see Figure 1) V_1 and V_2, we seek to determine: (i) *Relative lane localization:* If V_1 is in left lane, same lane or in right lane with respect to V_2's lane; and, (ii) *Relative front-rear localization:* If V_1 is in front or rear with respect to V_2. We call the combined problem of "Relative Lane Localization" and "Relative front-rear localization" as *Relative Vehicle Localization*. Vehicle V_1 is said to be in *front* of V_2 if the *front* of V_1 is ahead of that of V_2 along the direction of travel. This definition also takes into account the case when one vehicle is in the blindzone of the other.

3.2 Overview and Challenges

MARVEL comprises of two components for every vehicle. The first component is a smartphone that could also be the personal smartphone of the vehicle's driver. The second component comprises of four wireless radios located at various positions on the lateral sides of the vehicle. We assume that each vehicle has a unique *vehicle id* which could possibly be derived from its license plate number. The ID of the vehicle (V_i) is known to the smartphone inside the vehicle as well as the four wireless radios located on the vehicle.

We assume that the smartphone is equipped with an accelerometer and is Wifi capable. The smartphone (or the MARVEL application running on it) is used to communicate with other smartphones in other vehicles as well as to display alert messages. A vehicle may have multiple smartphones present inside, however, we assume that in each vehicle, only one smartphone is running the relative localization algorithm. So, we assume that the unique *phone ID* of the smartphone in vehicle V_i is P_i. If the smartphone has GPS capability, then the velocity of the vehicle can be obtained from the GPS. This information along with the relative lane information can be used to develop various other applications (see Section 1).

Our algorithm running on the smartphone keeps track of relative locations of all nearby vehicles with whom the smartphone can communicate. For each vehicle, it tracks: (i) The unique vehicle ID of the other vehicle; and (ii) Relative location of the other vehicle, among the 6 possible regions (see Figure 1). It is also possible to extend this information to 2-hop neighbors by using multihop communication. This would provide extra information to the driver and the smartphone at the cost of higher message overhead. However, in this paper, we assume that the smartphone maintains information of vehicles that are only within one hop.

The second component of our system, comprises of four wireless radios located on the lateral sides of the vehicle. In Section 5.2, we will show that placing the radios close to each wheel of the vehicle maximizes the accuracy of localization. The four wireless radios communicate with radios located on other vehicles as well as with the smartphone located in the same vehicle. The radios may use Wifi or Zigbee to communicate with each other, while to communicate with the phone, they can use Wifi, Bluetooth or Zigbee depending on the capability of the radios and the smartphone. We also assume that the wireless radios are aware of the position where they are mounted on the vehicle.

In our vehicle localization algorithm, one of the two phones directs its four wireless radios to send beacons, while the radios on the other vehicle listen for the beacons, thereby estimating the path loss between the 16 pairs of wireless radios. Figure 3 shows a simpler case where only two wireless radios are mounted on the vehicle, one on each side. Our vehicle localization algorithm is based on the observation that when the two vehicles are in same lane, the two links AD and BC are symmetrical (see Figure 3). However, when the vehicles are in different lanes, the links AD and BC are assymmetrical (discussed in Section 3.3.3 in more detail). However, desigining a vehicle localization system based on RSSI readings of the links involves multiple challenges: (i) The best location to deploy the wireless radios that maximize the localization accuracy needs to be determined; (ii) If a vehicle changes its lane, the relative lane location of other vehicles should be updated with minimum latency while limiting the number of packets transmitted by radios so as to minimize congestion and energy consumption; (iii) Packets transmitted may be lost due to collision with other transmissions; and, (iv) Multi-

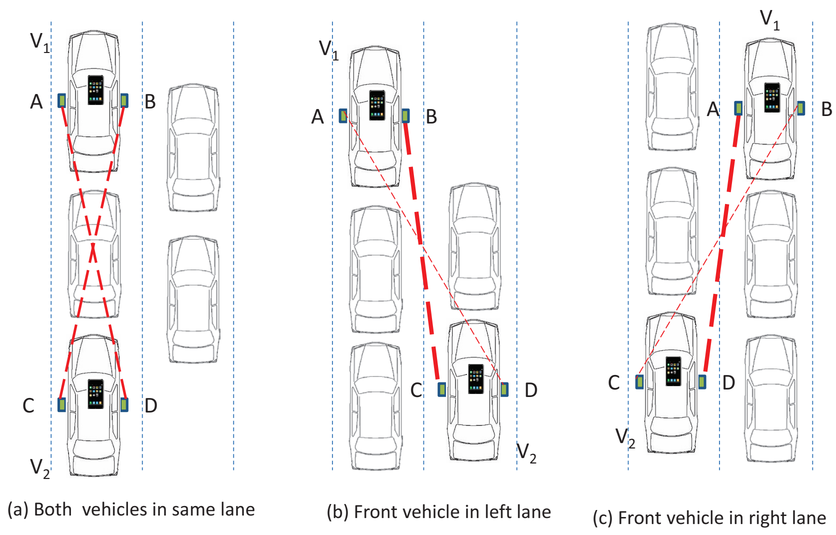

V1
A
B
C
D
V2
(a) Both vehicles in same lane
V1
A
B
C
D
V2
(b) Front vehicle in left lane
V1
A
B
C
D
V2
(c) Front vehicle in right lane

Figure 3: RSSI based Relative Lane Localization with two radios: A, B, C, and D are radios mounted on vehicles V_1 and V_2. The expected RSSI for two links AD and BC are shown for three different cases with thicker lines respresenting links with higher RSSI values. (a) When V_1 and V_2 are in the same lane, then the path losses for links AD and BC are almost symmetrical. (b) When V_1 is to the left of V_2, then BC is a direct line of sight link while AD passes through bodies of 2 vehicles (total of 4 walls and 1 vehicle's heavy machinery compartment). (c) When V_1 is right of V_2, link AD is stronger but BC is weaker. Nearby vehicles may affect the signal strengths slightly due to multipath.

path propagation may affect the accuracy of relative localization especially if the sender and the receiver are separated by a large distance.

3.3 Lane Localization Algorithm

In our vehicle localization algorithm, the smartphone transitions between three different phases (see Figure 4). We will now describe the three phases of the smartphone and the localization algorithm in more detail.

3.3.1 Monitor Phase

One naive way to trigger the lane localization algorithm is to invoke it at periodic intervals. However, this would involve considerable packet transmissions from

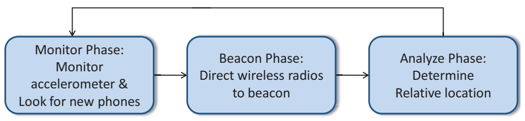

Monitor Phase: Monitor accelerometer & Look for new phones
Beacon Phase: Direct wireless radios to beacon
Analyze Phase: Determine Relative location

Figure 4: Transitions made by the smartphone among the three phases.

the wireless radios themselves, thereby affecting their battery lifetime. Further, it reduces the probability that a beacon transmitted is received correctly by a receiving radio. Since, lane changes or turns are not very frequent events, therefore, in the first phase, the smartphone itself *monitors conditions* that indicate the following: (i) A possible change in relative location of the vehicle with respect to other vehicles; and, (ii) Arrival of a new vehicle in the vicinity. Under both conditions, the smartphone will inform its neighbors of a possible lane change event and move to the beacon phase. A change in relative location of a vehicle with respect to another vehicle may happen due to two factors: (i) If the vehicle itself changes lanes or turns to a new street; or, (ii) It overtakes another vehicle. We will now discuss how the smartphone detects all such possible events.

