Driver Aggressivity Analysis within the Prediction In Dynamic Environments (PRIDE) Framework

C. Schlenoff, Z. Kootbally and R. Madhavan

Intelligent Systems Division, National Institute of Standards and Technology (NIST)
100 Bureau Dr. Stop 8230, Gaithersburg, MD 20899-8230, USA

ABSTRACT

PRIDE is a hierarchical multi-resolutional framework for moving object prediction that incorporates multiple prediction algorithms into a single, unifying framework. PRIDE is based upon the 4D/RCS reference model architecture and provides information to planners at the level of granularity that is appropriate for their planning horizon. This framework supports the prediction of the future location of moving objects at various levels of resolution, thus providing prediction information at the frequency and level of abstraction necessary for planners at different levels within the hierarchy. To date, two prediction approaches have been applied to this framework. In this paper, we provide an overview of the PRIDE (Prediction in Dynamic Environments) framework and describe the approach that has been used to model different aggressivities of drivers. We then explore different aggressivity models to determine their impact on the location predictions that are provided through the PRIDE framework. We also describe recent efforts to implement PRIDE in USARSim, which provides high-fidelity simulation of robots and environments based on the Unreal Tournament game engine.

Keywords: PRIDE, aggressivity, moving object prediction, traffic simulation

1. INTRODUCTION

The field of autonomous systems is continuing to gain traction both with researchers and practitioners. Funding for research in this area has continued to grow over the past few years, and recent high profile funding opportunities have started to push theoretical research efforts into practical use. Autonomous systems in this context refer to embodied intelligent systems that can operate fairly independently from human supervision.

Many believe that the DEMO III Experimental Unmanned Vehicle (XUV) effort represents the state of the art in autonomous off-road driving.[1] This effort seeks to develop and demonstrate new and evolving autonomous vehicle technology, emphasizing perception, navigation, intelligent system architecture, and planning. It should be noted that the DEMOIII XUV has only been tested in highly static environments. It has not been tested in on-road driving situations, which include pedestrians and oncoming traffic. There have also been experiments performed with autonomous vehicles during on-road navigation. Perhaps the most successful has been that of Dickmanns[2] as part of the European Prometheus project in which the autonomous vehicle performed a trip from Munich to Odense ($>$ 1600 kilometers) at a maximum velocity of 180 km/h. Although the vehicle was able to identify and track other moving vehicles in the environment, it could only make basic predictions of where those vehicles were expected to be at points in the near future, considering the vehicle's current velocity and acceleration.

What is missing from all of these experiments is a level of situation awareness of how other vehicles in the environment are expected to behave considering the situation in which they find themselves. When humans drive, they often have expectations of how each object in the environment is expected to move according to the

Commercial equipment and materials are identified in this paper in order to adequately specify certain procedures. Such identification does not imply recommendation or endorsement by NIST, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

Further author information: (Send correspondence to C.S.)
C.S.: E-mail: craig.schlenoff@nist.gov, Telephone: +1 301-975-3456
Z.K.: E-mail: zeid.kootbally@nist.gov, Telephone: +1 301-975-3428
R.M.: E-mail: raj.madhavan@nist.gov, Telephone: +1 301-975-2865; Research Staff Member, Computational Sciences and Engineering Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA.

situation they find themselves in. When a vehicle is approaching an object that is stopped in the road, we expect it to slow down behind the object or try to pass it. When we see a vehicle with its blinker on, we expect it to turn or change lanes. When we see a vehicle traveling behind another vehicle at a constant speed, we expect it to continue traveling at that speed. The decisions that we make in our vehicle are largely based on these assumptions about the behavior of other vehicles.

Most of the work in the literature dealing with drivers' actions and predicted behavior has been performed by psychologists in an attempt to explain drivers' behaviors and to identify the reason for certain dysfunctions.[3–5] To the best of the authors' knowledge, none of the reported research has been applied to autonomous driving. To address this need, we have developed a multi-resolutional, hierarchical framework, called PRIDE (PRediction In Dynamic Environments) that provides an autonomous vehicle's planning system with information that it needs to perform path planning in the presence of moving objects.[6,7] This framework supports the prediction of the future location of moving objects at various levels of resolution, thus providing prediction information at the frequency and level of abstraction necessary for planners at different levels within the hierarchy.

In this paper, we analyze the influence of driver aggressivity in predicting future location of vehicles within the PRIDE framework. We pay special attention to how the perceived aggressivity of a vehicle can affect the long-term predictions by simulating a handful of possible situations that a vehicle may encounter and modifying the aggressivity level to see how the behavior in those situations change.

This paper is organized as follows: Section 2 provides an overview of the PRIDE framework. Section 3 describes the integration of the PRIDE Framework with the "Mobility Open Architecture Simulation and Tools" (MOAST) and the "Urban Search and Rescue Simulation" (USARSim) simulation environment. Section 4 gives a description of the Road Network Database and how it is used by PRIDE. Section 5 details the long-term (LT), cost-based, probabilistic moving object prediction algorithms. Section 6 discusses the role of aggressivity in PRIDE and describes how it is addressed. Section 7 describes the results of varying aggressivity on long-term prediction and Section 8 concludes the paper.

