ForeSight: Mapping Vehicles in Visual Domain and Electronic Domain

Dong Li, Zhixue Lu, Tarun Bansal, Erik Schilling and Prasun Sinha
Department of Computer Science and Engineering, The Ohio State University
{lido, luz, bansal, schillie, prasun}@cse.ohio-state.edu

***Abstract*—Using broadcast in vehicular applications such as autonomous cruise control and collaborative driving can disturb unrelated drivers and fail to convey the message due to unspecified receiver, resulting in increased risk of accidents. For supporting the unicast communication primitive, it is important to know the electronic identities (EIDs), e.g., the IP addresses and the relative positions of the nearby vehicles. We show that the estimated GPS coordinates alone are not accurate enough to uniquely identify the intended vehicle. On the other hand, there is an increasing array of devices, such as on-board camera, RADAR, and DSRC radio that are becoming available in newer vehicles. These heterogeneously deployed devices provide information sources that have varying levels of accuracy and potentially different coverage regions, making it challenging to accurately identify the vehicle. As a first step, we design ForeSight, a system that dynamically integrates the information observed in the visual domain (e.g., from camera) and the electronic domain (e.g., WiFi radio) to match the vehicles observed in these two domains with high accuracy. The experiment and simulation results show that ForeSight is able to significantly improve the vehicle identification accuracy compared to using GPS or other algorithms. In our case study, ForeSight reduces disturbing messages by 14 $\times$ as compared to the number of a GPS-based communication method.**

I. Introduction

The communication between vehicles has traditionally been achieved using honks, yells, emergency alarms, etc. Such communication methods are broadcast in nature, and their success relies on the correct interpretation by their intended targets. Unicast communication is very important among drivers on the road and is useful in various instances such as to alert others when overtaking and passing other vehicles, to remind someone to turn on their headlights or to negotiate right-of-way at intersections. With emergence of Computer-Assisted Driving and Autonomous Driving technologies, reliable and accurate vehicle-to-vehicle communication is becoming increasingly crucial. One way to communicate in vehicular networks is to simply broadcast the message. However, if the IP address of the target vehicle is unknown, then the broadcast message may cause confusion at unintended receivers. For example, a driver performing an overtaking operation only needs to alert a specific vehicle that is passed by; and a driver whose view is blocked by a truck in front needs to get video images only from this truck. In these cases, broadcast messages could be misinterpreted by the unintended receivers. So, unicast primitive, which allows communication between a specific pair of vehicles is critical for vehicle-to-vehicle communication. Unicast messages when sent over the broadcast wireless channel will only be processed by the intended receiver, resulting in fewer misinterpretations.

This material is based upon work partially supported by a gift from Toyota InfoTechnology Center.

978-1-4799-3360-0/14/$31.00 ©2014 IEEE

For supporting unicast communication primitive, it is important to know the electronic identities (EID, e.g., the IP addresses) of the target vehicle. However, due to the dynamic nature of road-traffic, the EID of neighboring vehicles could be unknown or unpredictable. This makes the unicast communication problem especially challenging. Many existing protocols simply use broadcast communication [1] or assume that the target vehicle can be uniquely identified using the GPS coordinates [2]. However, this approach has two problems: First, GPS coordinates may be inaccurate. For example, [3] found that GPS reports the wrong lane number in 54% of the time; Second, even if the GPS receiver is accurate, it still requires the transmitting vehicle to estimate the GPS positions of the target vehicle (e.g., by camera). However, this estimation is inaccurate especially in bad light conditions and can introduce more inaccuracies. Similarly, additional features may be subject to different types of inaccuracies. For example, the plate number can uniquely identify a vehicle, however, it may not be always in the camera's view or may not be read correctly. Similarly, correctness of vehicle color measured by the camera depends on light conditions, distance and the quality of the camera itself.

Toward this, we design ForeSight, a solution that provides 2-way *lookup* service: i) **Given physical characteristics of a vehicle, it provides the IP address of the matching neighboring vehicle; and the reverse service**; ii) **Given the IP address of the vehicle, it provides the relative location of that vehicle**. The first service is useful when the vehicle needs to send unicast messages. Examples include notifying a neighboring vehicle of its broken lights, and negotiating right-of-way at an intersection. The reverse lookup service is useful when navigating to a known vehicle. For example, when joining an autonomous platoon of vehicles, the service can navigate the driver/vehicle to the desired vehicle to latch on.

ForeSight takes advantage of the increasing availability of new vehicle-equipped devices, such as rear-view cameras, and the popularity of smartphones. ForeSight assumes that a vehicle deployed with the system is equipped with one or more cameras, a GPS and a radio interface, such as 802.11 / WiFi. The cameras, GPS and DSRC built in some vehicles can serve such purposes. If the vehicle does not have such devices, ForeSight can use the camera, GPS and WiFi interface in the smartphone, assuming the smartphone is attached to the dashboard of the vehicle. Each vehicle in ForeSight is assigned a unique EID (such as MAC/IP address). Each vehicle assigns visual identities (VIDs) to vehicles that are detected in its camera. The VID is comprised of a set of features (color, make, model, etc) that describes the vehicle. Due to the differences in light conditions and hardware, it is possible that different vehicles may compute the VID of the same vehicle differently. There are devices used in self-driving cars like LIDAR that can model the 3D environment of a vehicle more accurately, however, it may not be available on all vehicle owing to its high cost (more than $\$70,000$ USD [4]). Designed as a low-cost solution, ForeSight does not assume the features of VIDs are measured 100% accurately. ForeSight is flexible in design such that it can use devices and features of different accuracy. ForeSight implements the two look-up services by matching the EIDs with VIDs. After that, through the available UI (touch, gesture, voice, etc.), the driver can send message to matched vehicles shown on the screen, and the system may automatically send a message to another car when needed, such as when overtaking. And if a message is received, the system can usually represent the sender to the driver through the sender's image or relative position.

