Collaborative Navigation Systems for Collision Avoidance

Hassan A. Karimi, Benjamin Zimmerman, David Nawn, Peter Sutovsky
Geoinformatics Laboratory, School of Information Sciences
University of Pittsburgh

***Abstract*-- Accidents on roads are one of the main causes of human loss and damages to property every day and anywhere that vehicles travel. Communications-based approaches to reducing highway accidents include vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications. However, while V2V and V2I are promising approaches for reducing accidents, they require installation of new sensors in vehicles and the information they can produce is limited. In this paper, we present the navigation-to-navigation (Nav2Nav) approach for reducing accidents on roads which is different from V2V and V2I from several perspectives including: (a) the widespread availability and use of navigation systems with no installation requirement, (b) the familiarity and acceptance of navigation systems by many people around the word, (c) the availability of map databases in navigation systems which facilitates map matching (the process of finding the correct segment of road as opposed to raw positioning data) and prediction in navigation, and (d) the availability of routes requested by drivers.**

***Index Terms*—Navigation, Satellite navigation systems, Mobile communication, Distributed decision-making**

I. INTRODUCTION

Much research has been conducted under the umbrella of Intelligent Transportation Systems (ITS) on crash avoidance, traffic monitoring, and autonomous driving. Most research projects focus on how information should be transferred within the dynamic network of vehicles on the road. Such research projects have led to the emergence of two major paradigms, vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I). In V2V, information such as the vehicle's location and speed is transferred between vehicles directly, over a wireless medium, typically by adding additional sensors to the vehicle that are integrated into an on-board system. Additionally, the wireless communication protocol must ensure information security and driver privacy. In V2I, information is transferred between vehicles to roadside infrastructure that then either uses that information to perform calculations and communicate back to the sending vehicle or forwards it on to other vehicles for further processing. Research also has focused on employing a hybrid approach by integrating V2V and V2I into one system. One major difference between V2I and V2V is that V2I is mainly developed, maintained, and operated by local or state government agencies, whereas V2V is integrated into a vehicle's on-board system which means it is developed by vehicle manufacturers and operated by end-users.

Collision avoidance is a main focus for both paradigms. A collision situation arises primarily because a driver is not able to perceive and react to the traffic reality on time. This is mainly due to difficulty in getting an instantaneous perception of the ever-changing traffic environment. Inaccurate perception can happen because of factors like angularity of the road, poor weather conditions, high speeds, or poor concentration.

In this article, we describe the concept of a navigation-to-navigation (Nav2Nav) paradigm for crash avoidance at intersections. Focus is placed on intersection collisions because they account for 36 percent of all traffic accidents, according to the National Highway Traffic Safety Administration's 2008 Report [1]. Nav2Nav will rely on information typically available in personal navigation devices (PNDs) without being integrated into the vehicle's monitoring systems or added as new sensors to the vehicle's infrastructure. Furthermore, no roadside devices are required to operate Nav2Nav. The navigation systems will exchange information with each other, and perform all the calculations for decision-making themselves. Communications between vehicles could take place using a wireless protocol, such as 802.11p[1]. This would permit Nav2Nav to provide useful information to the driver in sufficient time to ensure the security of the information that is shared by the vehicles and to protect the privacy of the drivers operating the vehicles. Nav2Nav relies on information typically available to navigation systems.

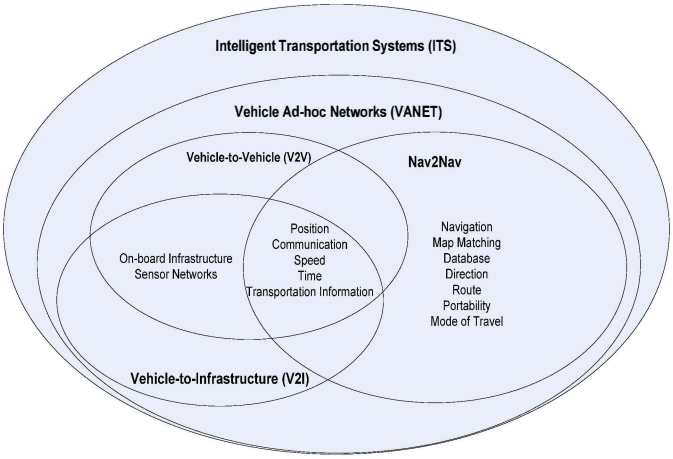

Fig. 1. Relationships between Nav2Nav, V2V, and V2I.