Detecting Possible Lane Change: To take into account the first condition, the smartphone (i.e., our application running on the smartphone) continuously monitors the readings from the accelerometer of the phone to detect possible lane changes or turns. If the smartphone observes that the accelerometer reading is beyond a certain threshold, the phone moves to the second phase. However, as we will show in Section 4, the accelerometer readings are not completely reliable due to factors such as potholes, bumps and slopes in the road surfaces and curvature of roads. Therefore, as discussed later in Section 4, MARVEL computes the difference between *maximum* and *minimum* accelerometer readings over a window of duration 4 seconds and uses that to detect a lane change event.

Detecting Possible Overtaking: Due to differences in speeds of different vehicles, it is possible that a vehicle V_1 that was initially in one of the three rear regions of V_2, later overtakes V_2, thereby changing its relative position to one of the three front regions of V_2. In such a case, the relative localization algorithm needs to be triggered by either V_1 or V_2 to update the relative locations. One way to do this would be to trigger the relative localization algorithm at periodic intervals. However, this would substantially reduce the lifetime of the radios. Another method that could be explored is where the phones themselves send beacons to each other at regular intervals and also monitor the RSSI of the corresponding received beacons. A change in the RSSI reading of the beacon indicates changes in the distance between the two vehicles. A pattern in the received RSSI where it changes from low to high and then back to low indicates that the other vehicle is getting closer and then again getting farther away. If the phone observes such a pattern, it moves to Phase 2 and triggers the localization algorithm.

Discovering New Vehicles in Vicinity: To establish connections with new smartphones that enter its vicinity and to determine the relative location of those vehicles, the smartphone broadcasts periodic *discovery beacons* using UDP. The phone also listens for discovery beacons broadcast by other phones. If the phone hears a discovery beacon from a previously unknown phone, the algorithm moves to Phase 2. Reducing this discovery time enables phones to determine the relative lane location of neighboring vehicles faster. In order to reduce this discovery time while managing the smartphone's battery consumption, MARVEL can work in conjunction with any of the neighbor discovery algorithms [10, 15] that are already available in the literature.

3.3.2 Beacon Phase

In the beacon phase, the wireless radios on the vehicle transmit or receive beacons. Let V_1 be the vehicle (with phone P_1) computing its relative position and V_2 (with phone P_2) be any vehicle among its set of neighbors. Then P_1 will instruct its four wireless radios to send *localization beacons* while $P2$ instructs its radios to listen to these beacons. Each wireless radio hears multiple localization beacons, and reports the RSSI of the received beacons to its associated phone, P_2.

The lane localization beacon broadcast by radios on V_1 contains three fields: (i) V_1; (ii) Location of the radio (among the four possible locations) on V_1; and (iii) Transmission power level of the beacon. The packets transmitted by radios on V_2 to the smartphone P_2 contains five fields: (i) V_1; (ii) V_2; (iii) Location of the radio sending this packet on V_2; (iv) Transmission power levels of the four lane localization beacons received; and, (iv) RSSI values of the four beacons received. By including the vehicle IDs in the messages, MARVEL is able to ensure that the radios and the smartphone only process messages that are either directed at them or the broadcast messages. Further, the radio on V_2 may not be able to receive beacons from all the four radios of V_1. In that case, the corresponding values in the packet are left empty.

In our experiments, we observed that due to collisions, packet losses may happen. In order to make the algorithm more robust, we require each radio to send multiple lane localization beacons that are randomly separated. The receiving radio computes an average RSSI value for each link, while ignoring outliers, before reporting it to the phone. This provides a good tradeoff between accuracy and overhead.

3.3.3 Analyze Phase

In the third phase, smartphone P_2 proceeds to compute the relative location on the basis of path loss of 16 links. P_2 can compute the path loss for the 16 links (four values from each radio) based on the information received from the wireless radios. Figure 3 shows how using two radios it is possible to distinguish the three possible cases of two cars in the same lane or in different lanes. Observe that when the two cars are in the same lane, then links AD and BC are roughly symmetrical. Thus, the path loss values of these two links would be similar. However, when the cars are in different lanes, the links are not symmetrical. Specifically, when V_1 is in the left lane of V_2, link BC has low path loss compared to path loss on link AD since BC is a direct line of sight path with no obstacles while AD passes through bodies of 2 vehicles (or 4 walls and the engine compartment of one of the vehicles).

So it is possible that the relative signal strength of these two links can be used to distinguish the three scenarios, thereby solving the relative lane localization problem. In the same setup, adding two more radios to each vehicle, similarly provides us with more information that can be utilized to solve relative front-rear localization problem. Later in Section 5, we show how the 16 averaged RSSI values are used as input to a Support Vector Machine (SVM) [6] based discriminator to identify the six relative locations (see Figure 1). After P_2 computes the relative location of V_1 with respect to V_2, it informs P_1 to update accordingly.

The next two sections describe the results from our real driving experiments where we evaluate how MARVEL performs in presence of multiple vehicles. In Section 7, we describe results from our ns-3 and SUMO based simulations, where we evaluate the performance of MARVEL with multiple vehicles under multihop communication scenarios.

4 Lane Change And Turn Detection

To reduce power consumption due to periodic beacon transmissions (see Section 3.3.1), we propose putting the radios into a sleep mode, and engaging them (for transmitting or receiving of beacons) by the phone only when either a lane-change or a turn event is detected, or when a neighbor requests for localization. This also reduces the number of packets transmitted, thereby increasing the chances that the transmitted beacons are received correctly by the receiving radios. Lane-change events can be detected by accelerometers within the phone. Toward this, we design a threshold based algorithm for accelerometers to detect the lane-change events. Using data collected from our experiments, we show that it is possible to detect lane-change events with high accuracy.

In this experiment, we used a Motorola Atrix smartphone with Android 2.3 to collect the accelerometer data. The phone has a built-in 3-Axis accelerometer, and was placed on the dashboard of the car. To simplify the procedure of data analysis, the phone was oriented in a way so that only readings along the X-axis will be affected within the duration of lane-change. If the phone is placed in an arbitrary orientation, then virtual reorientation [17] can be used to compute the acceleration along the width of the vehicle. On average, the accelerometer in the phone gives 100 readings every second, which are susceptible to noise. To cancel out the noise and inaccuracy of the accelerometer, we compute the reading at any point by taking an average of readings received in the previous 0.5 second. Figure 5 shows an example raw data set and the corresponding smoothed data.

Our algorithm is based on the observation that when a lane-change event happens, the force computed by the accelerometer along the width of the vehicle has a specific pattern. Assuming that the vehicle is changing its lane towards the direction of the accelerometer's positive X-axis, then the accelerometer reading first increases to a high value and then decreases back to a lower value.

To capture this pattern, our algorithm works as follows. The phone maintains the *maximum* and *minimum* readings of the accelerometer (along the width of the vehicle) within a window of last 4 seconds. This duration of the window is half of the maximum duration taken by drivers to perform a lane change[1]. Whenever a new reading is available, it updates both the *maximum* and *minimum* values, and checks their difference. If the difference is larger than a specified threshold (τ), the phone reports a lane-change event, communicates with neighboring phones asking them to engage their radios, and then moves to the Beacon Phase and triggers the lane localization algorithm.

As shown next, we found $1.08m/s^2$ to be an appropriate threshold to detect lane-changes and turns on both local roads and highways. Based on a total of 148 lane-change events performed in our experiments, we found (see Figure 6) that a tradeoff exists between the *recall* (= $\frac{TruePositive}{GroundTruth}$) and *precision* (= $\frac{TruePositive}{TotalPredictions}$)

[1]In our experiments, we observed that with $> 98\%$ probability, all the lane changes are completed within 8 seconds.