The challenges of this work are as follows: i) **Non-uniformity in accuracy and reliability:** The overall matching precision needs to be improved using features that are individually inaccurate and unreliable. Fusing these features is particularly challenging since the individual accuracy of these features depend on environment (e.g., GPS is more accurate in open areas than downtown) and prevalence (e.g., if all the neighboring cars are black, then *color* is not a good feature). ii) **Information asymmetry:** Due to difference in visual range and RF range, and due to limited view of the camera, it is possible that a vehicle may "see" another vehicle but not "hear" it or vice-versa. This asymmetry would result in imperfect matching, which is a critical issue for precise matching. iii) **Limited deployment:** If some vehicles do not have ForeSight deployed, then it may cause confusion as such vehicles can be seen by other vehicles but cannot be heard by any other vehicle in the system. This paper makes the following contributions:

- This paper presents ForeSight, the first solution that provides the functionality for looking up the IP address of neighboring vehicles that can be used for implementing unicast communication among vehicles.
- We propose two algorithms (Adaptive Weight algorithm and Cluster-AW algorithm) to fuse different features that dynamically assign weights to these features depending on the environment and the importance of the individual features. Its performance is significantly better than existing algorithms.
- We conducted a real world driving experiment that shows ForeSight can effectively improve the vehicle matching accuracy. Using comprehensive simulations under realistic settings, we show that ForeSight can achieve more than 90% of matching precision with 90% of recall only using currently observed GPS and color information.

II. SYSTEM DESIGN

This section gives an overview of the system and describes the feature model used in ForeSight.

A. System Overview

The minimum hardware requirement of ForeSight includes at least one camera, one GPS receiver and a radio interface. Cameras, GPS, DSRC, WiFi modules on either the vehicle or the smartphone can serve such purposes. Cameras are used to identify neighboring vehicles, assign VIDs and measure visual features, while the radio interface is used to collect EIDs and for other communication purposes. If multiple cameras are available on one vehicle, ForeSight can use them simultaneously, thereby identifying more visual neighbors.

A vehicle in ForeSight constantly measures the values of a set of features for itself and its visual neighbors. Features will be discussed with more details in Section II-B. A vehicle periodically broadcasts its ID and its own feature values to allow itself to be discovered and matched by neighboring vehicles. The ID and the feature values can be stored in a few bytes, and hence transmitted with low network overhead. Such heart-beat packet broadcasting will occur in the background without disturbing the drivers. When a given vehicle (say C), starts a new matching (say at time t), it first identifies its visual neighbors, assigns VIDs to them by measuring their features. At the same time, C retrieves the collected EIDs and their features. The feature values can also be computed from recent measurements if not available at time t with timeline alignment and synchronization techniques [5]. If there are unmatched VIDs and EIDs, C starts the matching procedure. Let $N_v(C)$ denote the set of visual neighbors that can be detected from the camera of C at time t, and $N_e(C)$ denote the set of electronic neighbors it has heard at time t. ForeSight matches $N_v(C)$ with $N_e(C)$ by comparing C's own measurements of the features with the measurements reported by $N_e(c)$. The probability that a VID and an EID is for the same vehicle can be modeled based on the similarity of the feature values. *Unlike previous works using Bayesian filtering, ForeSight focuses on the matching of vehicles at one time instance.* This allows ForeSight to work in dynamically changing environments. In ForeSight, $N_v(C)$ and $N_e(C)$ are not necessarily from the same set of vehicles in the real world, which makes the problem of precise matching more challenging.

In the following, we define the vehicle matching precision and recall, which are the evaluation metrics. Let $\mathcal{G}$ denote the set of vehicles that C has received in the electronic channel and have also been assigned VIDs by C, and let $\mathcal{R} \subseteq \mathcal{G}$ denote a subset of $\mathcal{G}$ whose corresponding VIDs are included in the matching result. Then *recall* is defined as $\frac{|\mathcal{R}|}{|\mathcal{G}|}$. Similarly, let $\mathcal{P} \subseteq \mathcal{R}$ denote the set of vehicles whose VIDs are correctly matched with their EIDs, and H denote the total number of matched pairs, *precision* is defined as $\frac{|\mathcal{P}|}{H}$. *The objective of ForeSight is to maximize precision for a given recall requirement.*

B. Feature Model

ForeSight uses a set of features to describe a vehicle. A feature is supported in ForeSight as long as it satisfies the two conditions: 1) it is known to (or can be measured by) the vehicle itself; and 2) it can be measured by its neighboring vehicles through camera. Such features include vehicle color, make, model, speed, plate number, lane number, GPS coordinates, etc. On the other hand, vehicle weight is not considered as a feature in ForeSight. The reliability of a feature depends on various factors such as light condition, weather condition, and the quality of the devices. For example, although plate number is a feature that could uniquely identify a vehicle, however, it may not be visible when it is far from the camera. Let $\mathcal{F}$ denote the set of features that are used to identify a vehicle (say A). We call a feature as *electronic feature* when it is measured by A itself. It is named electronic feature because the vehicle A will notify other vehicles of its own measurements through the electronic channel. When a feature of A is measured by other vehicles, the feature is called a *visual feature*. For example, A knows its own color and sends it to other vehicles. This color feature is an electronic feature. On the other hand, when vehicle B measures the color of A with a camera, the measured color of A is a visual feature. Note that a feature may be measured differently by different neighboring vehicles. For any feature f, we assume there exists a *similarity* function $S_f(m_f, m'_f)$ that describes how likely two feature values m_f and m'_f are two measurements of the same vehicle.

III. The Adaptive Weight Algorithm

Estimating the similarity between two sets of various features is a classical problem. Methods widely used in data fusion and data mining domains include cosine similarity, Mahalanobis distance, PCA etc. We present a novel Adaptive Weight Algorithm (AW), which considers the distinguishability of different features in the matching. It has better performance than commonly used algorithms because it selects features that best distinguish the vehicles.