Figure 1 shows the relationship between ITS, Vehicle Ad-hoc Networks (VANETs), V2V, and Nav2Nav, and V2I. In particular, the figure highlights the differences between Nav2Nav and V2V. While somewhat similar to V2V, Nav2Nav: (a) does not require augmentation to current vehicle

[1] http://grouper.ieee.org/groups/802/11/Reports/tgp_update.htm

COLLABORATECOM 2010, October 9-12, Chicago, USA

DOI 10.4108/icst.collaboratecom.2010.21

on-board systems, (b) could be portable between different vehicles, and (c) could provide entire routes requested by drivers (as opposed to only directions at intersections). Since Nav2Nav would operate without any support from the vehicle's monitoring system, Nav2Nav is not confined to use only in automobiles. Nav2Nav can just as easily be used by bicyclists on the roads. This permits Nav2Nav to be portable; for example, a unit can be changed from car-to-car to car-to-bike by the same owner. Map matching, a technique for "snapping" a vehicle's position to a road segment [12, 13], will be employed in Nav2Nav to assist with high location accuracies. As all navigation units rely on a road-network database to conduct map matching, the underlying database that will provide information to the unit must be of high quality. The units also provide navigation capabilities to the driver; the attributes of origin, destination, intended route, and direction at the current intersection can be leveraged for additional information within Nav2Nav. Nav2Nav also takes into account the potential of the integrated Global Navigation Satellite System (iGNSS). Nav2Nav is expected to take advantage of the planned iGNSS which in addition to U.S. GPS includes the EU's Galileo, Russia's GLONASS, and China's Compass [14].

One of the challenges in Nav2Nav is to acquire precise position information since almost all subsequent activities primarily rely on the position of the vehicle and nearby vehicles. Although hybrid sensors may improve the quality of a vehicle's estimated position [2-5]), reliance on iGNSS as the only positioning technology, mainly due to its global coverage, needs further research.

With a single GNSS, i.e., the US GPS, 4-10 meter accuracy in open-sky areas is achievable [6-7]. However, problems usually occur when the GNSS receiver operates in obstructed areas, such as in urban canyons and under foliage, where large signal attenuation and degradation can worsen the receiver's performance. Moreover, in deep urban canyons, the receiver may observe such an inadequate number of visible satellites that no position estimation can be acquired. However, iGNSS is expected to overcome such problems.

We believe that Nav2Nav transforms ITS to the next level of its evolution by integrating information already available in navigation systems and familiar tools for most drivers to assist in reducing collisions. In other words, Nav2Nav is a paradigm shift from "personal navigation" to "collaborative safety decision-making".

II. Nav2Nav Model

A. Interaction

Many crash situations are possible. Figure 2 illustrates two potential crash situations. As Vehicle B (V_B) intends to travel straight through the intersection, Vehicle A (V_A) has two potential crash situations, when V_A intends to go straight through the intersection or when turning left at the intersection. Nav2Nav in each vehicle would know their respective vehicle's intended routes. The units could then communicate their intended turning information to the other vehicle which would allow a decision making process to take place to determine which, if any, vehicles pose a potential crash situation. An alert would be displayed to the driver as to which vehicles to be cautious of and display a message if an action is necessary to avoid a collision, such as slowing down or stopping.

To communicate effectively and provide safety related information to a driver, all potential vehicle pathways at intersections are considered. Potential collision pathways of two vehicles at an intersection are analyzed in this section. Once the parameters of a simple (4-way) intersection are understood, additional research will be conducted on more complex situations involving multiple junctions and more than two vehicles. There are two main possibilities for analyzing locations of vehicles (configuration) with respect to each other at an intersection: vehicles on the opposite side of intersection (parallel configuration) and vehicles positioned on an orthogonal side of the intersection (orthogonal configuration).

In our analysis, a total of 54 possible situations of vehicle locations at an intersection were discovered. This number is derived from the equation below where *NmbrVhlLoc* is the number of different possibilities and *VA headChoice* and *VB headChoice* are the number of heading options for each vehicle. In case of a 4-way intersection, there are three heading options for each vehicle. There are 6 such location configurations (*nmbrVhlLoc*) at a 4-way intersection: 2 parallel configurations and 4 left-right configurations (Figure 1). Each of these location configurations can lead to nine situations (*VA headChoice* VB headChoice*). The total number of possible situations is given by:

$$NmbrVhlLoc * VA\ headChoice * VB\ headChoice = TotalVhlSituations,$$

where in our case *NmbrVhlLoc = 6, VA headChoice = 3, and VB headChoice = 3*. Table 1 summarizes our findings.

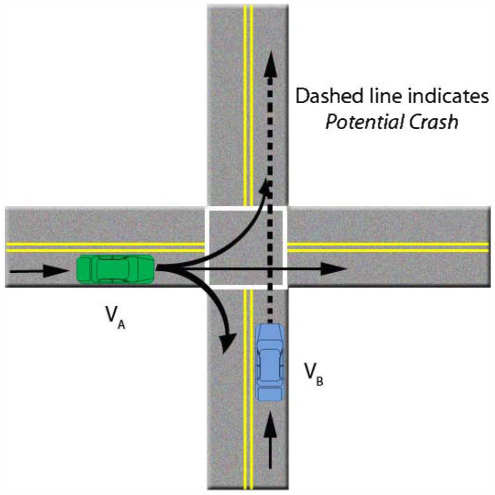

Fig 2. Potential crash situations

TABLE I
NUMBER OF POTENTIAL COLLISION PATHS AND CAR POSITIONS IN CASE OF 4-WAY INTERSECTION AT AN INTERSECTION.