2. THE PRIDE FRAMEWORK

To understand the way that PRIDE was developed and the functionality that it is intended to provide, it is important to understand the 4D/RCS architecture,[8] on which it was based. 4D refers to the four dimensions (three dimensions of space and one dimension of time), and RCS stands for Real-time Control Systems. 4D/RCS was chosen due to its explicit and well-defined world modeling capabilities and interfaces, as well as its multi-resolution, hierarchical planning approach. Specifically, 4D/RCS allows for planning at multiple levels of abstraction, using different planning approaches as well as utilizing inherently different world model representation requirements, as shown for a military environment in Figure 1. By applying this architecture, we can ensure that the representations being developed for representing moving objects can accommodate different types of planners that have different representational requirements.

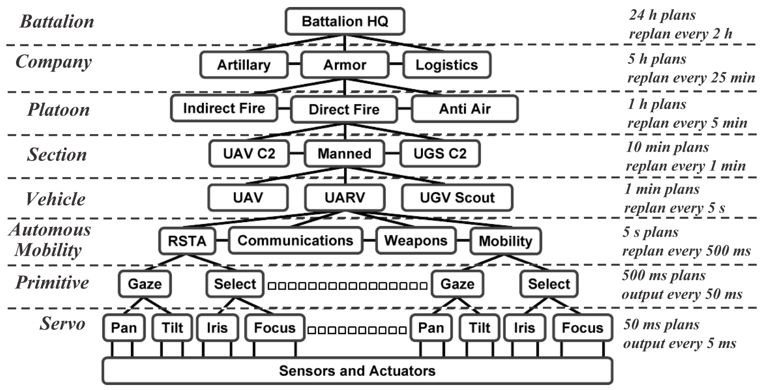


Figure 1. A high level block diagram of a typical 4D/RCS reference model architecture.

The RCS architecture supports multiple behavior generation (BG) systems working cooperatively to compute a final plan for the autonomous system. The spatial and temporal resolution of the individual BG systems along with the amount of time allowed for each BG system to compute a solution are specified by the level of the architecture where it resides. In addition to multiple BG systems, multiple world models are supported with each world model's content being tailored to the systems that it supports (in this case the BG system). As such, it is necessary for moving objects to be represented differently at the different levels of the architecture.

To support this requirement, the National Institute of Standards and Technology (NIST) has developed the PRIDE (PRediction In Dynamic Environments) framework. The underlying concept is based upon a multi-resolutional, hierarchical approach that incorporates multiple prediction algorithms into a single, unifying framework. This framework supports the prediction of the future location of moving objects at various levels of resolution, thus providing prediction information at the frequency and level of abstraction necessary for planners at different levels within the hierarchy. To date, two prediction approaches have been applied to this framework.

At the lowers levels, we utilize estimation theoretic short-term predictions via an extended Kalman filter-based algorithm using sensor data to predict the future location of moving objects with an associated confidence measure. It should be noted here that, in contrast to the long-term predictions, the estimation-theoretic short-term prediction algorithm does not incorporate *a priori* knowledge such as road networks and traffic signage and assumes uninfluenced constant trajectory. At the higher levels of the framework, moving object prediction needs to occur at a much lower frequency and a greater level of inaccuracy is tolerable. At these levels, moving objects are identified as far as the sensors can detect, and a determination is made as to which objects should be classified as "objects of interest". In this context, an object of interest is an object that has a possibility of affecting our path in the time horizon in which we are planning. At this level, we use a moving object prediction approach based on situation recognition and probabilistic prediction algorithms to predict where we expect that object to be at various time steps into the future. Situation recognition is performed using spatio-temporal reasoning and pattern matching with an a priori database of situations that are expected to be seen in the environment. In these algorithms, we are typically looking at planning horizons on the order of tens of seconds into the future with plan steps at about one second intervals. At this level, we are not looking to predict the exact location of the moving object. Instead, we are attempting to characterize the types of actions we expect the moving object to take and the approximate location the moving object would be in if it took that action. This is further described in Section 5.

Active research is exploring the integration of these two prediction approaches in a way that the predictions from one can help to enforce or not enforce the predictions of the other. We have developed a methodology termed *probability scaling* for integrating the long-term and short-term estimates to strengthen or weaken the predictions of each other. We also identified *critical time points* that aid the integration methodology at different time periods, demonstrated the utility of the integration methodology and how identification of critical time points makes PRIDE a better integrated framework. This enabled us to better understand the time frames in which both prediction algorithms provide valid results.[9]

At the point this paper was written, we have simulated many driving situations and have used approximately a dozen costs to determine the probabilities of one action over another. In this context, a cost is a penalty that is incurred by performing a maneuver or occupying a state. Current costs are incurred based on: 1) proximity to other objects in the environment as a function of necessary stopping distance, 2) exceeding or going below the speed limit by a given threshold, 3) changing lanes, 4) not being in the right most lane, 5) rapidly accelerating or decelerating, and 6) changing lanes where double yellow lines in the road exist, among other costs.

It should be emphasized that costs are not static numbers. The cost that a vehicle incurs by taking an action is heavily a function of the perceived aggressivity and intention of the moving objects. Using these costs, we are able to predict up to ten seconds into the future at a rate of two predictions per second. This paper will further explore the concept of aggressivity and show how PRIDE can model and account for drivers of varying aggressivity.