A. The Adaptive Weight Algorithm

Given two vehicles, a visual neighbor (say v_i) and an electronic neighbor (say e_j), we want to calculate $S(v_i, e_j)$, i.e., the probability that v_i and e_j correspond to the same vehicle in the real world. Recall that both v_i and e_j are defined by a set of features. Since there is no obvious hierarchy among the features, a parallel topology [6] is used to fuse them. Owing to difference in accuracy and prevalence of different features, we observe that manually assigning a static weight for each feature will not be suitable for the dynamic vehicular environments. For example, vehicle color is more distinctive during day time than at night, whereas vehicle lane number has low importance in distinguishing vehicles when all vehicles are in the same lane. This motivates the requirement for computing the weights dynamically (on the basis of the current environment) while taking into account the accuracy and the prevalence of all the features. Assume $\mathbf{m}_f = \{m_f^i, i \in 1 \cdots N\}$ is a vector of all the measurements of a feature f on N vehicles, and m_f^i is the measured value of vehicle i. For a given measurement m_f^i, we define the *distinguishability* of m_f^i as the probability that m_f^i is different with the other measurements in $\mathbf{m}_f$, and denote the probability as $D(m_f^i)$. If a measurement has high distinguishability, it implies that with high probability, this measurement can be used to uniquely identify the vehicle. For example, if a single vehicle is yellow while all other vehicles in neighborhood are black, then the distinguishability of the yellow measurement would be high. The probability that measurement m_f^i is from the same vehicle as measurement m_f^j but different with the other measurements is $S_f(m_f^i, m_f^j)\prod_{k\neq j, k\neq i}(1 - S_f(m_f^i, m_f^k))$. Then

$$D(m_f^i) = 1 - \sum_{j\neq i} S_f(m_f^i, m_f^j) \prod_{k\neq j, k\neq i} (1 - S_f(m_f^i, m_f^k)). \quad (1)$$

The distinguishability of $\mathbf{m}_f$ is defined as the mean of the distinguishability of individual elements:

$$D(\mathbf{m}_f) = E(D(m_f^i), m_f^i \in \mathbf{m}_f). \quad (2)$$

The AW algorithm first estimates the distinguishability of each feature with all the measurements using Equation (2), and assigns the weight of each feature $w_f = D(\mathbf{m}_f)$ with constraint $\sum_{f\in\mathcal{F}} w_f = 1$. The similarity between vehicle v_i and e_j is defined as the harmonic mean of individual feature similarity:

$$S(v_i, e_j) = \left\{ \sum_{f\in\mathcal{F}} w_f / S\left(m_f^{e_i}, m_f^{v_j}\right) \right\}^{-1}. \quad (3)$$

An important property of harmonic mean is that when w_f is given, the resulting similarity tends to 0 if any S_f tends to 0. This means that if the similarity between two vehicles on any feature is extremely low, the resulting mean similarity should be low even if the weight of the feature is high. The harmonic mean is also known to be good at mitigating the influence of large outliers. For example, if the similarities between two vehicles are small on most features, but large on the remaining small number of features, then those large values are probably outliers. By using harmonic mean, their influence could be mitigated.

When the number of measurements is large, the computing complexity becomes a drawback of this algorithm. In this case, more likely there exists a j such that $S_f(m_f^i, m_f^j) \sim 1$. With this condition, Equation (1) can be approximated by only considering the nearest neighbor of m_f^i when calculating $D(m_f^i)$, which is

$$D(m_f^i) = 1 - \max_{j\neq i} S_f(m_f^i, m_f^j). \quad (4)$$

The AW algorithm can dynamically adapt to the changes in the V2V environment. It will assign low weights to features that cannot be used to distinguish the vehicles. For example, if all vehicles are in the same lane, for the lane number feature $1 - S_f(m_f^i, m_f^k) = 0$ in Equation (2) and thus, $w_f = 0$. So, the lane feature will have no contribution in distinguishing the vehicles. However, if all the plate numbers are recognized correctly, then w_{plate} will be 1, resulting in higher weight for plate number feature and thus, higher emphasis during the fusion process.

After determining the similarity between VIDs and EIDs with AW, the problem can be modeled as a bipartite graph matching problem or a more advanced structure matching problem, to determine the matching between VIDs and EIDs. In the bipartite graph matching model, a graph $G = (\{N_v(C), N_e(C)\}; E)$ is created. The two set of vertices in G is the VIDs and EIDs, and the weight of edge $(v_i, e_j) \in E$ can use the similarity between v_i and e_j produced by the AW algorithm.

In comparison, the cosine similarity is not able to assign different weights to different features automatically. The Mahalanobis distance method can determine the distance of two measurements based on the correlation of different features, however, it does not consider the distinguishability of the same feature measurements. The PCA (principle component analysis) can transform the measurements to different dimensions. It may lose information if data in different dimensions are not correlated. It also does not consider distinguishability between measurements.

B. Adaptive Weight with clustering

The key component of the AW algorithm is to accurately estimate the weight for each feature by measuring its distinguishability. However, in vehicle matching, a vehicle only needs to distinguish with its neighboring vehicles. Therefore in a large collection of measurements, w_f may not represent the feature distinguishability accurately for a small group of vehicles. For example, if there is a set of vehicles in two lanes, the lane feature would be considered as an important feature in distinguishing the vehicles. However, to distinguish the subset of vehicles in one of the lanes, the lane feature is no longer important. To address this problem, we create the Cluster-AW method that estimates w_f more accurately. When there are multiple EIDs and VIDs, we put the VIDs and EIDs in the same data set and apply a clustering algorithm on the data. The algorithm clusters the measurements into one or several clusters. Inside each cluster, if the matching is still not finished, i.e., there are multiple VIDs or EIDs in the cluster, we will apply the AW algorithm on the smaller set of vehicles. The similarity of a VID and an EID inside a cluster is based on the AW algorithm, and the similarity of VIDs and EIDs in different clusters is set to 0.

IV. The DoubleCheck Algorithm

The cameras built in vehicles are typically rear-view cameras. On the other hand, drivers are more likely to attach their smartphones on the dashboard, with the cameras facing front. As a result, cameras in different vehicles may face different directions, and a vehicle may have multiple cameras. In this section, we introduce an algorithm called DoubleCheck (DC), which takes advantage of multiple cameras facing different directions to further improve vehicle matching results.