Relative position of cars at the intersection		
	Orthogonal	Parallel
Number of collision paths in 4-way intersection	5	5
Number of car positions at 4-way intersection	4	2

B. Sub-Zoning

As shown in Figure 3, the first step in the Nav2Nav model is for a vehicle approaching an intersection to start communicating with other vehicles approaching the same intersection. We consider three zones for the Nav2Nav model: the *broadcast zone*, the zone in which the vehicle needs to communicate with other vehicles; the *computation and decision-making zone*, the zone in which the Nav2Nav model makes decisions about whether to warn a driver; and the *actuation zone*, the zone in which the decision must be presented to the driver. In addition to common navigation information, in Nav2Nav, a map database containing information about the geometry and topology of intersections and speed limits on each road segment is also taken into account.

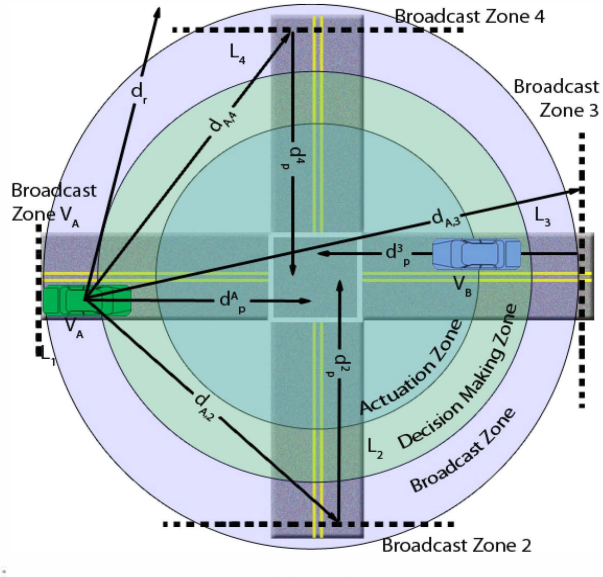

Fig. 3. The broadcast zone

Broadcast Zone: Once a vehicle is within the broadcast zone, it will communicate with other vehicles within its range to send and receive information about their directions including information on position (X, Y), speed, road segment identification, Θ (direction of movement with respect to the intersection), and time. If $V_i(\Theta_i)$ and $V_j(\Theta_j)$ are predicted to pass each other at the intersection and, based on their speed and directions, it is determined that they have the potential to crash, Nav2Nav will request additional information from the other vehicle's Nav2Nav, such as an uncertainty value with respect to taking computed routes by that driver.

Computation and Decision Making Zone: Once the vehicle is within the computation and decision-making zone, the potential collision pattern will be analyzed and an appropriate warning message will be decided upon and provided to the driver.

Actuation Zone: In this zone, the appropriate message to the driver must be prepared and communicated to the driver. Some reaction time should be reserved for the driver to respond to the warning information.

Initiating Broadcast Zone

Along each road segment, the broadcast zone is defined by a distance d_b from the intersection. This distance can be determined by using static information (e.g., geometry and topology of the intersection and the speed limit on the road segment) or dynamic information (e.g., current speed of the car and its location). The broadcast zone should be: (a) far enough from the intersection to provide sufficient time to exchange information (t_b), to compute a decision (t_c), and to provide drivers with sufficient reaction time (t_a) and (b) within communication range d_r of other vehicles that are likely to meet at the intersection.

The minimum distance, d_p, from the intersection includes the traveling distance of the vehicle during the broadcast time t_b, computation and decision-making time t_c, and the driver's response time d_r. Using the speed of the vehicle, d_p can be calculated by: $d_p = s * t_p$, where $t_p = t_b + t_c + t_a$ is the total processing time.

However, to communicate, two vehicles must be within the communication range d. Assume V_A is approaching the intersection on the road segment 1 with the speed s_A (Figure 2). Assuming all vehicles have the same processing time:

$$t_p = t^A_p = t^j_p$$

and the same communication range: $d_r = d^A_r = d^J_r$

where i=2,3,4 is index of the vehicles. With the speed limit s^i_L on the all road segments that intersect at the intersection (i=1..4), processing distance d^1_p of vehicle V_A from the intersection can be computed. First, processing distance can be computed as: $d^A_p = s_A * t_p$

Then, the processing distance from the intersection for other road segments can be computed as $d^i_p = s^i_L * t_p$, where i=2,3,4. Assuming that a vehicle's speed is the posted speed limit on the road segment (one could easily assume some margin above the posted limit as well), the vehicles that are on the road segment i and are further from the intersection than d^i_p will not arrive at the intersection at the same time to meet the vehicle at the intersection. This information, along with the geometry of the intersection, will be used to compute distance, $d_{A,i}$, for each vehicle V_A within computation and decision-making zone on road segment i. Vehicle V_A can communicate with relevant vehicles on road segment i if their distance $d_{A,i}$ is shorter than the communication range. The factor $d_{A,i}$ is a function of geometry of the intersection and the distances d^A_p and d_p.

Using d_r, d^A_p, d_p, and $d^{A,i}_p = f(\text{geometry}, d^A_p, d^i_p)$ where i=2..3, the broadcast zone can be computed as follows:

$$d^A_b = \begin{cases} d^A_p & \text{If } d_r > \min d_{A,i}\left(d^A_p, d^i_p\right) \\ d_r & \text{If } d_r < \min d_{A,i}\left(d^A_p, d^i_p\right). \end{cases}$$

For example, in the situation represented in Figure 3 $d^A_b = d^A_p$. Since $d^3_p > d_r$ vehicle V_A might not be able to communicate with some of the vehicles on the road segment L_3 that will arrive at the intersection at the same time as vehicle V_A.