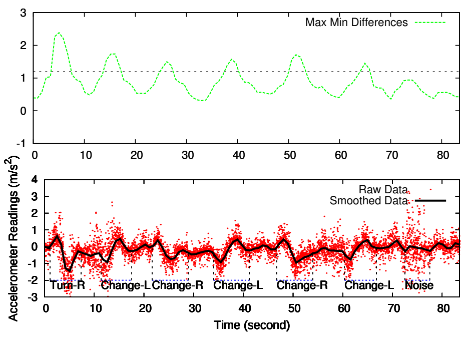


Figure 5: Bottom graph shows the raw and smoothed accelerometer readings along the X-axis. Left and right lane Change events are shown by "Change-L" and "Change-R", respectively while left and right turn events bu "Turn-L" and 'Turn-R", respectively. Top graph shows the variation in the difference of values of *maximum* and *minimum* smoothed accelerometer readings within the window of last 4 seconds.

of the prediction. To make sure that the detection algorithm does not miss any lane-change events, a higher recall rate is desired. From Figure 6, it is clear that with a threshold $1.08m/s^2$, 100% recall is possible with roughly 80% precision for local and 60% precision for freeway. This ensures that whenever a lane-change event happens, the smartphone is able to detect it. At the same time, it may also generate some false positive events (roughly $1-2$ for every 5 predictions). However, since a more accurate relative localization is performed in Phase 2 and Phase 3, such false positives would not affect the accuracy of our solution. In practice, we do not expect too frequent lane changes, thus we believe that the precision is acceptable.

5 Real Driving Experiments

In this section, we will describe the results from our real driving experiments performed with different vehicles under varying traffic and road types. The purpose of the experiments was: (i) To determine the best mounting position of wireless radios; (ii) To evaluate the correctness and generalizability of MARVEL under different road types as well as traffic conditions; and, (iii) Determine if MARVEL's accuracy depends on the type of vehicle used. The conclusions from the experiments are described in Section 5.8.

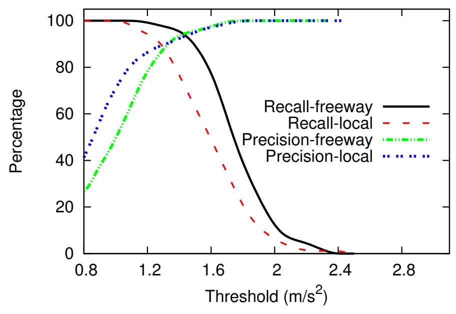


Figure 6: The tradeoff between recall and precision of predictions with variation in detection threshold τ on freeway and local drives.

5.1 Experiment Setup and Data Processing

For our driving experiments, we used four different vehicles: (i) Car 1: 2006 Toyota Solara Coupe; (ii) Car 2: 2003 Mazda M6 Sedan; (iii) Car 3: 1999 Nissan Sentra Sedan; and, (iv) SUV 1: 2011 Hyundai Santa Fe. To evaluate the accuracy of MARVEL on roads with different speed limits, we also chose two different kinds of roads: (i) A 5 lane local (speed limit 35 mph) roadway in an urban environment with parked vehicles and storefronts on both the sides; and (ii) Freeway (speed limit 65 mph), with sound walls and crash barriers on the sides of the road at few places. Traffic was observed to be *moderate* in all the experiments, except Section 5.4 where we evaluated the generalizability of our solution with varying traffic conditions. Since, the experiments were performed under uncontrolled settings, sometimes other vehicles may intervene between our two test vehicles.

We used the commercially available TelosB motes as the wireless radios mounted on the sides of the vehicles. The body of the mote (along with batteries) was observed to be around 1.5cm thick with a 3cm antenna. Since these dimensions are very small compared to the width of the vehicle, we do not anticipate any significant reduction in efficiency of the car due to the additional wind drag. The range of the TelosB to TelosB communication was observed to be 60 meters (when there are no obstacles in between) in parking lots. However, we could not measure the range during driving due to unavailability of appropriate equipment. For the purpose of experiments, instead of the smartphone, we used another TelosB mote connected with a laptop, inside the vehicles, to collect RSSI readings from the TelosB motes mounted outside. To handle possible packet losses due to collisions, TelosB motes send 5 localization beacons at a random spacing of 25-50ms.

The path loss value for a link is then computed by taking an average of the 5 values.

We identify the problem of finding another vehicle's position, among the *six* possible positions (see Figure 1), as a supervised classification problem and address it using machine learning algorithms. For this, we used the RapidMiner software (version 5.1.104) [16] for training and testing. In RapidMiner, we chose the SVM's implementation libSVM with the SVM type set as C-SVC and kernel type as linear. The values of C, cache size and epsilon were set to 0.0, 80 and 0.001, respectively. We define the *Classification accuracy* as the percentage of the testing data for which the relative location is correctly predicted by the classification model. For different experiments, we use different training and testing techniques to evaluate the accuracy of MARVEL. In some cases, we divide the collected data set into two parts, train on the first and test on the second. However, in some other cases, we use independent test sets, i.e., we train the system on one kind of data (e.g., for local roads) and test it on another kind of data (e.g., freeways).

Errors in classification accuracy may arise due to a variety of factors: (i) Presence of nearby driving vehicles on the roadway and urban obstacles (store fronts, crash barriers, parked cars, sound walls etc.) may affect the value of the signal strength; (ii) Transmissions from the roadside Wifi access points may affect the packet drop rate of the transmitted beacons; (iii) Sometimes a car may straddle two lanes; and, (iv) Different vehicles may have different physical profile (width, organization of machinery under the car hood etc.) which may affect the path loss values.

The results shown in this paper are based on our 500 miles of driving. During the driving experiments, one of the passengers recorded the ground truth by manually recording the time and the relative vehicle positions while the drivers changed lanes at regular intervals. We randomly split the finally collected data set into *testing data set* and *training data set*. It was ensured that each of the 6 classes (corresponding to six different relative positions from Figure 1) have equal amount of data in both the training sets and the testing sets.

5.2 Determining the best configuration

In this experiment, we varied the number and positions of radios on the vehicle to determine the configuration that maximizes the localization accuracy. In our experiment, we explored 21 candidate configurations on the exterior of a car to mount the TelosB motes. Figure 7 shows four such possible configurations along with their labels and accuracies. In each driving test, we mounted the specified number of radios on the two cars. The total duration of driving for each test was around 60 minutes with about 30 minutes of local driving and 30 minutes of freeway driving. In this subsection, we used four types of driving data for each radio position combination: (i) Data between two sedans on local roads; (ii) Data between two sedans on freeway; (iii) Data between the coupe and SUV on local roads; and, (iv) Data between the coupe and SUV on freeway. The data was then divided into equal-sized training set and a testing set.

We observed that when four radios are placed vertically above the four wheels on the vehicle's body, we get the maximum accuracy of 99.78%. In the later evaluations, we only use configuration A, unless mentioned otherwise. Also, from Figure 7, we can observe that using more radios increases the classification accuracy since more radios capture more information about the relative locations of vehicles. The results of the experiment also indicate that the position of the radios affect the accuracy of the classifier. For example, C and H both have three radios but their accuracy is significantly different.

5.3 Evaluation with varying road types

The purpose of this experiment was to evaluate the generalizability of MARVEL with varying road types: (i) Local roads; and, (ii) Freeways. The speed limit on local roads and freeway were 35 MPH and 65 MPH, respectively. The traffic was observed to be *moderate* when collecting the two data sets. We used the coupe and the SUV's driving data as well as the two sedan's driving data in this experiment. After the experiment, we created a local driving data set $Local$ and a freeway driving data set $Freeway$.