The main idea of DC is that it exploits the mutual measurements from two vehicles to determine the similarity between a VID and an EID. If two vehicles can observe each other through cameras, these two vehicles have a chance to verify their matching results. When a vehicle C estimates the similarity between a visual neighbor v_i and any electronic neighbor e_j, C first determines the direction of v_i in the camera of C, which is denoted by $\theta(v_i)$. If e_j has a camera that could

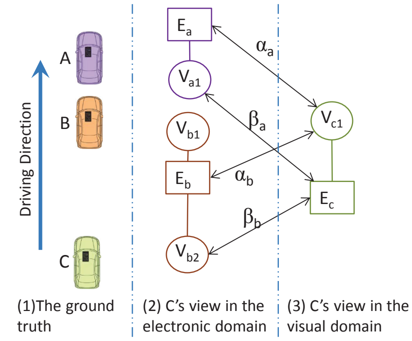

Fig. 1: An example showing how vehicle C distinguishes vehicle A and B using DoubleCheck. (1) shows the ground truth of the relative positions of the three vehicles. (2) shows vehicle C receives the EIDs of A and B, and their visual neighbors. (3) shows C's view in visual domain. The EIDs are shown in the square frames, and their corresponding visual neighbors are shown in the circles. The α_a, α_b, β_a and β_b shows the similarities between the vehicles connected by the arrows.

observe the opposing direction of $\theta(v_i)$, then we can let e_j check if it has a vehicle C' at the angle close to $\theta(v_i) + \pi$ in its camera. If there is no vehicle present at the angle of $\theta(v_i) + \pi$ in the camera of e_j, it probably means e_j is not a good match of v_i. In case that there is a such vehicle C', let us denote $S(e_j, v_i)$ by α, and $S(C, C')$ by β. If e_j is actually the EID of v_i, then both α and β should be high if the observed features are accurate. On the other hand, if e_j is not the EID of v_i, then both α and β should be low. So we redefine the similarity between e_j and v_i as the arithmetic mean of α and β. Figure 1 illustrates one sample use case of DC. In the figure, C derives the similarity between its visual neighbor V_{c1} and its electronic neighbor E_a using the mean of α_a and β_a, and derives the similarity between V_{c1} and E_b by the mean of α_b and β_b. Even if both α_a and α_b are high, β_b is more likely to be higher than β_a, because V_{a1} is actually the VID of vehicle B, and corresponds to E_b, rather than E_c. DC is more robust to the inaccuracies in similarity measurements, because it can reduce the possibility that the similarity of some correct matching becomes extremely low.

In practice, angle estimation errors could exist. We use the range $(-10°, 10°)$ on each camera to model such errors. ForeSight uses a loose threshold $(-20°, 20°)$ to define opposing angles, to tolerate angle estimation errors. Although using a loose threshold could potentially compromise the efficiency of the DC algorithm, such compromise is very limited due to the fact that the chance that two vehicles with similar visual features appearing in close angles of the same EID is very limited. Note that DC an be used along with any other similarity algorithm.

V. Experiments

Evaluating ForeSight with multiple cars using real-world driving requires multiple cars and devices, and the cooperation of the drivers. All such factors make the experiment extremely difficult to conduct. Instead, we performed a small scale

TABLE I: GPS accuracy in determining relative position

	Freeway	Downtown	Local
Front/Back	0.64	0.57	0.85
Six classes	0.24	0.25	0.20

driving experiment with three different cars in real-world traffic. The goal of the experiment is to show the accuracy of distinguishing the vehicles. One of the three cars is used as an observer car, and the other two are used as target cars. The target cars are denoted by T_1 and T_2. T_1 is a gold color corolla, and T_2 is a black Nissan Sentra. The observer car is always driving behind the target cars, while the two target vehicles randomly change their relative positions during the experiment. The observer car has a smartphone mounted at the center of the dashboard while the target cars have a smartphone mounted in the center of the rear window. The smartphones are used to record videos (at the default resolution of the smartphone) and GPS coordinates during the driving experiment. Recording video and doing offline processing (including comparison with ground truth) allowed us to quantify the accuracy of different algorithms. The driving path includes local streets, freeways and downtown areas. The driving time in each area is more than 10 minutes. Moderate traffic was observed on streets during the experiment.

1) GPS Accuracy: GPS is known to be unreliable in the downtown area. To evaluate the accuracy of GPS, we examine how GPS can be used to distinguish the relative position of the two target vehicles. If GPS cannot estimate the relative position correctly, then GPS alone cannot be used to distinguish the two target vehicles accurately. We first classify the position of T_1 relative to T_2 into six classes using lane-wise relative position (left, same, right) and front-back (front, back) relative position. The six classes are therefore, left-front, left-back, same-front, same-back, right-front and right-back. The ground truth relative position can be obtained by examining the videos after the driving experiment. We use the GPS positions at each time instance to determine the relative positions of the two target cars, and compare it with the ground truth. Table I lists the relative position estimation results using GPS. The results are only slightly better than random guessing.

2) Vehicle Matching with AW algorithm: In this step, we first obtain the visual features and electronic features of the two target vehicles, then use the AW algorithm to obtain the similarity between VIDs and EIDs, and match them using the bipartite graph matching algorithm. Two features are used in this experiment: the GPS position and the color. The visual features are extracted from the recorded video in the observer vehicle. We implemented a vehicle detection algorithm to detect vehicles in each image. The vehicle detection precision and recall of the software is 92.2% and 59.2%. In addition, the algorithm can find the dominant color and estimate the position of the detected vehicles relative to the camera. To find the dominant color of a vehicle, a clustering algorithm [7] is used to cluster the colors to find the largest color cluster, then the mean of the largest cluster is used as the color of the vehicle. The other vehicles present on the road add inaccuracies to the color feature measurements. We use the size of lane marks in the images as a reference size to estimate

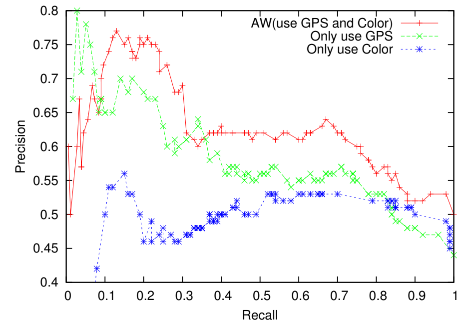

Fig. 2: The experiment result

the relative position of the detected vehicle. Since the vehicle detection software is not perfect, sometimes the target vehicles could be missing and some “ghost” vehicles may appear in the vehicle detection result. The “ghost” vehicles would be treated as vehicles without EIDs in ForeSight, and the target vehicle that is not detected is treated as a vehicle that has EID but does not have VID. On the other hand, the observer vehicle obtains the electronic features by directly querying the target vehicles.