III. NAV2NAV SIMULATION

The Nav2Nav model and protocol is currently being simulated in the Geoinformatics Laboratory at the university of Pittsburgh to identify various considerations about the decision-making process and protocol requirements. A simulation also allows a degree of control that could not be achieved otherwise due to little cost of implementation as opposed to engineering actual units. Aspects of the simulation can be modified to test the abilities of Nav2Nav. Traffic volume and the allotted Nav2Nav process time are just two variables that can be adjusted in a simulation environment. The expectation is that the system will be able to accurately predict the possibility of a crash based on the information coming from other cars' Nav2Nav units.

Design and implementation of a new simulation environment would be time consuming, so existing simulation systems that deemed relevant were considered first. Microscopic Traffic Simulation Model (CORSIM) [8], VisSim [9], and TransModeler [10] were examined. This allowed identification and examination of simulation aspects that might be important to develop in the Nav2Nav simulation environment. It was realized that an agent-based simulation would be most suitable for the Nav2Nav model. In the environments examined, a traffic controller fed cars into the environment and assured safe traffic flow, as their aim was more at discovery of network flow problems. Our focus is on realizing driver behavior to provide safer intersection crossings, so driver agents are required for a simulation to be meaningful.

The list of objects that we deem appropriate (see Table 2) for the simulation are: the Nav2Nav units, the cars, intersections, drivers, a large road network with many intersections, and iGNSS quality of service (iGNSS QoS). Each object will hold the information that it would use in a real-life implementation. The Nav2Nav unit will possess such information as the car's position, speed, bearing, and route. This information can then be broadcast to Nav2Nav units within other cars approaching the intersection so they can analyze the information and decide if a crash is imminent, and if so, can warn the driver. The car objects will hold the Nav2Nav unit object and the driver object. Driver objects will maintain a model of the driver. The goal is to be able to model drivers accurately with characteristics such as aggressive and passive, which would have a major impact on how they approach intersections when other cars are around. Aggressive drivers might speed up to try to make the light, while passive drivers may approach the intersection with more caution. While the driver and Nav2Nav objects are a part of the car object, each will still have different affects on the simulation. The Nav2Nav unit will be responsible for alerting the driver of an imminent collision and the driver will decide what to do with that information. The intersection objects will maintain the list of adjacent intersections, the type of intersection, and all of the objects on the intersection. An additional entity that will be modeled is iGNSS QoS to model future geopositioning technologies.

TABLE II
SIMULATION ENTITIES, PARAMETERS, AND PURPOSE

Entity To Model	Parameters	Purpose
Nav2Nav Unit	Position, Speed, Route, Time, Orientation	To develop the decision making processes, machine-learning techniques, communication protocols and other functions of Nav2Nav units.
Individual Car with Nav2Nav unit	Position, Speed, Route, Time, Orientation, Deviation of Driver from Navigation Instructions, Vehicle Type	To model driver behavior when using a Nav2Nav unit, to test accuracy of Nav2Nav unit
Traffic Volume	Number of Cars on Road Network, Traffic Rate	To test that the communication and decision making developed in each car will allow for wireless communication and provide enough time for decision-making
Individual Intersection	Position, Number of Cross Streets, Geometry of Intersection	To represent one instance of the collision avoidance problem
Road Network with Many Intersections	Topology	To represent the traffic flow between intersections, how the collision avoidance problem can affect other intersections, road conditions
iGNSS QoS	Availability, Accuracy, Reliability	To test the Nav2Nav unit's ability to conduct dynamic prediction of GNSS QoS

IV. SIMULATION ENVIRONMENT

The current Nav2Nav simulation environment is comprised of the model presented above one 4-way intersection. The Nav2Nav process is simulated for a set of cars on this one intersection. The entire simulation environment is developed as a multi-agent based model of how Nav2Nav would operate. The model was built similar to the work done in [11]. Several entities are simulated in this phase including cars, drivers, and Nav2Nav units. Each object instance is its own thread, allowing it to behave in real-time and autonomously from the overall simulation controller object. Java was used as the programming language mostly due to its multi-threading capabilities. The classes and interfaces used in the environment are shown in Figure 4.