We first used data from $Local$ to train an SVM classifier. When tested on data from $Freeway$, the classifier's accuracy was found to be 97.33%. In the next test, we trained another SVM classifier using $Freeway$ data and tested it on $Local$ data. This time, the classifier gave an accuracy of 99.39%. This high classification accuracy indicates that the path loss patterns on local roads and freeways are similar and that training is independent of road type and speed.

5.4 Evaluation under varying traffic conditions

In this experiment, we want to determine whether MARVEL is robust to variations in traffic conditions, and then decide whether we need to include the driving data of different traffic conditions to train the classifier. For this, we created a heavy

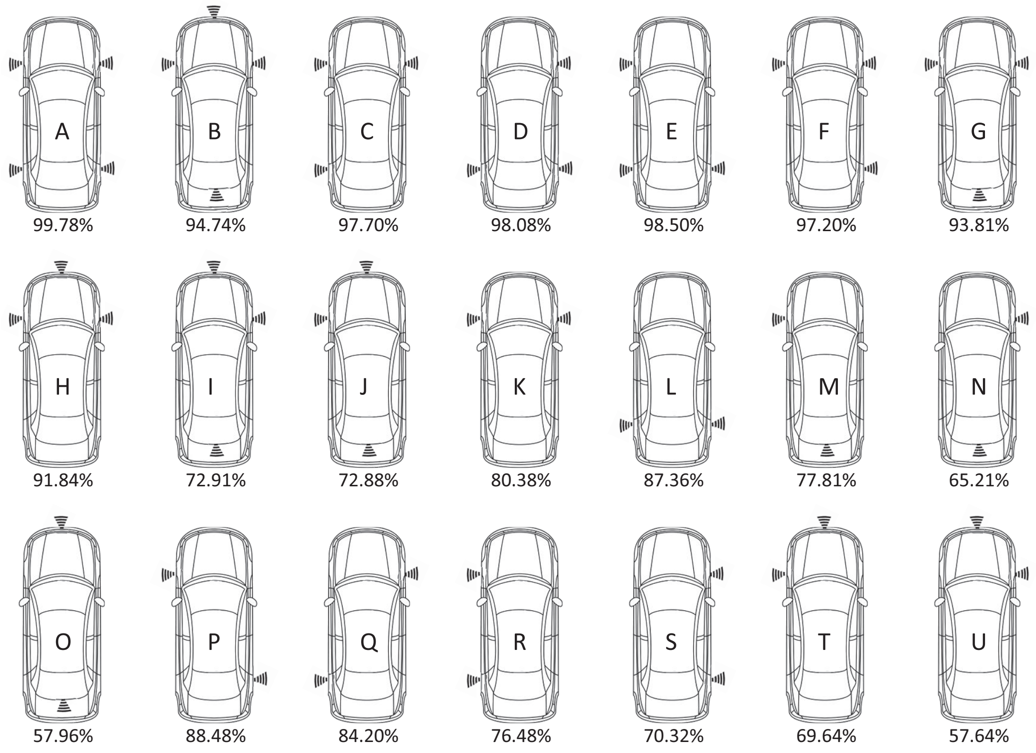


Figure 7: The different combinations of placement of the wireless radios and the corresponding relative lane localization accuracy. The classifier from configuration A has the highest prediction accuracy.

traffic data set called $Heavy$ and a light traffic data set called $Light$. $Heavy$ contains the data between the coupe and SUV on a mix of busy local roads and busy freeways. The same two vehicles were also driven under light traffic conditions to create the dataset $Light$.

We used $Heavy$ to train an SVM classifier, and then on testing it with $Light$, the classification accuracy was found to be 38.68%. In the next test, we used $Light$ as the training set. When tested by $Heavy$, the new classifier had accuracy of 25.22%. In both cases, the accuracy is quite low. This indicates that the radio's path loss pattern in light and heavy traffic conditions are different. This is because in heavy traffic, frequently there are multiple other vehicles between the two testing vehicles. This result is consistent with our observation that the body of a car can affect the path loss of the wireless radio signal.

However, we find that splitting $Heavy$ into two equal sized sets $Heavy1$ and $Heavy2$, training the classifier on $Heavy1$ and testing it on $Heavy2$, increases the classification accuracy to 96.91%. Similarly, if $Light$ is split into two sets $Light1$ and $Light2$, with training performed on one and testing on the other, it also increases the classification accuracy to 99.84%. This result indicates that both heavy traffic data and light traffic data have clear and consistent patterns. Therefore, to estimate the relative position in different traffic conditions, one way is to include the traffic condition as an input to the classifier.

We created a mixed traffic data set Mix by combining equal amounts of data from both $Heavy$ and $Light$. Then we split Mix into two equal sized data sets $Mix1$ and $Mix2$, trained the SVM classifier on $Mix1$ and tested it separately on $Mix2$, $Light2$ and $Heavy2$. This classifier's accuracy was observed to be 97.21%, 95.73% and 99.11%, respectively. The mixed traffic condition classifier has high accuracy indicating that for high accuracy classification in various traffic conditions, it is useful to train it under both light and heavy traffic conditions. This removes the requirement to provide the traffic condition information as an input.

5.5 Evaluation with variation in vehicle types

In this experiment, we tested how the body of a car affects the performance of the classifier. We call the data collected by using the coupe and the SUV on both local roads and freeways as $SUVData$, and the data collected using two sedans on same roads as $SedanData$. For both types of roads, the traffic was observed to be moderate.

We first use $SedanData$ to train an SVM classifier, and upon testing it with $SUVData$, we observed an accuracy of 88.32%. Similarly, in the next test, we trained the SVM classifier with $SUVData$ and tested it using $SedanData$ and observed the accuracy to be 93.50%. Compared with the results in Section 5.2, the results here are around 10% lower. This indicates that different types of vehicle bodies create different signal strength patterns; however, the variation in the patterns is much smaller than the variation due to different traffic conditions. We can see from the result in Section 5.2 that when we include different types of vehicles into the training set, the accuracy was very high (99.78%). This indicates that the classification model does not have a bias problem with respect to vehicle type, therefore, it is not required to provide vehicle body as an input to the model.

5.6 Evaluation with variation in wireless radio position

As discussed in Section 5.2, the best position for mounting the wireless radios is above the wheels. However, sometimes due to human-error, the radios may not be mounted at exactly the right place. In this experiment, we studied the variation in accuracy of MARVEL with random errors in mounting the wireless radios. This time, for collecting data, we drove two sedans on local as well as freeway and collected three data sets: (i) *Correct*: Radios mounted at correct position with correct orientation; (ii) *Incorrect10*: Radios mounted anywhere within 10 cm of the correct position with random orientation; and, (iii) *Incorrect20*: Radios mounted anywhere within 20 cm of the correct position with random orientation. We then divided *Correct* in two equal parts, *Correct1* and *Correct2*. By training the SVM classifier on *Correct1*, and testing on *Correct2*, *Incorrect10* and *Incorrect20* we obtained accuracies of 99.2%, 98.0% and 96.5%, respectively. Next, we mixed all the three data sets to get Mix and used it for training. When tested on *Correct*, *Incorrect10* and *Incorrect20*, the SVM classifier gave an accuracy of and 99.1%, 98.4% and 97.3%, respectively.

5.7 Mix all types of driving data together

In this section, we trained the classifier on a data set combined from all driving tests and evaluated its accuracy on different types of driving data. In the combined training set, we include: (i) the two sedan's local and freeway driving data in moderate traffic; and, (ii) the coupe and SUV's local and freeway driving data under both moderate and heavy traffic conditions. The testing was done on different sets of driving data separately as well as on another data set from a drive on a *curved freeway*. Due to unavailability of a local long enough road with multiple lanes and curves, we skipped testing the classifier on *curved local roads*. From Table 1, we can see that for all the cases, the mixed mode's accuracy is at least 92.8%. When testing on curved freeways, the accuracy is slightly lower, as the classifier predicts some of the same lane cases as left lane or right lane. The overall accuracy was observed to be 96.8%. This result confirms that MARVEL is robust to most of the variations in driving cases.