3. INTEGRATION OF PRIDE WITH MOAST/USARSIM

3.1. OVERVIEW OF THE MOAST/USARSIM FRAMEWORK

The PRIDE framework is currently implemented within an open source project, the Mobility Open Architecture Simulation and Tools (MOAST) framework. While MOAST is designed to be used within multiple development environments and on real systems, considerable effort has been exerted to ensure that it is totally compatible with the Urban Search and Rescue Simulation (USARSim). This creates a joint MOAST/USARSim framework that provides a comprehensive set of open source tools for the development, testing, and performance evaluation of autonomous agents. USARSim is based upon the Unreal Tournament game engine and provides realistic environments and embodiment for agents. The environments are full 3D worlds that have photo-realistic textures and objects (Figure 2). Embodiment is aided by the Karma physics engine that allows for physics-based interactions with objects in the environment. The MOAST framework provides the intelligence for the embodied agent. It consists of an architecture, control modules, interface specs, and data sets. MOAST is fully integrated with the USARSim simulation system. The framework is intended to provide tools to aid a researcher through

(a) (b) (c)

Figure 2. 3D worlds in USARSim.

all phases of development and testing of an autonomous agent system. MOAST is made up of a reference model architecture that dictates how control responsibilities are divided between modules, a communication interface specifications that dictate how and what modules will communicate, a sample control modules for the control of a sample simulated robotic platform, a sample datasets for use in the simulation and at last, tools to aid in development and debug of the control system.

MOAST is designed to support real agent components and simulated components running cooperatively in the same world (real/virtual operation). For example, sensor processing may be simulated while mobility is performed on a real agent. With this in mind, a complete structure is provided to interface into the USARSim simulation system. MOAST implements a hierarchical control technique which decomposes the control problem into a hierarchy of controllers with each echelon (or level) of control as depicted in Figure 1, adding additional capabilities to the system. The echelons may be generalized as follows: 1) Primitive (PRIM): The PRIM echelon accepts mobility commands in the form of constant curvature arcs or line segments to follow. It then computes the SERVO commands and issues these to the SERVO echelon. 2) Autonomous Mobility (AM): The AM echelon accepts mobility commands in the form of waypoints and converts these into constant curvature arcs for the PRIM echelon to follow. 3) Vehicle (VEH): The VEH echelon accepts a general vehicle behavior and then coordinates multiple AM commands in order to carry out this behavior. An example behavior is exploration. 4) Section (SECTION): The SECTION echelon coordinates the behavior of multiple vehicles. It decides on the individual goals and behaviors of the vehicles to accomplish a high-level mission.

The MOAST framework connects into USARsim via a Simulation Interface Middleware (SIMware)[10] and provides additional capabilities for the system. These capabilities are encapsulated in components that are designed based on the hierarchical 4D/RCS Reference Model Architecture. Each echelon (or level) of the 4D/RCS architecture performs the same general type of functions: sensory processing (SP), world modeling (WM), value judgment (VJ), and behavior generation (BG).

3.2. INTEGRATION OF PRIDE

The communication between PRIDE and MOAST is performed by exchanging information using the Neutral Message Language (NML).[11] The PRIDE framework assumes knowledge of the current position and the velocity of the vehicles on the road to predict their future locations. At this step, the predicted position and the velocity of each vehicle are computed and sent to MOAST to exert control at the PRIM level. The primitive echelon BG is in charge of translating constant curvature arcs or position constraints for vehicle systems into velocity profiles for individual component actuators based on vehicle kinematics. For example, the AM Mobility BG will send a dynamically correct constant curvature arc for the vehicle to traverse. This trajectory will contain both position and velocity information for the vehicle as a whole. During the trajectory execution, BG will read vehicle state information from the Servo Echelon WM to assure that the trajectory is being maintained and will take corrective action if it is not. Failure to maintain the trajectory within the commanded tolerance will cause BG to send an error status to the AM Mobility BG.

4. THE ROAD NETWORK DATABASE

To compute the future location of an autonomous vehicle, the PRIDE framework should have knowledge of the environment where the vehicle evolves. The PRIDE algorithms consider different information from the road network structure, the length and width of a lane are used to compute the future location of the vehicle inside this lane, the curvature center is utilized so that the vehicle uses the right angle to turn, the vehicle has to know the speed limit on a particular lane to avoid speed excess. All this information is essential for a vehicle to be able to navigate a road network. The road network must be described in such a way that an autonomous vehicle knows, with sufficient precision and accuracy, where the road lies, rules dictating the traversal of intersections, lane markings, road barriers, road surface characteristics, and other relevant information.

The purpose of the road network database[12] is to provide the data structures necessary to capture all of the information necessary about road networks so that a planner or control system on an autonomous vehicle can plan routes along the roadway at any level of abstraction. Each level of planning requires data at different levels of abstraction, and as such, the road network database must accommodate these requirements. Some components of the road network database used by PRIDE are described below.

- *Lane*: A lane is a single pathway of travel that is bounded by explicit or implicit lane marking. Lanes span the length of a lane cluster in which they are a part of.
- *Lane Segment*: A lane segment is the most elemental portion of a road network captured by the database structure. Lane segments can be either straight line or constant curvature arcs. In the case of a straight line, the location of the lane segment if fully defined by the beginning and end point of the lane segment. For a constant curvature arc, the lane segment is defined by the beginning and end of the lane segment and the curvature center point. One or more lane segments compose a lane.
- *Junction Lane Segment*: A junction lane segment is a constant curvature path through a portion of a lane junction. Apart from some subtle differences pertaining to connectivity of these junction lane segments, they are extremely similar to lane segments.