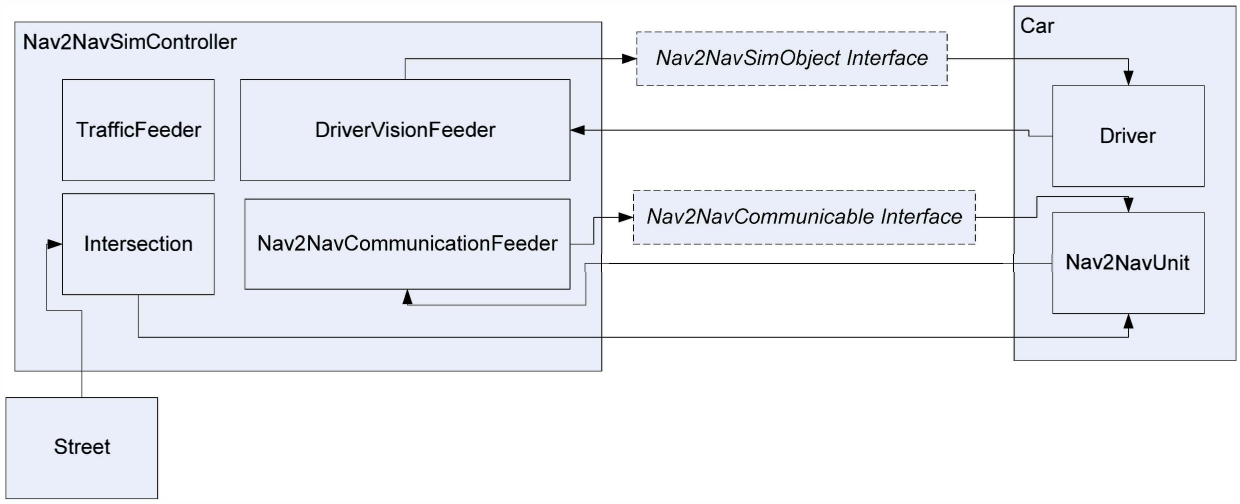

Fig. 4. Classes and Interfaces used in Nav2Nav Simulation

The Nav2NavSimController object is the controller of the simulation. This object is tasked with maintaining an influx of traffic, conveying what is visible to Drivers, and simulating over-the-air transmissions between communicating Nav2NavUnits. It also contains the information (geometry, topology, and attributes) of the intersection within the respective Intersection object. The Intersection object is passed to the Nav2NavUnits to serve as the unit's navigable database. The controller utilizes three "feeder" objects to accomplish its tasks. The TrafficFeeder object creates new Car objects and evenly distributes them across the entrance roads to the intersection according to some specified traffic rate (cars/min). It assigns an initial speed and bearing to the Car. Control of the Car after this initialization is conducted completely by the Driver object. The TrafficFeeder randomly picks both an entrance point and exit point for the Car. This destination is communicated to the Driver within the Car, where the Driver specifies this information to the Nav2NavUnit.

Driver vision is simulated by the DriverVisionFeeder object. Each Driver object contains a polygon which represents its field of view. The DriverVisionFeeder iterates through all Driver objects, obtains their field of view polygon, and determines which other objects are within this field. The list of objects is conveyed back to the Driver object, which uses this information to carry out decision making based on the entities it "sees".

The Nav2NavCommunicationFeeder operates in a similar manner to the DriverVisionFeeder. Each Nav2NavUnit contains a circle which represents the area within which the unit is able to communicate with other devices. The Nav2NavCommunicationFeeder object iterates through all Nav2NavUnit objects, obtains the zone that represents the unit's communicable range, and determines which other units would be able to send/receive over-the-air (OTA) messages with the unit. The list of communicable devices is passed in to each unit to provide the means of communication.

Car objects are used to represent vehicles on the road. They have attributes such as bearing and speed which allow for movement of the Car through the simulated space. The Car object updates its position based on its current speed and bearing. The Car's body is represented by a simple rectangle. Each Car object contains two sub-objects, a Driver and Nav2NavUnit.

Driver objects allow for simulation of driver behavior within the simulation environment. Driver objects are the only entities that are able to cause acceleration of Car objects. Each Driver object contains a field of view polygon which allows the DriverVisionFeeder to provide the Driver with objects that are within view. The Driver object makes a judgment based on what they "see" and their perceived distance to the other entity. There are many parameters where we can simulate driver behavior. For instance, aggressive drivers will maintain a smaller distance between vehicles. Drivers with poor vision will have a smaller field of view with which to perceive. Reaction time can also be simulated based on how often the Driver executes through its decision making processes. As stated previously, each Driver that is created as part of new Car objects created by the TrafficFeeder, is given a destination. The Driver knows this destination and determines which way to turn as it approaches the intersection.

Nav2NavUnit objects are simulated to carry out the decision making processes for collision avoidance. As a navigation device, the unit contains a reference to the Intersection object contained in the controller to serve as its navigable database. It also provides geopositioning by taking the true (known) position of the Car and introducing some error to the position to simulate accuracy of GNSS technologies. After obtaining the simulated position, the unit uses map matching to determine the street on which its host Car is currently travelling. Each Nav2NavUnit contains a circular object to represent the area within the unit's communication range. The Nav2NavCommunicationFeeder uses this circle to determine which other units are within this unit's range to provide simulated OTA communication between devices. The Nav2NavUnits derive the three zones as discussed in Section 2.2.

Two interfaces are also implemented to allow uniform behavior across several classes. The Nav2NavSimObject interface provides a set of methods for simulation of entities that would be tangible in the real world, such as cars, people, and intersections. Methods include means to get an object's position, the type of object, and the object's bearing. This interface is mostly used for the Driver's vision. As the driver

needs to be able to "see" objects, determine what they are, judge the distance to them, and take appropriate actions. The Nav2NavCommunicable interface provides methods to simulate OTA communications between devices. Currently, as we envision Nav2Nav units to be a standalone system for collision avoidance, only Nav2NavUnit objects implement this interface to allow communication. However, it may be possible for Nav2Nav units to communicate with certain infrastructure-like devices. For example, a road construction crew working near an intersection may set up a beacon to communicate with Nav2Nav units of their whereabouts to assist in the safety of construction crews. Using this interface will allow for such instances to be simulated in the future, and ensure a uniform communication means amongst entities.