In each matching instance, there are two EIDs which are the two target vehicles, while the number of VIDs depends on the vehicle detection result. To evaluate the matching result, we randomly selected 200 images from the video and manually labeled the correct matches. Next, we use a varying threshold to eliminate low similarity matching results. Different threshold value will lead to different recall rates. To make a fair comparison, we examine the matching accuracy at the same recall rate when comparing different algorithms. Figure 2 shows the precision of the vehicle matching result of ForeSight using AW algorithm, GPS and color. The presence of other vehicles on the road that have colors similar to our target vehicles, makes the matching result of “only use color” low when threshold is set to high value (when recall < 0.15 in Figure 2). Although, the errors in the vehicle detection and vehicle position estimation also affect the precision of AW matching, however, on an average, for a given value of recall, precision of AW was 4.7% and 16.3% higher than that of “Only use GPS” and “Only use color”, respectively. This demonstrates that by combining multiple features, ForeSight is able to improve the accuracy of vehicle matching.

VI. Simulations

To evaluate the performance of Foresight in large vehicular networks, we performed realistic simulations with SUMO [8] and NS-3 [9]. SUMO is an open source, continuous road map and traffic simulator to create road network, while NS-3 is a network simulator used to simulate communication between vehicles (smartphones) and evaluate the algorithms. To make the simulation more realistic, data from our measurement study as well as other literature was used as the input, to set up parameters of error models, wireless interference, packet losses, etc. We err on the side of being conservative, and use values of the worst cases to configure these parameters. Therefore, the simulation result would be a conservative estimation of real world implementation. Next, we will present how the feature error models are generated in our simulation.

A. GPS

In our simulator, a vehicle will generate its own GPS position at different time instances. In our simulations, we use the GPS error model from the official Performance Standards & Specifications [10]. In this model, the GPS error has 4 meters standard deviation. Observe that this error model approximately matches with the other independently performed measurements [3]. When a vehicle A observes vehicle B, A estimates the GPS position of B by using its own GPS measurement, and incorporating the distance and angle estimation using A's Camera. As a result, a model is required to generate such measurement errors. To model the measurement errors in distance and angle estimation, we use the results from [11], which proposed a method that measures distances using smartphone. According to their measurement, the error in distance detection is at most 12%. Thus, in the error model of our simulation, the estimated distance was generated randomly within 12% of the ground truth distance. We did not find existing works that can provide the angle detection error model, so we assume the angle estimation error is uniformly distributed in range $(-10^\circ, 10^\circ)$.

B. Vehicle color

In the simulator, the colors of vehicles are generated according to the automotive color popularity in the market. Specifically, we use the color distributions provided by the color popularity report from DuPont [12] in 2011. According to the report, in North America white was ranked first (23%) and black second (18%), followed by silver (16%), gray (13%), red (10%), blue (9%), brown (5%), yellow/gold (3%), green (2%). These colors covered 99% of the vehicles. Note that, in reality, there could be many shades of the same color, which could potentially be used to further distinguish the vehicles. However, since the detailed popularity distribution of colors is not available, we do not further classify vehicles of the same color category.

To quantify the similarity of two color measurements, we use the CIELAB color space [13] to represent colors. In CIELAB, each color is represented by a point in a three dimensional space, where L quantifies the lightness, A quantifies the balance between magenta and green, and B quantifies the balance between yellow and blue. The advantage of using CIELAB, over the commonly used RGB and HSV color space, is that the relative perceptual difference between any two colors in CIELAB can be approximated by simply measuring Euclidean distance between them. The errors in color measurements can be caused by weather condition, light, camera configuration, etc. To measure the variation in color estimation with distance, we video-taped vehicles in a parking lot on different days with three smartphones (Lumia 900 with Windows Phone 7 OS, iPhone 5 with iOS 6.1, and Samsung Galaxy S3 with Android 4.1.1). We take one snapshot image of the vehicle body for every 1 meter and converted the color encoding to CIELAB color space. Since there are variations of the color at different parts of the vehicle body because of decorations, shadows, reflections, dirt, etc., we needed to find the dominant color of the vehicle from each image. The dominant color of a car is found using the same clustering algorithm as in Section V. Figure 3 shows a set of measurement results of nine vehicles with distance varying from 5 meters to 80 meters (using the Lumia 900 phone). Measurements from iPhone and Galaxy S3 show similar results. In the simulation, the errors in color estimation were generated using the distribution shown in Fig. 3.

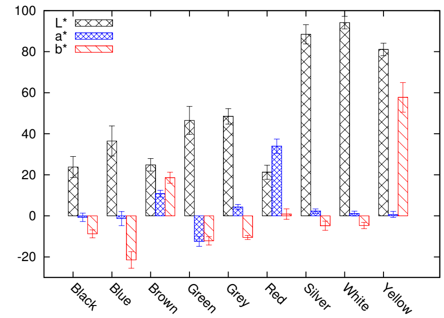

Fig. 3: The mean and standard deviation of the measured vehicle colors in CIELAB space. The distribution is obtained by measuring the color of nine vehicles at distances varying from 5 meters to 80 meters.