5.8 Conclusion from the experiments

From the experiment results (Section 5.2), we can make the following conclusions: (i) The best configuration among all the configurations we tested is A (Fig-

Table 1: Test results of the classifier trained by mixing all the driving data.

Testing data set	**Accuracy**
Mixed data	96.8%
Two sedans, Local, Moderate traffic	98.5%
Two sedans, Freeway, Moderate traffic	97.9%
Coupe and SUV, Local, Moderate traffic	97.7%
Coupe and SUV, Freeway, Moderate Traffic	99.4%
Coupe and SUV, Local, Heavy Traffic	92.8%
Coupe and SUV, Freeway, Heavy Traffic	95.7%
Two sedans on a freeway with curves	94.0%

ure 7), where the four radios are placed above the four wheels of the vehicle; (ii) Driving speed does not affect the accuracy significantly; (iii) Heavy traffic conditions and light traffic conditions have different path loss patterns, but their own patterns are consistent which allows us to train the model with mixed traffic data and still get high classification accuracy; (iv) The classifier's prediction accuracy varies with variation in vehicle's body but after combining the training data of different car bodies, it is possible to achieve high prediction accuracy; and, (v) Our design is robust to placement errors of the radios. Finally, after mixing all the training data and testing it individually on each of the testing data sets, we observed that the classifier has 96.8% pediction accuracy when tested under moderate traffic conditions. In heavy traffic condition and curved drive, the accuracy will decrease, but the overall accuracy remains above 92.8%.

6 Increasing Robustness through Aggregation

In Section 3, we discussed how the radios on the vehicle can be used to determine the relative location of two vehicles. However, sometimes it may lead to incorrect results. In this section, we explain how the accuracy of localization can be further increased through aggregation, e.g., if V_1 is left of V_2 and V_2 is left of V_3, then it is possible to infer that V_1 is also left of V_3. Similar kind of aggregation can be used for front and rear relationships. Accuracy of relative localization can be improved by applying aggregation rules separately to *left-same-right* and *front-rear* relationships between neighboring vehicles. Consequently, in this section, we discuss two algorithms. The first one improves the accuracy of lane localization while the second improves front and rear localization accuracy. Both the algorithms are distributed in nature and are executed by every vehicle in the network. Whenever the phone (or the corresponding vehicle) determines its relative position with respect to some neighboring vehicle, it executes both the algorithms to further improve the robustness. However, desinging an aggregation algorithm that improves localization accuracy involves the following challenges: (i) To avoid dependence on a centralized server, it is preferable to use a distributed algorithm; (ii) The set of neighbors of a vehicle may change with time, making it hard to find a fixed set of aggregating vehicles; and, (iii) The relative location determined by the SVM classifier may be incorrect in some cases.

6.1 Aggregating relative lane localization information

To improve the accuracy of left-same-right relationship, we assign every vehicle a *coordinate system* and a *virtual lane number* corresponding to that coordinate system. Virtual lane numbers are *comparable* among vehicles that belong to the same coordinate system, i.e., any two vehicles that belong to the same coordinate system have the same virtual lane numbers *iff* they are in the same physical lane of the roadway. Similarly, a vehicle on the left side has higher virtual lane number than a vehicle on the right side if both of them belong to same coordinate system. The virtual lane location is computed using the relative locations with respect to multiple neighbors. Thus, it can fix errors that may arise in relative localization.

A coordinate system C_a is denoted by the tuple (T_a, I_a) where T_a is the timestamp when the coordinate system was created and I_a is the ID of the vehicle that created the coordinate system. A coordinate system C_a is said to be smaller than coordinate system C_b if one of the following conditions is true: (i) $T_a < T_b$; or, (ii) $T_a = T_b$ and $I_a < I_b$. A vehicle may create a new coordinate system if any of the following conditions is true: (i) When the app on the phone is started; (ii) When the app detects that the vehicle is in motion after a stop that lasted more than T_{stop} time; or (iii) If the accelerometer triggers a possible lane change event or a turn event. Upon creation of a new coordinate system with the current timestamp and its own ID, the vehicle (say V_i) initializes its lane number (L_i) in the new coordinate system to be 1 and the corresponding probability (α_i) also as 1.

After initializing its coordinate system, V_i may seek to join the coordinate system of another vehicle. For this, it communicates with all its neighbors, to obtain the information of their current coordinate systems and their lane numbers; it then joins the smallest coordinate system among all its neighbors (Lines 1-2 of Algorithm 1). This ensures that eventually all neighboring vehicles will belong to the same coordinate system, thereby increasing the chances that their virtual lane numbers are comparable. When joining the new coordinate system, V_i also determines its lane number with the help of its neighbors (denoted by S_{low}) that are in the smallest coordinate system (Lines 3-19). Then, V_i determines its relative location with respect to all vehicles in S_{low} using the algorithm discussed in Section 3 (Line 6). Then, it proceeds to compute its lowest possible lane (min) and highest possible lane (max) numbers (Line 7) which are one lower and one higher than the lowest and highest lane number of all vehicles in S_{low}, respectively.

Then, for every lane among the possible lanes from min to max, it computes the confidence that the vehicle is located in that lane (Lines 8-17). To compute the confidence of being in a particular lane (say l), it computes the relative location of l with the lane number of vehicle V_j (Lines 11-16). This confidence is computed over all vehicles and their sum ($confi_l$) reflects the confidence that V_i is in l based on information of all vehicles in S_{low} (Line 17). Here c is the output of the SVM based classifier (see Section 3) and its value lies between 0 and 1. Finally, the lane that maximizes the $confi_l$ is designated as the lane number of V_i and its probability value (α_i) is also updated by normalizing it across all possible lanes (Lines 18-19). For increased robustness, each vehicle in our algorithm, also invokes Algorithm 1 periodically to update its virtual lane number. Since each vehicle now has a virtual lane number, two vehicles can now arrive at a more accurate result for their relative localization by just comparing their virtual lane numbers.

6.2 Aggregating front-back localization information

Similarly, in order to improve the accuracy of the front-back localization, each vehicle obtains relationship information with its neighbors. Using that information and its own relative position with respect to its neighbors, it constructs a relationship graph $G = (W, E)$ where W is the set of vehicles (See Algorithm 2). Further, there is an edge from vehicle V_a to V_b *iff* V_a believes that V_b is in its front[2]. Presence of a directed cycle in G implies that relative location for at least one pair of vehicles is incorrect (See lemma A.1 for proof). Therefore, by removing cycles from G, Algorithm 2 increases the accuracy of relative localization.