Figure 5 depicts a graphical representation of the entities discussed above.

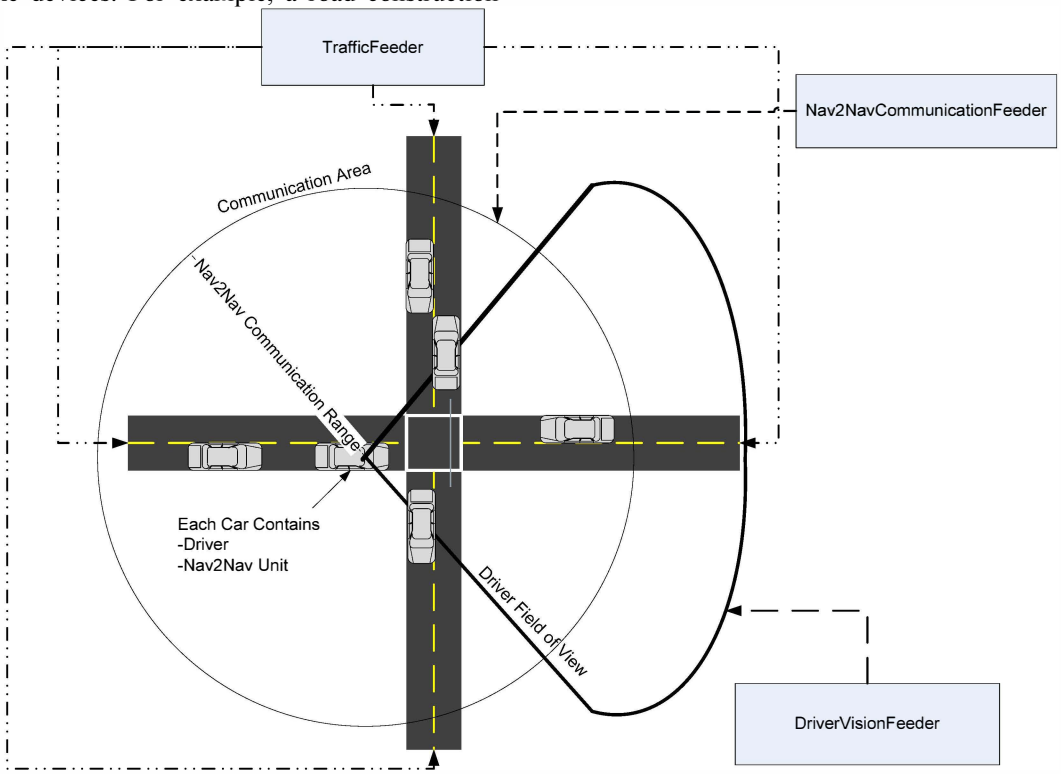

Fig. 5. Graphical Representation of Simulation Entities

V. Analysis of Results

Initial simulation runs enabled us to refine the Nav2Nav model while analyzing its execution. This includes what information will be contained in the broadcast message that is sent out to communicable navigation units involved in the process, the derivation of the broadcast, decision making, and actuation zones, and how to present the important results from this process to a driver while still giving them enough time to process the information and act on it. The first iteration has become a framework of the process to which more functionality will be added.

The broadcast message is integral to the success of the Nav2Nav process. The contents must contain enough information about the current car that other cars' Nav2Nav units can use them to adequately determine the possibility of the two cars colliding and help prevent an accident. The message components that were first explored are outlined in Table 3.

TABLE III
BROADCAST MESSAGE CONTENTS

Message Content	Data Type	Description
Navigation Device ID	String	Unique identifier of the car generated by the navigation unit
Position: X-Coordinate	Double	X-Coordinate of where the navigation unit believes the car is
Position: Y-Coordinate	Double	Y-Coordinate of where the navigation unit believes the car is
Speed	Double	Current speed of the car determined by the navigation unit
Bearing	String	Radians from due East
Intention at Intersection	String	Based on the destination, the navigation unit's direction to the driver
Time to Intersection	Double	Amount of time it will take the car to reach the middle of the intersection with its current speed and bearing

The Navigation Device ID is the unique identifier of the car within the ad-hoc network of vehicles that is created

around each intersection. For the process to be as precise as possible, the ID must be unique, but as privacy concerns are also an issue, it must be anonymous as well. It was determined that one of the ways to generate a unique and anonymous ID would be to create an ID that is valid only for the specific intersection being approached, at that point in time. A new ID could be used for each intersection by the same car. The process created the ID when the navigation unit entered the broadcast zone of the intersection and invalidated it once it had left the process on the other side of the intersection. This way, there would be no single ID for a navigation unit over time, but a new one for every Nav2Nav process that the car was involved in. This helps ensure anonymity. Taking advantage of the principle that no two units would be able to occupy the same space and time, the ID is comprised of the position of the car as determined by the navigation unit and the current time in milliseconds. These values are concatenated together to create a unique and anonymous string as follows:

String NavID = Position.X + Position.Y + CurrentTimeInMilliseconds

There is still a possibility of *Position.X*, *Position.Y*, and *CurrentTimeInMilliseconds* being the same for two cars if both cars are in two lanes next to each other travelling at the same speed. If they cross into the broadcast zone at the same time, all three variables could end up being the same for both vehicles. The main issue would be the coordinates that the navigation unit has for the car's location. There would be some amount of GNSS error that might give two cars the same position. Exploring this problem further in the simulation will allow us to judge what resolution is necessary to ensure that a situation like this does not occur, or that it becomes very statistically unlikely.