C. Simulation Description

We first use SUMO to generate a road map with its shape similar to the letter 'H' in the simulator. The total length of the road is 9 kilometers. Each direction of a road has 3 or 4 lanes, and the speed limit is 50km/h. Overall, 1440 vehicles entered the road map in the 15 minutes simulation period. SUMO creates trace files that contain the position of each vehicle at each time instant (every second). These trace files are fed as input to NS-3 simulator to simulate the position of corresponding smartphones and their communication in NS-3.

To evaluate the performance of ForeSight at different deployment rates, at the beginning of the simulation, only a random subset of vehicles are installed with ForeSight. A vehicle running ForeSight has a GPS device, a WiFi radio, and one or two cameras. The angle of view of each camera is set to 120°. Cameras detect 95% of the vehicles in their field of view that are within 100 meters. If a vehicle has one camera, then that camera is installed in front of the vehicle facing the driving direction of the vehicle. To simulate the visual blockage of the real world, we modeled the geometrical shape of each vehicle. Further, a vehicle can only see vehicles in front of the camera that have at least one side of the exterior visible from this vehicle. If a vehicle has two cameras, the second camera will be installed at the rear of the vehicle facing the back. The radio is used to let each vehicle communicate with neighboring vehicles using the 802.11 ad hoc mode. The maximum communication range of the radio is 150 meters.

After generating the driving traces of the vehicles in SUMO, we replay the driving traces in NS-3. For each vehicle in NS-3, the electronic features of itself and the visual features of the vehicles in its cameras are created based on the error models described earlier. Each vehicle executes the matching

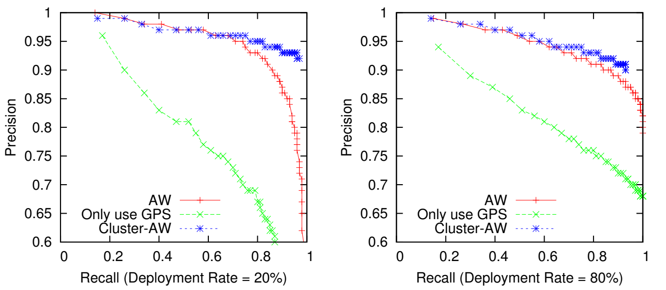

Fig. 4: The performance of GPS and ForeSight when using AW and Cluster-AW.

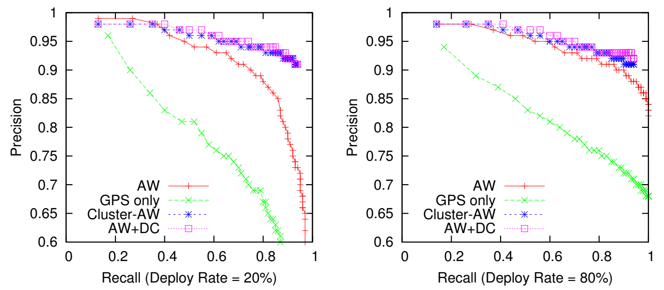

Fig. 5: The performance of ForeSight with AW, Cluster-AW and AW+DC when two cameras are used in each vehicle.

algorithm once every 5 seconds. At the beginning of each execution, it sends a broadcast message to broadcast its ID and its electronic features. In the next step, we evaluate our algorithms based on the data collected on vehicles that have ForeSight installed.

D. Evaluate AW

Figure 4 shows the simulation results of AW and Cluster-AW using GPS and color as two features. The Cluster-AW algorithm is implemented with [7]. We choose Pearson correlation coefficient provided by [7] to estimate the distance between two measurements. A threshold in range $(0, 1)$ is used to eliminate the matches with low similarity value. When apply different threshold, we obtain different values corresponds to different recalls and precisions. Figure 4 shows that AW has significantly improved the matching precision for different recall rates. When the deployment rate is higher, the matching precision of ForeSight is also higher. Note that even in 20% deployment rate, the Cluster-AW algorithm precision is more than 90% when the recall rate is high ($>$ 90%). We simulated the matching performance when using the Mahalanobis distance, PCA and the cosine similarity to estimate the similarity between vehicles in ForeSight. However, their performances are all below the matching accuracy of GPS.

E. Evaluate DC

In this section, we evaluate the scenario that each vehicle is deployed with one front-view camera and one rear-view camera. To evalute the DC algorithm, we first apply the AW algorithm and then apply the DC algorithm (AW+DC) to obtain the matching result. Compared with the results in Figure 4, Figure 5 shows that there is no obvious difference when the vehicle uses two camera than one camera. AW+DC shows a maximum of 2% improvement over Cluster-AW, which is still considerable as the improvement space is very limited.

VII. Case Studies: Applications

We study two applications of ForeSight in this section: The first demonstrates how vehicles in ForeSight can cooperate to improve the GPS accuracy of each other; The second demonstrates how ForeSight reduces the number of messages delivered to unintended recipients.

A. Improving Localization Accuracy using ForeSight

As discussed in Section VI, GPS accuracy might be affected by multiple factors. Some of those errors such as signal multipath error and device clock error, however, can be mitigated by taking multiple measurements from multiple devices. We run a simulation to study how much GPS accuracy can be improved by using ForeSight. In this simulation, we assume all vehicles have deployed ForeSight with the Cluster-AW algorithm. Under this deployment rate, the matching recall is set to 95%, and the corresponding precision of ForeSight is 88%. Each vehicle performs a matching every 5 seconds. After the matching is completed, it sends a packet to each vehicle in its matching result, which consists of its estimation of the target vehicle's GPS coordinates. Therefore, one vehicle could send different measurements to multiple vehicles, and one vehicle could receive different measurements from multiple vehicles. Estimations of the same vehicle by different observing vehicles can be combined together to improve the accuracy of GPS localization.