The road network database structure is designed to accommodate a control system that may contain planners with various levels of abstraction. Planners used by PRIDE are described in Table 1.

Planner Name	*Planner Description*
Elemental Maneuver Planner	Carries out real-time maneuvers to slow down, stop, speed up, and change lateral position. Plans on the order of 10 s into the future.
Goal Path Trajectory Generator	Calculates the lane segment path dynamic trajectory as a goal path to carry out commanded move while controlling for skid and immediate obstacle response. Plans on the order of 1 s into the future. Plans up to 5 m distances.

Table 1. Planners at different levels of abstraction.

5. LONG-TERM PREDICTION

The algorithm described in this section is used to predict the future location of moving objects for longer time horizons. Figure 3 graphically shows the overall process flow.

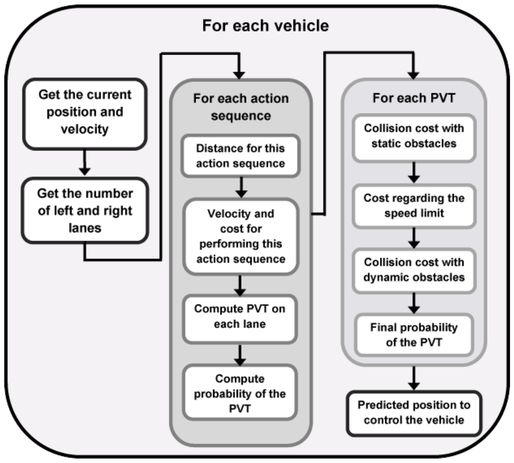

For each vehicle
Get the current position and velocity
Get the number of left and right lanes
For each action sequence
Distance for this action sequence
Velocity and cost for performing this action sequence
Compute PVT on each lane
Compute probability of the PVT
For each PVT
Collision cost with static obstacles
Cost regarding the speed limit
Collision cost with dynamic obstacles
Final probability of the PVT
Predicted position to control the vehicle

Figure 3. The situation-based probabilistic (long-term) prediction process.

5.1. POSSIBLE VEHICLE ACTIONS

The process of predicting several time steps into the future consists of a series of continuous actions which constitute a driving procedure. Each action is accomplished in one time step, thus, for a time of prediction n, n actions will be completed. The long-term prediction algorithms use different types of actions. The first type of actions consists of a set of speed profiles: Quick Acceleration (QA), Slow Acceleration (SA), Keep the same Speed (KS), Quick Deceleration (QD), Slow Deceleration (SD). The second type of actions concerns the changing of lanes: a vehicle has the possibilities of staying in its lane (SL), changing to the right lane (CR), changing to the left lane (CL). The last type of action pertains to intersections, a vehicle has the possibility to turn left, to turn right or to go straight through an intersection.
At this step, for each vehicle on the road, the algorithm computes all possible sequences of actions, regarding the current velocity and location. Some actions may not be possible due to the vehicle's current velocity (for example, a vehicle moving slowly cannot change lanes in one second during a deceleration). In this case, those actions are not considered. Each sequence of actions is generated in a realistic way using rules. Presently, a single rule is applied to all of the possible action sequences to generate the most realistic ones. To evaluate these rules, we associate a value to each 'acceleration profile': 2 for QA, 1 for SA, 0 for KS, -1 for SD, and -2 for QD. The rule states that a vehicle can only switch from an action to another action if their values differ at most by one. An example of action sequences and their associated validity is shown in Table 2.

Actions				*Validity*	*Description*
SD	SD	SD	SD	Valid	
QD	QD	QA	QA	Invalid	QD to QA illegal

Table 2. Example of valid and invalid sequences of actions.

5.2. COST MODEL

The sequences of actions are deemed finite, and the probabilistic LT prediction algorithms use an underlying cost model that simulates the danger that a driver would incur by performing an action or occupying a state.[6] These costs are being used by multiple efforts within the program that this effort is a part of. Thus, there is value of building the probabilities directly from these costs to allow for synergy with other efforts. These costs can be separated in two different categories:

1. The cost representing the vehicle's actions: This cost represents the penalties for performing an action

as a function of the amount of attention needed. For example, the changing lane action needs more concentration than going straight in the same lane, thus the cost for changing lane is greater.

2. Cost representing the vehicle's state on the road: The proximity to other static and dynamic objects on the road is assigned to a cost of collision with these objects. Examples of static objects on the road are road blocks, debris, etc. Examples of dynamic objects on the road are other vehicles. The costs associated with static or moving objects is proportional to the danger and imminence of collision. For example, a road block at one kilometer ahead is less dangerous than another vehicle passing at three meters ahead.

Examples of costs are shown in Table 3.

Action	*Cost*
Quick Acceleration (QA)	5
Quick Deceleration (QD)	5
Changing lane (CL, CR)	20
Opposite direction	500
Collision (CO)	1000
Being under the speed limit (US)	5
Being over the speed limit (OS)	5

Table 3. Example of actions with their corresponding costs.