The issue of what data type to use for the Navigation Device ID was examined as String and Double are both valid types, each with their own advantages. Double was originally preferred as it requires 64-bits of storage and bandwidth whereas a String is 8-bits per character and the ID would contain a large amount of characters to ensure no duplicates would be generated if a String were used. The String will always take up more space than the Double. If the NavID were generated as a Double, the three values would simply be added together and the resulting value would be the ID. It seemed more likely that this process might create some duplicates, while just concatenating the three values into a String would be much less likely to end up with multiple of the same IDs.

The X and Y coordinates provided by the navigation unit in the message will be necessary for other cars' navigation units to determine where other units are located. Using this information along with the speed and bearing allow the receiving navigation unit to determine the movement of the car along the road network. The *Intention at Intersection* component of the message is based off of the destination that was entered into the navigation unit by the driver. Assuming that the driver will stick to this destination and follow the directions outlined by the navigation unit, the navigation unit will know what the driver will do at the intersection, such as turn left, turn right, or go straight. Each of these will be quantified in the Intention at Intersection component as "LEFT", "RIGHT", or "STRAIGHT". This is very important as receiving units can take this information and decide if a collision is possible at all very quickly. If both cars involved in the process are travelling in opposite directions along a road parallel to each other and each intends to continue going straight through the intersection, there would be a very low chance of a collision in terms of the vehicles crossing paths. The *Time to Intersection* is calculated by the sending navigation unit by dividing the the distance the car is from the center of the intersection by the speed of the car. This is important for receiving units to know as it gives an approximate time it will take the car to reach the intersection. This can then be used to calculate the percentage chance of the sending and receiving units arriving at the middle of the intersection at the same time, if their intentions at the intersection will cause them to cross paths.

The edge of the broadcast zone of the Nav2Nav model is the section which initiates the Nav2Nav process. Once a car crosses this line, it begins transmitting its broadcast message to all other cars involved in the process. The overall distance from the intersection of the process must be carefully examined as the three zones will split up this total distance to establish their location in the model. In order to determine the appropriate radius, the optimal speed and the overall time it takes for the Nav2Nav process to run from start to finish for one intersection had to be outlined. The current simulation takes the highest speed limit of the roads that cross at the intersection and assigns that value to the optimal speed of the intersection. For initial simulation runs, the process time of 15 seconds was chosen. This provided the simulation with a manageable distance for the zones along with sufficient time for each step of the process. The time of 15 seconds is split up evenly between each zone for now. This will change in future simulation runs as 5 seconds for the decision making zone is more than enough time, while 5 seconds for the actuation zone may not give drivers enough time to react. Time will be taken from the decision making zone and added onto the time spent in the actuation zone. The simulation currently derives the broadcast radius, r_b, of the process by multiplying the optimal speed and the overall process time.

$$r_b = optimalSpeed * processTime$$

This total distance is then equally divided up between the three zones as outlined in Figure 6.

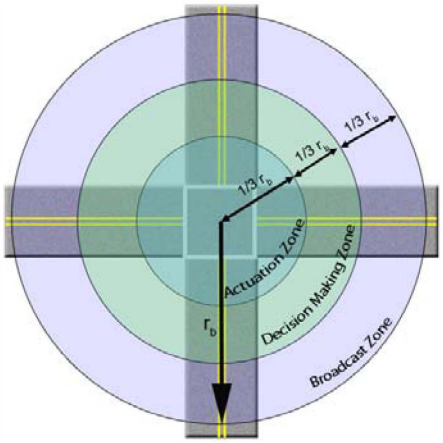

Fig. 6. Derivation of Nav2Nav Zones

The intersection that was developed has an optimal speed of 20.11 meters/second. Using this information with the assumed process time of 15 seconds, the radius of the broadcast zone is 301.65 meters, r_b. When a car is 301.65 meters away from the intersection in this situation, the Nav2Nav process will begin. Each zone makes up 100.55 meters of this total radius. For future simulation runs, this will be modified to give the actuation zone more area, while the portion of r_b dedicated to the decision making zone will be significantly decreased.

Throughout the process and specifically the decision making zone, each car keeps track of what cars it might collide with and the percentage chance of both cars reaching the center of the intersection at the same time. Assuming each car maintains the speed provided in the broadcast message, the percentage of a possible collision situation can be calculated by taking the car with the smaller approximate time to reach the intersection and dividing this value by the approximate time to reach the intersection for the remaining car. A message is then generated by the navigation unit for the car involved in the decision making process with the following information:

258.16559999999964352.01260844172250 has a 77.27% chance of colliding with 484.0116.56567999999983 1260844149298

Once the car enters the actuation zone, all of the information pertaining to potential collision situations will be aggregated together and will be passed to the driver through the navigation unit. What the format of the message should be needs further research. Currently the navigation unit object will pass a message to the driver object that contains the number of vehicles to be aware of and for each one, which direction they are coming from as well.