Since counting the number of cycles in a graph is an NP-Hard problem (See theorem A.1 for proof), therefore, we define a heurisitcal metric $\beta(G)$ that cap-

[2]Equivalent to saying that V_b believes that V_a is in rear since the vehicle that computes relative location passes on the result to the other vehicle (Section 3.3.3)

Algorithm 1: For a given vehicle (V_i), distributed algorithm for updating coordinate system (C_i), lane number (L_i) and corresponding probability (α_i) for V_i

```
1  For every neighbor $V_j$, obtain $V_j$'s coordinate system ($C_j$), $V_j$'s most probable
   virtual lane number ($L_j$) and probability that $V_j$ is in $L_j$ ($\alpha_j$)
2  $C_{low} \leftarrow \arg \min_{j:V_j \text{ is a neighbor of } V_i} C_j$
3  $S_{low} \leftarrow$ Set of neighboring vehicles that are in $C_{low}$
4  if $C_{low} \leq C_i$ then
5      $C_i \leftarrow C_{low}, L_i \leftarrow \phi$
6      Determine relative lane location with respect to each vehicle in $S_{low}$.
7      $min \leftarrow \min_{j:V_j \in S_{low}} L_j - 1, max \leftarrow \max_{j:V_j \in S_{low}} L_j + 1$
8      for $l = \min \ to \ \max$ do
9          $confi_l \leftarrow 0$
10         for all Vehicle $V_j \in S_{low}$ do
11             if $L_j - l > 0$ then
12                 $c \leftarrow$Confidence that relative location of $V_j$ with respect to $V_i$ is left
13             if $L_j - l = 0$ then
14                 $c \leftarrow$Confidence that relative location of $V_j$ with respect to $V_i$ is
                   same lane
15             if $L_j - l < 0$ then
16                 $c \leftarrow$Confidence that relative location of $V_j$ with respect to $V_i$ is
                   right
17             $confi_l \leftarrow confi_l + \alpha_j \times c$
18     $maxLane \leftarrow \arg\max_i confi_i$
19     $L_i \leftarrow maxLane$, $\alpha_i \leftarrow \frac{confi_{maxLane}}{\sum_{l=min}^{max} confi_l}$
```

tures the number of cycles in G. To determine if a directed edge (V_i, V_j) is a part of some cycle, we only need to check if V_i is reachable from V_j.

$$\beta(G) = \{|e : e \in E \text{ and } e \text{ is part of some directed cycle}|\}$$

To remove cycles from G without losing any relative location information, Algorithm 2 reverses edges in G until G becomes acyclic. For this, at each step, it iteratively reverses the edge that reduces the number of cycles in G by the largest amount. Reversing an edge (V_a, V_b) indicates that based on information of neighboring vehicles, it is more likely that V_a is actually in front of V_b. To minimize the number of edges reversed, Algorithm 2 first tries to find a single edge that can be reversed to reduce the number of cycles (Lines 5-11). However, it is possible that it is not able to find any such edge even though the graph is cyclic. In that case, it picks a vertex and reverses all the incoming edges (Lines 15-19) or all the outgoing edges (Lines 20-24). Algorithm 2 performs this search over all vertices and performs the operation on the vertex that maximizes the number of cycles reduced (Lines 13-25). It can be shown that Algorithm 2 eventually terminates and has a polynomial time complexity (For proof, see theorem A.2).

7 Simulations

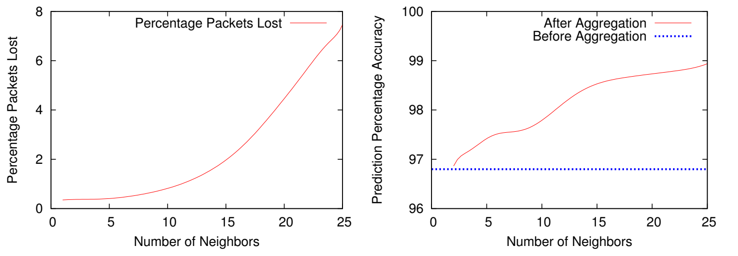
In Section 5, we described the experiment results obtained on a variety of vehicles. However, evaluating the behavior and performance of our algorithm for a larger set of vehicles was labor intensive as every vehicle requires one driver and one passenger to record the ground truth. So, we performed *trace-driven simulations* using ns-3 [1] and SUMO [5]. SUMO is a simulator for VANETs which given a road network, generates a pre-determined number of trips for vehicles. For each trip, it chooses a random starting point, a random ending point and generates a route for the vehicle completing that trip. SUMO is a microscopic-level road traffic simulator which implies that SUMO models multiple lanes in the roadways and vehicles in SUMO perform automatic lane-changes and overtaking of vehicles as they move from their starting point to their ending point in the map. The objectives of the simulations were as follows: (i) To determine the increase in accuracy achieved by using aggregation algorithms; and, (ii) To determine the packet loss rate and its affect on accuracy when multiple vehicles are running MARVEL.

For our simulations, we chose a 3 miles $\times$ 3 miles area around downtown of Austin[3]. Further, we used SUMO to generate 1000 trips (or vehicles) and their

[3]Randomly chosen city

Algorithm 2: For a given vehicle (V_i), distributed algorithm for updating front-back relationship of V_i with respect to its neighbors

```
$G = (W, E) \leftarrow$ Front-back graph based on neighborhood info where
$W \leftarrow \{V_i\} \cup$ Neighbors of $V_i$
$E \leftarrow \{(V_a, V_b) : V_a$ believes that $V_b$ is in its front $\}$
while $\beta(G) > 0$ do
    $maxdiff \leftarrow 0, edgeToReverse \leftarrow \phi$
    for all $e \in E$ do
        $E' \leftarrow E \backslash \{e\} \cup \{\overline{e}\}, G' \leftarrow (W, E')$
        if $\beta(G) - \beta(G') > maxdiff$ and $\overline{e}$ is not a part of some directed cycle in $G'$ then
            $maxdiff \leftarrow \beta(G) - \beta(G')$, $edgeToReverse \leftarrow e$
    if $maxdiff > 0$ then
        $E \leftarrow E \backslash \{edgeToReverse\} \cup \{\overline{edgeToReverse}\}$
    else
        $edgesToReverse \leftarrow \phi$
        for all $V_i \in W$ do
            $F \leftarrow \{(V_j, V_i)\ for\ some\ V_j \in W\}$
            $\overline{F} \leftarrow$Obtained after reversing all edges in $F$
            $E' \leftarrow E \backslash F \cup \overline{F},\ G' \leftarrow (W, E')$
            if $\beta(G) - \beta(G') > maxdiff$ then
                $maxdiff \leftarrow \beta(G) - \beta(G')$, $edgesToReverse \leftarrow F$
            $F \leftarrow \{(V_i, V_j)\ for\ some\ V_j \in W\}$
            $\overline{F} \leftarrow$Obtained after reversing all edges in $F$
            $E' \leftarrow E \backslash F \cup \overline{F},\ G' \leftarrow (W, E')$
            if $\beta(G) - \beta(G') > maxdiff$ then
                $maxdiff \leftarrow \beta(G) - \beta(G')$, $edgesToReverse \leftarrow F$
        $E \leftarrow E \backslash \{edgesToReverse\} \cup \{\overline{edgesToReverse}\}$
```



(a) Percentage Packets Lost (b) Localization accuracy before and after performing aggregations

Figure 8: Simulation results with varying number of neighbors.

corresponding routes. The simulation was executed for 900 seconds which was enough for every vehicle to reach its destination. The output of SUMO (position of every vehicle at each instant of time) was used for generating the locations of corresponding nodes in ns-3.

In ns-3, every vehicle consisted of 5 nodes: a smartphone and the 4 wireless radios. The mobility of the 5 nodes is generated from the SUMO's output. The smartphones send UDP *discovery beacons* every 10 seconds that are heard by neighboring smartphones. The range of radio to radio communication was set to 65 meters while the range of smartphone to smartphone communication was set to 100 meters.

The wireless radios transmitted *localization beacons* when directed by the smartphone. To simulate the effect of vehicle's body and the neighboring urban structures (parked cars, other vehicles and store fronts) on the signal strength, we took the signal strength values at the receiving wireless radios from the data obtained through experiments corresponding to the ground truth location of the 2 vehicles. However, the packets from one wireless radio (or smartphone) to another were still dropped by ns-3 if some other neighboring node was also observed to be transmitting at the same time. Similar to experiments, here also every wireless node transmitted 5 *localization beacons* at random spacings of 25-50 ms. The smartphone computed the relative location of its vehicle with respect to neighboring vehicles by passing the 16 averaged RSSI values to the SVM classifier. For the simulations, we assumed that the smartphone triggers lane change events with 100% recall and 60% precision (See Section 4).