5.3. PREDICTED VEHICLE TRAJECTORY

Cost of collision between vehicles are computed using Predicted Vehicle Trajectories (PVTs) which represent the possible movements of vehicle throughout the time period of prediction being analyzed. A PVT is a vector whose origin represents the current position of the vehicle $(x_{IP}, y_{IP}, t_{IP} = 0)$ at $time = 0$ and its extremity represents the predicted position $(x_{PP}, y_{PP}, t_{PP} = t_{pred})$ where t_{pred} is the predetermined time in the future for the prediction process. Also contained within the PVT is the action-cost and action-probability information.
A collision is detected when PVTs cross each other, the location and time of the collision is determined using a parameterization of each PVT. These information can be obtained by using a parametrization of each PVT as represented in the following equations.

$$\begin{cases} x_1(t_1) = x_{PP_1} t_1 + x_{IP_1}(1 - t_1) \\ y_1(t_1) = y_{PP_1} t_1 + y_{IP_1}(1 - t_1);\ t_1 \in [0, 1] \end{cases} \tag{5.1}$$

$$\begin{cases} x_1(t_2) = x_{PP_2} t_2 + x_{IP_2}(1 - t_2) \\ y_1(t_2) = y_{PP_2} t_2 + y_{IP_2}(1 - t_2);\ t_2 \in [0, 1] \end{cases} \tag{5.2}$$

where t_1 and t_2 are the parameters for each PVT. Equations (5.1) and (5.2) create a linear system where t_1 and t_2 can be solved using Cramer's rule:

$$t_1 = \frac{\begin{vmatrix} x_{IP_2} - x_{IP_1} & x_{IP_2} - x_{PP_2} \\ y_{IP_2} - y_{IP_1} & y_{IP_2} - y_{PP_2} \end{vmatrix}}{\begin{vmatrix} x_{PP_1} - x_{IP_1} & x_{IP_2} - x_{PP_2} \\ y_{PP_1} - y_{IP_1} & y_{IP_2} - y_{PP_2} \end{vmatrix}} \quad t_2 = \frac{\begin{vmatrix} x_{PP_1} - x_{IP_1} & x_{IP_2} - x_{IP_1} \\ y_{PP_1} - y_{IP_1} & y_{IP_2} - y_{IP_1} \end{vmatrix}}{\begin{vmatrix} x_{PP_1} - x_{IP_1} & x_{IP_2} - x_{PP_2} \\ y_{PP_1} - y_{IP_1} & y_{IP_2} - y_{PP_2} \end{vmatrix}}$$

The two vehicles will cross each other at two different times, (t_1, t_{pred}) for the first vehicle, (t_2, t_{pred}) for the second vehicle. For a small difference between the two times, the collision is probable or certain. Conversely, for a large difference, the collision is improbable. Thus if the PVTs cross and the difference of time is less than a predetermined time (τ), we use Equation (5.3) to determine the collision cost:

$$Collision\ Cost = CO\left(\tau - (t_{pred}|t_1 - t_2|)\right) \tag{5.3}$$

where CO is the predetermined maximum cost than can occur when colliding with a specific object (Table 3) and τ is the predetermined time difference in which a cost for collision will be incurred.

5.4. FROM COST TO PROBABILITY

As discussed previously, the PRIDE algorithms compute n realistic sequences of actions with an associated cost. Based on this cost, we can determine the probability that the vehicle will perform that sequence of actions in the following way. The first step is to create a ratio of the cost for performing a given sequence of actions to the sum of all of the costs for performing n sequences of actions:

$$ratio_i = \frac{\sum_{j=1}^{n} cost_j}{cost_i}, \forall\ i \in [1, n]$$

We then normalize the ratio of each sequence of actions by dividing it by the sum of all of the ratios, as shown in Equation (5.4):

$$proba_i = \frac{ratio_i}{\sum_{j=1}^{n} ratio_j}, \forall\ i \in [1, n] \tag{5.4}$$

Equation (5.4) computes the normalized probability of a given sequence of actions occurring as compared to all sequences of actions that are possible at that time.

6. AGGRESSIVITY

Different drivers drive in different ways. One driver may be very conservative, only changing lanes when absolutely necessary, never exceeding the speed limit, etc. On the other hand, another driver may drive very aggressively, weaving in and out of lanes, greatly exceeding the speed limit, and tailgating other drivers. In most cases, one would experience both kinds of drivers on any trip (along with many drivers that fall somewhere in the middle), and a moving object prediction framework needs a mechanism to account for all such circumstances.
When a driver is first encountered, it is extremely rare that one can instantaneously determine the perceived aggressivity of the driver. This information is often determined after observing the driver for a certain amount of time, characterizing their driving behaviors, and assigning an aggressivity. The aggressivity that is assigned greatly impacts PRIDE's predictions as to where that driver will be at times in the future. For example, we would likely assume that a conservative driver will remain in their lanes whenever possible and stay a safe distance behind the vehicle in front of it. An aggressive driver would have a higher probability of changing lanes.
We may also find that the aggressivity of the driver may change over times. There are times when one can observe a driver for many seconds at a time. In this case, the driver's aggressivity may change, perhaps they are very aggressively trying to get to a certain lane but become more passive when they get there.
The PRIDE framework addresses all of these driver types and all of the situations mentioned above. As discussed earlier, the probability that a driver will perform an action is roughly inversely proportional to the cost that they would incur by performing that action. So, for example, a driver would be more likely to perform an action that would only incur a cost of 1 than one that would incur a cost of 5. Examples of costs within PRIDE that would be incurred by performing various actions are represented in Table 4.