VI. Conclusion and Future Research

Nav2Nav could have a huge benefit for the safety of drivers. It would catch details that drivers might miss with their own eyes. This becomes especially helpful in situations involving low visibility (e.g., fog) and at confusing intersections (e.g., 5-way and 6-way intersections). Obtaining information from other cars' Nav2Nav units also makes it possible to learn other drivers' intentions. Knowing the routes, something inherent to personal navigation devices, allows the Nav2Nav unit to decide if a collision is imminent with no augmentation to the car's onboard systems or roadside infrastructure.

For Nav2Nav to be practically implemented, the process would have to be standardized in order for all personal navigation device manufacturers to implement the process in their devices. This would ensure that all devices' wireless communication radios have a minimum acceptable range, as well as that the navigation information communicated is in a standard format, such as GML.

Future work will involve implementing a large-scale simulation to be run on a grid computing environment with each CPU representing an intersection, one instance of the intersection collision avoidance problem. By dedicating one CPU to each intersection, a large number of cars and Nav2Nav units can be simulated on each intersection without losing performance. To simulate topology, each CPU will receive a list of CPUs (intersections) that it connects to, thus modeling a road network. When a car object leaves the broadcast zone of the intersection that it is on, the car and the objects it contains will be transferred to the adjacent intersection of that segment for processing. Many different types of intersections will be examined to cover as many scenarios as possible. Modular design of the simulated entities will allow the simulation to be run multiple times with different variables such as traffic volume, with/without Nav2Nav units, different road network layouts, and driver behavior.

Simulating a network of intersections will be crucial to test the effectiveness of the Nav2Nav process. Several situations are in need of examination. First, while we focus on one intersection as one instance of the collision avoidance problem, the network of intersections that exists in a road network renders them not mutually exclusive. Since cars travel between them, the traffic flow between them will test the effectiveness of the Nav2Nav process and help us identify scalability issues as we test. Second, the radius of the broadcast zone may cover two intersections at the same time. This would almost certainly be true in network-dense, urban environments. This could cause some interesting problems with the decision making process.

References

[1] NHTSA (2008). National Motor Vehicle Crash Causation Survey: Report to Congress. July 2008. Accessed October 13, 2009. http://www-nrd.nhtsa.dot.gov/Pubs/811059.PDF.

[2] Chiang, K.W. and El-Sheimy, N. (2002). INS/GPS integration using neural networks for land vehicle navigation applications. In *15th International Technical Meeting of the Satellite Division of The Institute of Navigation*. Portland, OR, USA, 24-27 Sept.: Institute of Navigation, 3975 University Drive, Suite 390, Fairfax, VA, 22030, USA.

[3] Kao, W.-W. (1991). Integration of GPS and dead-reckoning navigation Systems. In *Proceedings of IEEE Vehicle Navigation and Information Systems Conference (VNIS '91)*, 20-23 October. p.635-643.

[4] Phuyal, B. (2002). Method and use of aggregated dead reckoning sensor and GPS data for map matching. In *the Institute of Navigation (ION) annual conference*. Portland, OR, September 27.

[5] Shin, E.H. and El-Sheimy, N., Accuracy improvement of low cost INS/GPS for land applications. 2003, University OF Calgary.

[6] Retscher, G. and Kealy, A. (2006). Ubiquitous Positioning Technologies for Modern Intelligent Navigation Systems. *Journal of Navigation*. **59**: p. 91-103.

[7] Rizos, C. (2005). Trends in geopositioning for LBS, navigation and mapping. In *Proceedings International Symposium and Exhibition on Geoinformation*. Penang, Malaysia, 27-29 Sept.

[8] http://mctrans.ce.ufl.edu/featured/tsis/Version5/corsim.htm

[9] http://www.vissim.us/

[10] http://www.caliper.com/transmodeler/default.htm

[11] Yuhara, Naohiro and Jun Tajima. Multi-driver agent-based traffic simulation systems for evaluating the effects of advanced driver assistance systems on road traffic accidents.

Cognition, Technology and Work. Volume 8 , Issue 4 (October 2006): 283 – 300
[12] Quddus, M. A. (2006). High integrity map matching algorithms for advanced transport telematics applications. Ph.D. thesis, Imperial College, London, United Kingdom.
[13] Karimi, H. A., Conahan, T. and Roongpiboonsopit, D. (2006). A methodology for predicting performances of map-matching algorithms. *Preceedings of the 6th International Symposium on Web and Wireless Geographical Information Systems (W2GIS 2006)*, Hong Kong, 202-213.
[14] Roongpiboonsopit, D. and Karimi, H. A. A Multi-Constellations Satellite Selection Algorithm for Integrated GNSSs. *Journal of Intelligent Transportation Systems*, Vol. 13, No. 3, pp. 127-141 (2009).