Suppose at time t, vehicle C received M estimates of its own GPS coordinate from neighboring vehicles. So, C now has a total of $M + 1$ estimated coordinates (including its own GPS reading). Some measurements could have large errors if the sender did not perform correct matching. For simplicity, C updates its coordinates with the mean of those $M + 1$ coordinates. Let d_1 denote C's original GPS error without assistance, and d_2 denote C's current GPS error with assistance. For large values of M, we expect d_2 to be smaller than d_1, and this improvement is dependent upon the number of estimated coordinates available and the error distribution of those estimates. Note that although there will be some estimation error, we believe they can also be mitigated by taking the mean of measurements from multiple vehicles. When the simulation is finished, we classify all the cases (all vehicles, at each second) based on the number of estimates received (See Figure 6). In the figure, the ForeSight error rate represents the probability that an estimate has large error due to incorrect matching (this is simply the complement of precision). We found that ForeSight can reduce the GPS error by as much as 40% when three estimates from neighboring vehicles are available. The error rate decreases when the number of estimates available increase from 0 to 4. However, the improvement is compromised by the increasing ForeSight error rate beyond 4 estimates. One interesting observation is that when a vehicle only uses its own GPS to estimate its

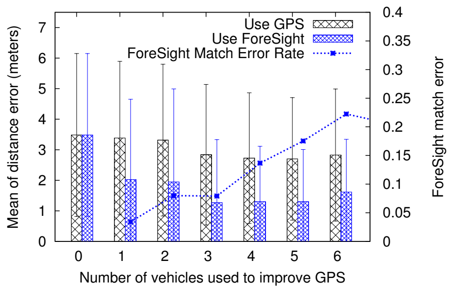

Fig. 6: The mean and standard deviation of the GPS estimation error.

coordinates, high number of assisting vehicles indicates that this vehicle has low GPS error rate. This implies that for any specific vehicle, although all its measurements follow the GPS error model, if it has low error in a GPS measurement, it will be more likely to be matched with its EID by more vehicles. Another observation is that ForeSight error rate tends to increase when the number of available estimates increase. This is because higher number of estimates usually imply denser traffic, which means vehicles are less distinguishable, resulting in lower precision of matching.

B. Reducing Degree of Electronic Disturbance with ForeSight

In this application, each vehicle on the road sends an alert message to the vehicle in front whose speed is lower than this vehicle to notify a potential overtaking event. We study the degree to which vehicles on the road are disturbed (i.e., how many unintended vehicles receive this alert message) by these messages sent to specific targets. The methods include using broadcast, using GPS, and using ForeSight. In the broadcast approach, messages are received by all nearby vehicles within the communication range of it. When using GPS, messages are sent to some dedicated GPS coordinates based on the host vehicle's estimation. Due to inaccuracy in GPS and ForeSight, they have to notify vehicles other than the target vehicle to achieve a given recall rate. Here we use 95% recall rate. In our GPS error model with 4 meter standard deviation, the simulation shows that vehicles within 8 meters of those GPS coordinates will receive the alert messages to achieve 95% of recall. ForeSight will send messages to a set of EIDs that may possibly match the target with 95% of recall to ensure the target vehicle receives the message. We use a moderate traffic profile of 100 seconds on the same map. In the simulation, there are a total number of 1021 instances of vehicles that need to be notified. Broadcast disturbed a total number of 29823 vehicle instances, which is 29 times more than the target vehicle instances. The method of using GPS disturbed 1685 vehicles instances. ForeSight only disturbed 120 vehicle instances. Overall, ForeSight reduces the number of unintended receivers by a factor of 14 when compared to GPS-based approach. Observe that due to the broadcast nature of wireless communication, same number of radios would receive the message in all the three methods. However, when using broadcast, the radios receiving the message will deliver it to higher layers resulting in unintended alerts to the driver. On the other hand, the radio (or the Foresight application) in the GPS scheme and the ForeSight system can simply discard the message if it is not the intended receiver. Compared with the broadcast method, in the GPS based method and ForeSight, vehicles have an extra overhead of periodic broadcasting their IDs and GPS coordinates. Further, in ForeSight, vehicles also broadcast their other features periodically. Since a feature can be encoded in a few bytes, the communication overhead is expected to be small.

VIII. Related Work

The most related problem is the multiple target tracking problem [14, 15], which involves two sub-problems are studied: i) Data association problem where the objective is to find a mapping between the measurements at different times; and, ii) Estimation problem, where the objective is to estimate the state of the targets. Authors in [16] provided a particle filter based method that statistically estimates the state of the targets, and uses parameter estimation to obtain the data associations. However, this solution cannot quickly adapt to changing environments. [17] reviewed several statistical data association techniques. The first step is to do sensor value normalization [6], in order to estimate the similarity between elements in different data sets. If the association decisions can be postponed, then there are two advanced algorithms: joint-probabilistic and multiple-hypothesis algorithms [6, 17], which can be applied to improve the association accuracy. The joint-probabilistic algorithm requires the number of targets. The multiple-hypothesis algorithm cannot meet the real-time constraints in vehicular networks. In this work, we provided algorithms to derive a matching between the VIDs and EIDs in a single time instance.

There are existing works [18, 19] on matching vehicles that appear in non-overlapping camera sites. Authors in these works provide algorithms for matching two vehicles in visual domain. These works provide methods to extract visual features of vehicles from cameras and track vehicles. Our work differs from aforementioned works in that we need to associate information in electronic domains in the matching.

There are methods in machine learning, data mining and data fusion areas that can combine multiple features to derive a single similarity value. For example, Mahalanobis distance and cosine similarity can be used to estimate the distance or similarity of two vectors. PCA (principal component analysis) is an unsupervised distance metric learning [20] method that can be used to reduce the number of dimensions of the features. These methods have been shown to perform poorly for our problem (see Section III-A).

Recent research works[21, 22] have focused on matching information in different domains. These works are for target tracking using stationary deployed cameras. On the other hand, vehicles in ForeSight cooperate with their dynamically changing set of neighboring vehicles. In addition, there are recent research works[3, 23] focusing on improving the localization of vehicles by using other features besides GPS. The objective of these works are orthogonal to ForeSight and can be used in ForeSight to add more features or improve the accuracy of existing features.