In row 3 of Table 4, *SecDist* represents a cost for a vehicle that does not adhere to the security distance toward other objects i.e., for distance(vehicle, other object) > security distance. The cost assigned to the proximity of a vehicle to other objects is then computed according to the security distance, the aggressivity of the driver and the cost of collision with other objects (CO). The parameters $A_P = 1$, $A_{AV} = 5$ and $A_{AG} = 10$ represent the aggressivity value assigned respectively to a passive driver, an average driver and an aggressive driver. These

Action	*Passive driver*	*Average driver*	*Aggressive driver*
Changing lanes	30	20	10
Quick acceleration/deceleration	5	5	5
Proximity to other objects	$\frac{SecDist \times CO}{A_P}$	$\frac{SecDist \times CO}{A_{AV}}$	$\frac{SecDist \times CO}{A_{AG}}$
Exceeding the speed limit	$V_{diff} \times \text{OS} \times A_P$	$V_{diff} \times \text{OS} \times A_{AV}$	$V_{diff} \times \text{OS} \times A_{AG}$
Going below the speed limit	$V_{diff} \times \text{US} \times A_P$	$V_{diff} \times \text{US} \times A_{AV}$	$V_{diff} \times \text{US} \times A_{AG}$

Table 4. Sample aggressivity models.

values have been found from empirical tests. V_{diff} in row 4 and row 5 of Table 4 represents the difference between the predicted velocity of the vehicle and the speed limit of a lane on the road, OS (Over Speed) and US (Under Speed) are the cost for being over the speed limit and being under the speed limit respectively, as shown in Table 3. When the predicted velocity exceeds or goes below the speed limit, the cost of this prediction is increased, so that its probability becomes lower.
From the table above, it is evident that because passive drivers incur a greater cost by performing a given action, they are less likely to perform that action. This mimics the expected behavior of these types of drivers on the roadway. In PRIDE, there are different cost models for each type of driver encountered on the roadway. When a vehicle is first encountered and little is known about its aggressivity, a default model is used with costs representing an average aggressivity. As more information is gained about the aggressivity of the driver, different aggressivity models are applied to that driver as appropriate. It is not the goal of PRIDE to have a unique cost model for every driver. Instead, cost models have been developed for a discretized set of driver types (i.e., very aggressive, moderately aggressive, average, moderately passive, very passive). Drivers that are perceived in the environment can bounce among these aggressivity models as more observations are collected.

7. RESULTS OF VARYING AGGRESSIVITY ON LONG-TERM PREDICTION

We have simulated different traffic situations for multiple Ackerman steered vehicles (Figure 4) in on-road driving scenarios. The vehicles perform different tasks in the presence of obstacles (static and dynamic).

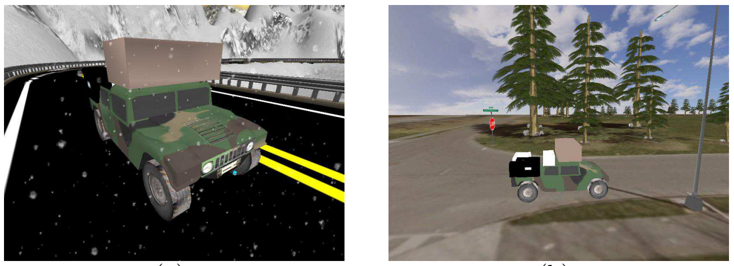
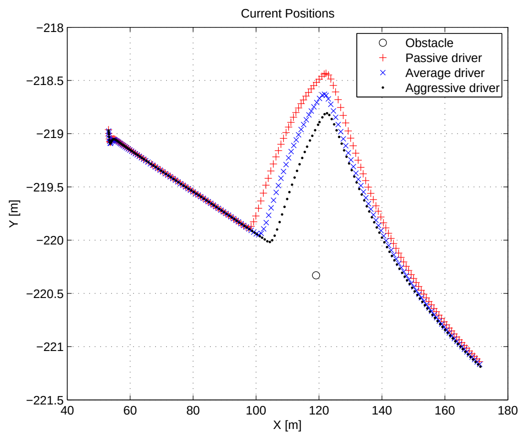
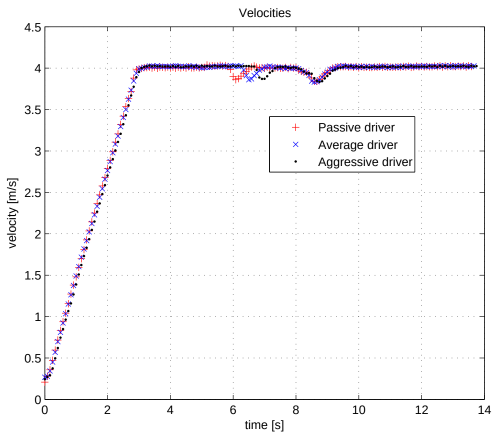

(a) (b)

Figure 4. High-Mobility Multipurpose Wheeled Vehicles in USARSim.