Simulation Results: Figure 8a shows that packet loss rate increases with increase in number of neighbors. The average packet loss rate across all nodes was observed to be 2.3%. Since, in MARVEL, radio nodes broadcast *lane discovery* beacons multiple times, it is expected that this low loss rate would not decrease the accuracy. However, with increase in number of neighbors, it is possible for MARVEL to increase accuracy by making use of aggregation. Figure 8b shows the variation in localization accuracy before and after we performed aggregation using the two algorithms. With aggregation, MARVEL is able to improve the localization accuracy to 99% when the number of neighbors is very high. When number of neighbors is low, aggregating information across vehicles provides limited benefit. Over all vehicles in the simulation, we observed that aggregation improved accuracy to 98%.

8 Discussion and Future Work

In this section, we discuss some mechanisms that can make MARVEL more accurate and practical.

8.1 Increasing lifetime of wireless radios

By using the accelerometer of the phone to trigger the relative localization algorithm, we have been able to avoid periodic triggering of wireless radios, thereby increasing their lifetime. Results from our experiments and simulations show that the standard TelosB motes exhibit a battery lifetime of at least 6 months for an average of 2 hours daily driving and a maximum reaction time of 1 second after the lane change event is triggered (See Section B for details). To remove the dependence on batteries, the possibility of connecting the wireless radios directly to the vehicle's power supply can also be explored.

8.2 Incremental Deployment

MARVEL can determine relative location only if both vehicles are equipped with wireless radios. It cannot determine the relative location if only one of the vehicles is equipped with radios. However, MARVEL can provide incremental benefit to vehicles that are equipped with 4 radios. In USA, FCC has already reserved 75 MHz of spectrum for Dedicated Short Range inter-vehicle Communication (DSRC). It is expected that in the future, all vehicles will be equipped with at least one antenna for DSRC. Our experiments show that if one of the vehicles is equipped with only one antenna (located on the vehicle's body near the front passenger side wheel) and the other with 4 antenna, MARVEL correctly predicts the relative location with 64% accuracy. Therefore, as DSRC becomes more common, vehicles with 4 antennas would be able to use MARVEL with 64% and 96.8% accuracy when they encounter other vehicles with a single antenna and with four antennas, respectively. Using simulations, we also observed that if 50% of the vehicles have 1 antenna while the other 50% have 4 antennas, then average localization accuracy was 87% for one antenna-four antenna vehicle pairs and 96% for four antenna-four antenna vehicle pairs. Thus, with aggregation, it is possible to achieve higher localization accuracy even when all vehicles are not equipped with four radios.

8.3 Training Cost

To achieve higher localization accuracy, it is beneficial to train MARVEL for the specific vehicle. As we saw in Section 5.5, it still provides an accuracy of 90% when trained on a vehicle with different physical profile. Also, in our experiments we saw that testing a classifier on sedan-sedan pair that is trained with sedan-coupe pair still gives 96% accuracy. This implies that it is not necessary to train MARVEL separately for vehicles that have similar physical profiles, although to get higher accuracy it is beneficial to train MARVEL on only those vehicles that have significantly different profiles.

9 Conclusion

In this paper, we proposed a novel antenna diversity based solution called MARVEL, for determining the relative location of two vehicles on roadways. Relative location information has the potential to not only enhance the driving experience by providing relevant alerts but also reduce the chance of collisions. MARVEL has low cost and is easy to install on newer as well as exisiting vehicles. Results from our driving experiments performed under varying conditions show that MARVEL predicts relative location of two vehicles with an average accuracy of 96.8%. MARVEL is able to determine the relative position of vehicles that are not in the immediate neighborhood, thereby giving the vehicle driver more time to react. To reduce energy consumption of wireless radios and to reduce the number of packets transmitted, we also proposed using the phone's accelerometer to trigger the localization algorithm. We presented two algorithms that increase the localization accuracy by aggregating information across multiple vehicles. Our trace-driven simulations show that by aggregating information, MARVEL is able to increase the localization accuracy to 98%.

10 Acknowledgements

This material is based upon work partially supported by the National Science Foundation under Grants CNS-0721817 and CNS-0831919.

References

[1] The ns-3 network simulator. http://www.nsnam.org/.

[2] Valeo Raytheon Systems. www.valeo.com.

[3] AASHTO. *A Policy on Geometric Design of Highways and Streets*. 2001. pp 384–386.

[4] Sanjeev Arora. Lecture 7: Counting Classes. `http://www.cs.duke.edu/~reif/courses/complectures/Arora/lec7.pdf`.

[5] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz. SUMO - Simulation of Urban MObility: An Overview. In *Proc. of Third International Conference on Advances in System Simulation (SIMUL)*, 2011.

[6] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2007.

[7] R. Chellappa, G. Qian, and Q. Zheng. Vehicle Detection and Tracking Using Acoustic and Video Sensors. In *Proc. of IEEE Int'l Conf. Acoustic, Speech, and Signal Processing*, 2004.

[8] H. Y. Cheng, B. S. Jeng, P. T. Tseng, and K. C. Fan. Lane Detection with Moving Vehicles in the Traffic Scenes. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):571–582, 2006.

[9] Moteiv Corporation. Telos Datasheet. `http://www2.ece.ohio-state.edu/~bibyk/ee582/telosMote.pdf`.

[10] Prabal Dutta and David Culler. Practical Asynchronous Neighbor Discovery and Rendezvous for Mobile Sensing Applications. In *Proc. of ACM SenSys*, pages 71–84, 2008.

[11] Energizer. 3-315WC Data Sheet. See graph on tape players, `http://data.energizer.com/PDFs/3-315wc.pdf`.

[12] Garmin. What is GPS. http://www8.garmin.com/aboutGPS/index.html.

[13] A. Gern, U. Franke, and P. Levi. Advanced Lane Recognition-Fusing Vision and Radar. In *Proc. of IEEE Intelligent Vehicles Symposium*, 2000.

[14] F. Heimes and H. Nagel. Towards Active Machine-Vision-Based Driver Assistance for Urban Areas. *Int'l J. Computer Vision*, 50:5–34, 2002.

[15] A. Kandhalu, K. Lakshmanan, and R. (Raj) Rajkumar. U-connect: A Low-Latency Energy-Efficient Asynchronous Neighbor Discovery Protocol. In *Proc. of ACM IPSN*, 2010.

[16] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. YALE: Rapid Prototyping for Complex Data Mining Tasks. In *Proc. of ACM SIGKDD*, 2006.

[17] P. Mohan, V. N. Padmanabhan, and R. Ramjee. Nericell: Using Mobile Smartphones for Rich Monitoring of Road and Traffic Conditions. In *Proc. of ACM EMNETS*, 2008.

[18] NHTSA. Evaluation of Lane Change Collision Avoidance Systems Using the National Advanced Driving Simulator. DOT HS 811 332, U.S. Department of Transportation, May 2010.

[19] G. Padmavathi, D. Shanmugapriya, and M. Kalaivani. A Study on Vehicle Detection and Tracking Using Wireless Sensor Networks Open Access. *Wireless Sensor Network*, 2(2):173–185, 2010.

[20] S. Park, K. Kim, S. Kang, and K. Heon. A Novel Signal Processing Technique for Vehicle Detection Radar. *IEEE MTT-S Int'l Microwave Symp. Digest*, 2003.

[21] J. P. Piran, G. R. Murthy, and G. P. Babu. Vehicular Adhoc and Sensor Networks: Principles and Challenges. *International Journal of Ad hoc, Sensor and Ubiquitous Computing*, 2(2):38–49, 2011.