IX. Discussion and Future work

In this paper, we have shown how to exploit multiple features and multiple cameras to improve the matching precision between visual neighbors and electronic neighbors. GPS and vehicle color were carefully modeled to be used as features in the simulations and experiments. The presented AW algorithm exploits the relative distinguishability of each feature to determine the similarity between different vehicles. Our simulation shows that using multiple features yields better matching precision at different recall rates.

As a low-cost solution, it is true that the matching precision provided in ForeSight is not high enough for safety critical applications, however it can be used in many other types of vehicular applications, such as driving advisory applications (e.g., overtaking alert), cooperative driving applications (e.g., platooned vehicles). Drivers deployed with the system can benefit from sending and receiving notifications to interact with neighboring vehicles. Given the design of ForeSight, the matching precision is expected to improve progressively when high-accuracy devices such as LIDAR and stereo-cameras are used. In addition, ForeSight currently only uses the information available in a single time-slot. It is possible to improve the accuracy of a single feature by combining history information. ForeSight can benefit by exploiting such a priori knowledge.

There are several interesting questions remaining in this work. Instead of using bipartite graph matching model to match the vehicles, we can use a structure matching model. For a vehicle C, the features of $N_v(C)$ and $N_e(C)$ can be taken as points in a multi-dimensional feature space. So, $N_v(C)$ could form a relational structure R_v based on their relative positions in the feature space. Correspondingly, $N_e(C)$ could form another relational structure R_e. The problem of matching $N_v(C)$ and $N_e(C)$ can be transformed into finding the weighted maximum common parts of R_v and R_e. The relative structure matching problem is widely used in computer vision research, and the general solution is to convert the problem to the problem of finding the maximum clique in an association graph [24]. However, how to properly create the association graph is problem specific.

After the matching, neighboring vehicles can exchange their matching results and discover conflicts. Besides the DoubleCheck method, there are some other methods that can be used to discover hidden conflicts within different matching results. For example, if vehicle C finds A and B are neighbors in its matching result, C can check with A and B, to see if their matching results agree with C's. If disagreement exists, a possible conflict is found. The matching precision could be further improved if such conflicts are properly resolved. The We will focus on conflict resolution with the aforementioned structure model.

References

[1] N. Wisitpongphan and et al., "Broadcast Storm Mitigation Techniques in Vehicular Ad Hoc Networks," *IEEE Wireless Communications*, vol. 14, no. 6, pp. 84–94, 2007.

[2] X. Yang, J. Liu, N. Vaidya, and F. Zhao, "A vehicle-to-vehicle communication protocol for cooperative collision warning," in *Mobile and Ubiquitous Systems: Networking and Services*, August 2004, pp. 114–123.

[3] D. Li, T. Bansal, Z. Lu, and P. Sinha, "MARVEL: Multiple Antenna based Relative Vehicle Localizer," in *Proc. of ACM MOBICOM*, 2012.

[4] "Google discloses costs of its driverless car tests," http://content.usatoday.com/communities/driveon/post/2012/06/google-discloses-costs-of-its-driverless-car-tests/1#.UfA7_BchCHe, 2012.

[5] H. Moustafa and Y. Zhang, *Vehicular Networks: Techniques, Standards, and Applications*. Auerbach publications, 2009.

[6] H. B. Mitchell, *Multi-sensor data fusion: an introduction*. Springer, 2007.

[7] M. J. de Hoon, S. Imoto, J. Nolan, and S. Miyano, "Open source clustering software," *Bioinformatics*, vol. 20, no. 9, pp. 1453–1454, 2004.

[8] Simulation of Urban Mobility, http://sumo.sourceforge.net.

[9] The NS-3 Network Simulator, http://www.nsnam.org.

[10] GPS.gov, "Global Positioning System Standard Positioning Service Performance Standard," 2008.

[11] Q. Wang, A. Lobzhanidze, S. D. Roy, W. Zeng, and Y. Shang, "Positionit: An Image-based Remote Target Localization System on Smartphones," in *Proc. of ACM Multimedia*, 2011.

[12] DuPont Media Center, "2011 Automotive Color Popularity Report," http://www2.dupont.com/Media_Center/en_US/color_popularity/index.html, 2011.

[13] J. Schanda, *Colorimetry: Understanding the CIE system*. Wiley-interscience, 2007.

[14] S. Kim and et al., "Front and rear vehicle detection and tracking in the day and night times using vision and sonar sensor fusion," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.

[15] R. Mobus and U. Kolbe, "Multi-target multi-object tracking, sensor fusion of radar and infrared," in *Proc. of IEEE Intelligent Vehicles Symposium*, 2004.

[16] C. Hue, J.-P. Le Cadre, and P. Pérez, "Sequential monte carlo methods for multiple target tracking and data fusion," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 309–325, 2002.

[17] I. J. Cox, "A review of statistical data association techniques for motion correspondence," *International Journal of Computer Vision*, vol. 10, no. 1, pp. 53–66, 1993.

[18] T. Huang and S. Russell, "Object identification in a Bayesian context," in *International Joint Conference on Artificial Intelligence*, 1997.

[19] K. She, G. Bebis, H. Gu, and R. Miller, "Vehicle tracking using on-line fusion of color and shape features," in *Proc. of Intelligent Transportation Systems*. IEEE, 2004.

[20] L. Yang, "An overview of distance metric learning," in *Proc. of the Computer Vision and Pattern Recognition Conference*, 2007.

[21] J. Teng, J. Zhu, B. Zhang, D. Xuan, and Y. F. Zheng, "EV: Efficient visual surveillance with electronic footprints," in *Proc. of IEEE INFOCOM*, 2013.

[22] B. Zhang, J. Teng, J. Zhu, X. Li, D. Xuan, and Y. F. Zheng, "EV-Loc: Integrating electronic and visual signals for accurate localization," in *Proc. of ACM MOBIHOC*, 2012.

[23] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments," in *Proc. of the Robotics: Science and Systems Conference*, 2007.

[24] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo, "The maximum clique problem," in *Handbook of combinatorial optimization*. Springer, 1999, pp. 1–74.