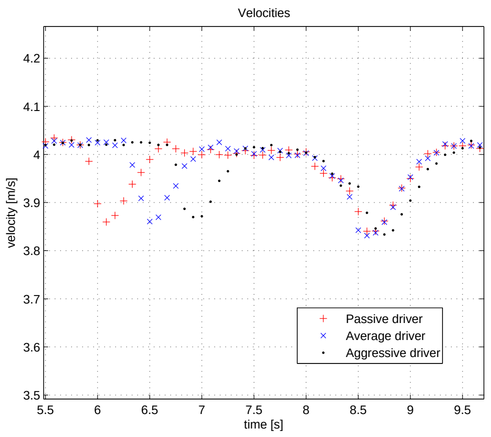
In the first traffic situation, we simulate the action of one vehicle driving in a straight lane avoiding one obstacle. Figures 5(a) and 5(b) respectively depict the positions and the velocities of the vehicle using different levels of aggressivity, the positions and the velocities of the passive driver are represented by a '+', by a 'x' for the average driver and by a '.' for the aggressive driver. Figure 5(c) is a closer view of the velocities when the vehicle passes the obstacle. The vehicle drives from the left to the right and the obstacle is located on the right lane. The vehicle moves on the right lane with an increasing speed (Figure 5(b)), we can clearly see that the aggressive driver is the one that starts to shift to the left lane closer to the obstacle (about 14 m). As seen in Table 4, the cost of collision is computed using the aggressivity of the vehicle; higher the aggressivity lower the cost. Before starting the maneuver of obstacle avoidance, the vehicle with the higher aggressivity comes closer to the obstacle, due to a lower cost of collision, the passive driver starts to move to the left lane far away from the obstacle (21.42 m), the average driver reaches a distance of 18.16 m from the obstacle before starting the passing maneuver. We can see different shapes of the peaks when the vehicle reaches the left lane. The aggressive driver moves around the obstacle very closely, the average and passive drivers avoids the obstacle more carefully, and thus the distance between the obstacle and the vehicle is large for smaller aggressivity. Indeed, we would expect the average and the passive driver to be more conservative than the aggressive driver around an obstacle. In Figure 5(c), we can see a variation of speed of the vehicle when passing the obstacle, there is a small decrease (about 4 %) when the vehicle changes lane. It should be noted here the different times at which the vehicles perform the obstacle avoidance process, the passive driver starts to shift lane at 5.83 s, the average driver at 6.25 s and the aggressive driver at 6.66 s. In the next step, the vehicle changes to the right lane. At this point, the cost for not being in the right lane is used by the vehicle. Since this cost is proportional to the aggressivity value, the aggressive driver goes back in the right lane in a shorter distance after passing the obstacle, the average drive takes a longer distance to reach the right lane and more longer for the passive driver.



(a)



(b)

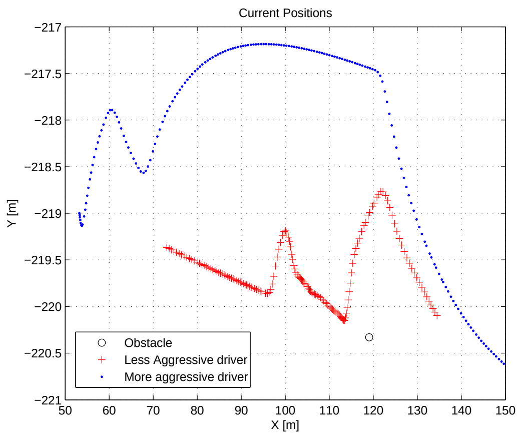


(c)

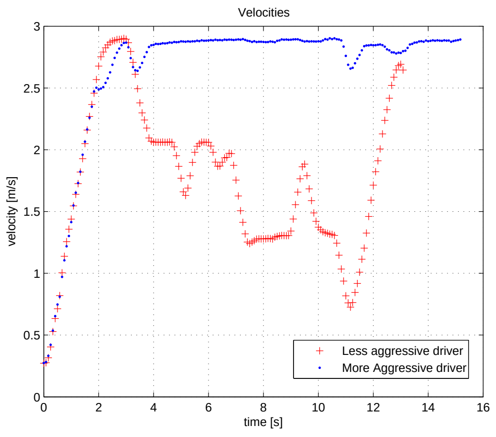
Figure 5. Positions and velocities for different types of driver.

In the second traffic scenario, we use one obstacle (on the right lane) and two vehicles moving in the same lane, one vehicle in the back of the other one. We observe the behaviors of the vehicles for each type of aggressivity. Figures 6(a) and 6(b) show respectively the current positions and the velocities for a more aggressive driver in the back ('.') and a less aggressive driver in the front ('+'). The simulation starts with the two vehicles moving in the same lane and in the same direction (from the left to the right in Figure 6(a)). The more aggressive driver