[22] C. Wang, C. Thorpe, and A. Suppe. Ladar-Based Detection and Tracking of Moving Objects from a Ground Vehicle at High Speeds. In *Proc. of IEEE Intelligent Vehicle Symp.*, 2003.

[23] C. C. Wang, S. S. Huang, and L. C. Fu. Driver Assistance System for Lane Detection and Vehicle Recognition with Night Vision. In *Proc. of IEEE Intelligent Robots and Systems*, 2005.

[24] J. Yang, S. Sidhom, G. Chandrasekaran, T. Vu, H. Liu, N. Cecan, Y. Chen, M. Gruteser, and R. P. Martin. Detecting Driver Phone Use Leveraging Car Speaker. In *Proc. of ACM MobiCom*, 2011.

A Proofs

Lemma A.1. *Presence of a directed cycle in G implies that relative location for at least one pair of vehicles is incorrect.*

Proof. If the vehicles (or vertices) (say $V_0, V_1, \ldots, V_k$) form a directed cycle, then a directed edge from V_i to V_{i+1} implies that V_{i+1} is in front of V_i for $i \in \{0, k-1\}$. On the basis of this information, it can be inferred that V_k is in front of V_0 . Also, the edge from V_k to V_0 implies that V_k is in the rear of V_0 which contradicts our previous conclusion. Hence, relative location for at least one pair of vehicles must be incorrect. □

Theorem A.1. *Counting the number of directed cycles in a graph is an NP-Hard problem.*

Proof. See [4] for the proof. □

Lemma A.2.

No edge is reversed by algorithm 2 more than once. Also, algorithm 2 does not create new cycles in the graph G.

Proof. Algorithm 2 may either reverse a single edge or a subset of edges incident to the same vertex. When it reverses only one edge, then it is ensured that the incident edge is not part of any cycle. Similarly, when it reverses multiple edges incident to the same vertex, then none of those edges, after reversal, can be part of any cycle (Since after reversal all edges incident on the vertex are either outgoing or all are incoming). Therefore, once algorithm 2, reverses an edge, then reversed edge is not part of any cycle. This implies that after an edge is reversed, it is not a part of any cycle and it will never become part of any new cycle. Hence, it will never be reversed again. Further, since the reversed edge is not part of any cycle, therefore during its execution, algorithm 2 does not create any new cycles. □

Lemma A.3. *If there is at least one cycle in G, then Algorithm 2 will find some edge that can be reversed such that number of cycles is reduced and $\beta(G)$ is reduced.*

Proof. If G has a cycle, then $\beta(G)$ will be positive since some of the edges in G are part of directed cycle. Therefore, Algorithm 2 will stay in the while loop. Observe that if there is a cycle in G, then all vertices that are part of some directed cycle have at least one incoming edge and one outgoing edge. By reversing the direction of these edges (for some vertex) such that either all of them are incoming or outgoing, it is possible for algorithm 2 to break the cycle. This would also lead to reduction in the value of $\beta(G)$ since the reversal reduces the number of cycles without creating any new cycles (From lemma A.2). Since, the algorithm explores all the vertices of G, therefore, it would find at least one vertex for which reversing the edges incident to it would break at least one cycle and hence reduce the value of $\beta(G)$. $\square$

Theorem A.2. *Algorithm 2 eventually terminates and has a polynomial time complexity.*

Proof. We first show that when Algorithm 2 terminates, then there are no cycles in W: From Lemma A.3, we know that as long as there are cycles in G, algorithm 2 will be able to reduce $\beta(G)$ for every iteration of while loop. A zero value of $\beta(G)$ is possible *iff* G is acyclic. Therefore, in algorithm 2 eventually $\beta(G)$ will become 0, thereby terminating the algorithm.

Since, a graph has $O(|V|^2)$ edges, each edge is reversed at most once (See Lemma A.2), and finding an edge (or edges) to reverse takes polynomial time, therefore, algorithm 2 has a polynomial time complexity.

It is possible to compute shortest path between all pair of vertices using Floyd-Warshall algorithm ($O(|V|^3)$ complexity). This can in turn be used to compute $\beta(G)$ ($O(|V|^3)$ complexity). Since in every iteration of while loop, algorithm 2 will find at least one edge to reverse and each edge is reversed at most once, therefore, algorithm 2 will execute while loop $O(|E|)$ times. Within the while loop, $\beta(G)$ is computed at most $O(|E| + |V|)$ times, therefore it is possible to implement Algorithm 2 with $O(|E|^2|V|^3)$ complexity. In our simulation results, we observed that Algorithm 2 terminates within 10 milliseconds with very high probability since the graph generally consists of 1-hop neighbors of the vehicle and is small in size. $\square$

B Battery Life

In this section, we investigate the battery lifetime of the wireless radios deployed on the vehicle. For this, we assume that off the shelf TelosB motes with AA batteries are being used as wireless radios. We further assume that the vehicle is driven an average of 2 hours everyday (See Table 2). A single AA battery generally provides 2100 mAh of energy before its voltage drops below 1.05 V [11]. We used 1.05 V as the cutoff as the Telosb motes require a minimum of 2.1

Table 2: Parameters for computing battery lifetime

Parameter	**Value**
Average driving time	2 hours
AA battery capacity	2100 mAh [11]
Batteries per Telosb mote	2
Telosb wake up interval (Driving mode)	0.5 seconds
Telosb wake up interval (Vehicle parked mode)	120 seconds
Telosb current consumption with radio and MCU on	21.8 mA [9]
Telosb current consumption in deep sleep mode	5.1 μA [9]
Percentage of wake ups that are long	50% (Section 7)
Long wake up duration	0.4 seconds
Short wake up duration	0.1 seconds

V to operate [9]. To ensure a reaction time of at most 1 second, we set the wake up interval of TelosB motes to be 0.5 second when vehicle is in motion. However, when vehicle is not in motion, then the motes wake up once in 120 seconds to establish communication with the smartphone (if it is there).

When the vehicle is in driving mode, radios would wake up every 0.5 seconds, however, upon waking up, they would not need to perform relative vehicle localization in all the cases. Relative vehicle localization is required only if it is triggered by accelerometer of the smartphone, a new neighboring smartphone is discovered, the vehicle overtakes (or is overtaken) by another vehicle, or when a neighboring vehicle requests it. We observed in our simulations that radios need to perform relative vehicle localization in at most 50% of the instances they wake up.

If wireless radios are performing localization (i.e. wakeup is *long*), then their wakeup duration is assumed to be 0.4 seconds, while if they do not perform localization, then it is possible for them to go back to sleep with in 0.1 seconds. In our experiments, we observed that the duration of 0.4 seconds was enough for the motes to (i) listen for the message from the phone; (ii) Transmit or receive localization beacons; and (iii) Report RSSI values back to phone.

From these parameters, we can conclude that the average wake up duration = $0.5 \times 0.4 + 0.5 \times 0.1 seconds = 0.25$ seconds Therefore, when the vehicle is driving, then the total time spend my motes in awake state is = $\frac{2 \times 3600}{0.5} \times 0.25 = 3600$ seconds Similarly, when the vehicle is in parking mode, then the total time spend by motes in awake state is = $\frac{22\times3600}{120}\times 0.1 = 66$ seconds The total time spend by motes in awake state per day = $3600 + 66 = 3666$ seconds. The total time spend by motes in standby state per day = $24\times3600-3666 = 82734$ seconds. Based on these values, we can finally compute the battery lifetime of motes as = $\frac{2\times2100\times3600\ mAs}{3666\times21.8+82734\times5.1\times10^{-3}\ mAs\backslash day} = 188$ days