starts to change lane when he detects an other vehicle within its time period of prediction. At X=60 m, we can see the positions of the vehicle going back in the right lane before reaching the left lane. Since the speed limit for the aggressive driver at this point is not high enough and the distance between the two vehicles is prominent for a passing maneuver, the cost for not being in the right most lane is applied to the aggressive driver. At X=68 m, the aggressive starts a passing maneuver by swerving to the left lane, this process ends at X=121 m where the vehicle starts to move to the right lane.
On the other side, the less aggressive driver starts to move to the left lane (X=95.94 m), at this point, the passive driver detects the static obstacle and starts to move to the left lane far away from the obstacle (23.12 m). Since a passing maneuver is being performed at this time by another vehicle, the less aggressive driver swerves to the right lane to avoid the collision with a dynamic obstacle (aggressive driver), we can see the speed of the passive driver decreasing, the passive driver lets the aggressive vehicle finishing the passing action. The less aggressive driver avoids the static vehicle when there is no more danger with any vehicle in the left lane, the distance between the passive driver and the static obstacle is 5.65 m before moving to the left lane. We should note the superiority of the speed for the aggressive driver in Figure 6(b), to pass a dynamic obstacle, the driver with the higher aggressivity increases his speed. The observed rise and fall pattern of the plotted velocity for the passive driver is due to the presence of an other vehicle with a higher aggressivity on the left lane. The predicted vehicle trajectories of the two vehicles interact constantly during this scenario, the passive driver modifies his speed and thus lead to a more conservative behavior. The velocity of the passive driver starts to increase constantly at 11.17 s when the aggressive driver is far away from the passive driver.
We used this interesting example to show the behavior of a more aggressive driver while driving behind a less aggressive driver. In other simulation tests, we change each type of aggressivity for each vehicle. When the two vehicles have the same level of aggressivity or when the vehicle in the back has the lowest aggressivity, it drives within the security distance with the front vehicle, no vehicle passing maneuver is performed in this situation. As seen, the variation of the velocity for the less aggressive driver is important. The algorithm could be modified to reduce this variation in future simulation tests. We need to also consider the full size of static obstacles. So far, we have only used the position of obstacles on the road. In this way, the PRIDE framework would move closer to reality in the obstacle avoidance process.



(a)



(b)

Figure 6. Positions and velocities for two vehicles with different types of aggressivity.

8. CONCLUSIONS AND FUTURE WORK

In this paper, we have described the PRIDE framework which predicts the future location of moving objects in the environment for the purpose of path planning for autonomous ground vehicles. Specifically, we described the two prediction algorithms in PRIDE (long-term and short-term prediction), how PRIDE has been integrated into the MOAST/USARSim framework, and how it uses the Road Network Database. The emphasis of this

paper was on how PRIDE addresses different aggressivities of drivers; specifically, how this information is used to predict where those drivers' vehicles will be at different times in the future and how this information can be used for traffic simulation. There was also discussion on how varying these aggressivities change the predictions of where these vehicles will be, as shown by simulating a handful of situations that a vehicle may encounter and modifying the aggressivity level to see how the behavior in those situations change.
The addition of aggressivities is the latest enhancement to the PRIDE framework. There are more enhancements planned including introducing more vehicles into traffic situations, exploring the use of fuzzy logic to determine activity selection (as opposed to strictly cost-based rules), using more complex road networks, and validating the results of the PRIDE/MOAST/USARSim frameworks as compared to real-world driving scenarios.

REFERENCES

1. C. Shoemaker and J. Bornstein, "Overview of the Demo III UGV Program," in *Proceedings of the SPIE Robotic and Semi-Robotic Ground Vehicle Technology Conference*, pp. 202–11, 1998.
2. E. Dickmanns, "The Development of Machine Vision for Road Vehicles in the Last Decade," in *Proceedings of the International Symposium on Intelligent Vehicles*, pp. 644–651, 2002.
3. S. Espie, F. Saad, and B. Schnetler, "Microscopic Traffic Simulation and Driver Behavior Modeling: The ARCHISM Project," in *Proceedings of the Strategic Highway Research Program and Traffic Safety on Two Continents*, (Lille, France), 1994.
4. A. Champion, S. Espie, and J. Auberlet, "Behavioral Road Traffic Simulation with ARCHISM," in *Proceedings of the Summer Computer Simulation Conference*, (USA), 2001.
5. S. Hoseini, M. Vaziri, and Y. Shafahi, "Combination of Car Following and Lane Changing Models as a Drivers' Optimization Process," pp. 601–605, 2004.
6. C. Schlenoff, R. Madhavan, and Z. Kootbally, "PRIDE: A Hierarchical, Integrated Prediction Framework for Autonomous On-Road Driving," in *Proceedings of the 2006 International Conference on Robotic Applications (ICRA)*, 2006.
7. C. Schlenoff, J. Ajot, and R. Madhavan, "PRIDE: A Framework for Performance Evaluation of Intelligent Vehicles in Dynamic, On-Road Environments," in *Proceedings of the Performance Metrics for Intelligent Systems (PerMIS) 2004 Workshop*, 2004.
8. J. Albus et al., "4D/RCS Version 2.0: A Reference Model Architecture for Unmanned Vehicle Systems," Tech. Rep. NISTIR 6910, National Institute of Standards and Technology, Gaithersburg, MD, 2002.
9. Z. Kootbally, R. Madhavan, and C. Schlenoff, "Prediction in Dynamic Environments via Identification of Critical Time Points," in *Proceedings of the IEEE Workshop on Situation Management (SIMA), Military Communications Conference*, 2006.
10. S. Carpin and M. Lewis and J. Wang and S. Balakirsky and C. Scrapper., "Bridging the Gap Between Simulation and Reality in Urban Search and Rescue," in *Robocup 2006: Robot Soccer World Cup X, Springer, LNAI*, 2006.
11. W. Shackleford, F. Proctor, and J. Michaloski, "The Neutral Message Language: A Model and Method for Message Passing in Heterogeneous Environments," in *Proceedings of the 2000 World Automation Conference*, (Maui, HI), 2000.
12. C. Schlenoff, S. Balakirsky, A. Barbera, C. Scrapper, E. Hui, M. Paredes, and J. Ajot, "The NIST Road Network Database: Version 1.0," Tech. Rep. NISTIR 7136, National Institute of Standards and Technology, Gaithersburg, MD, USA, July 2004.