PRIDE: A Framework for Performance Evaluation of Intelligent Vehicles in Dynamic, On-Road Environments

Craig Schlenoff, Jerome Ajot, and Raj Madhavan
National Institute of Standards and Technology
Gaithersburg, MD 20899

ABSTRACT

We are developing a novel framework, PRIDE (PRediction In Dynamic Environments), to perform moving object prediction for unmanned ground vehicles. The underlying concept is based upon a multi-resolutional, hierarchical approach that incorporates multiple prediction algorithms into a single, unifying framework. The lower levels of the framework utilize estimation-theoretic short-term predictions while the upper levels utilize a probabilistic prediction approach based on situation recognition with an underlying cost model.

In addition to predicting the location of moving objects in the environment, we have extended PRIDE to generate simulated traffic flow during on-road driving. In this paper, we explore applying the PRIDE-based traffic control algorithms for the purpose of performance evaluation of autonomous vehicles. Through the use of repeatable and realistic traffic flow simulation, one is able to evaluate the performance of an autonomous vehicle in an on-road driving scenario without the risk involved with introducing the vehicle into a potentially dangerous roadway situation. In addition, by varying a single vehicle's parameters (e.g. aggressivity, speed, location) with the traffic flow, we can show how the entire traffic pattern is affected. We will show the successes that have been achieved to date in a simulated environment, as well as enhancements that are currently being researched and expected in the near future.

KEYWORDS: *autonomous vehicle, on-road driving, traffic simulation, performance metrics, PRIDE*

1.0 Introduction/Problem Statement

The field of autonomous systems is continuing to gain traction both with researchers and practitioners. Funding for research is 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 [10]. 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 the DEMO-III XUV has only been tested in static environments. It has not been tested in on-road driving situations, which include pedestrians and oncoming traffic.

There have been experiments performed with autonomous vehicles during on-road navigation. Perhaps the most successful has been that of Dickmanns [4] as part of the European Prometheus project in which the autonomous vehicle performed a trip from Munich to Odense (over 1,600 kilometers) at a maximum velocity of 180 km/hr. 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 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, we often have expectations of how each object in the environment is expected to move based upon the situation. 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 a function of the assumptions we make about the behavior of other vehicles.

To date, the authors are not aware of any autonomous vehicle efforts that account for this information when performing path planning. To address this need, we have developed a 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 [8]. In this paper, we describe how we leveraged the algorithms in the PRIDE framework to simulate traffic patterns during on-road driving. We can then use these simulated traffic patterns to control vehicles in the environment in an on-road driving arena being developed at NIST [9] in a way to assess the performance of autonomous vehicles being tested within these arenas.

In Section 2, we survey some related work in traffic simulation. In Section 3, we describe the various components of the PRIDE framework. In Section 4, we show how the PRIDE framework can be applied to traffic simulation for on-road driving. In Section 5, we explain how this traffic simulation can be used to apply

performance metrics for autonomous vehicles. Section 6 concludes the paper.

2. RELATED WORK

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 reasons of certain dysfunctions.

There have been a few efforts that have tried to simulate traffic patterns. One of more prominent ones in the literature is ARCHISM [3,5], but even this effort is based upon driving psychology studies. These traffic simulations use laws that can be applied for a specific environment or a specific situation. Some of those postulates can be expand to generic situations but still attached to a kind of situation.

3. THE PRIDE FRAMEWORK

We are using the 4D/RCS (Real-Time Control System) reference model architecture [1] as the basis in which to apply the representational approaches that are being developed in this effort. 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. 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.

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, 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. Estimation-theoretic schemes using Kalman Filters (KFs) are well established recursive state estimation techniques where estimates the states of a system are computed using the process and observation models [6]. The recursive nature of the algorithm utilizes the system's CPU more uniformly to provide estimates without the latency resulting from batch processing techniques. The (linear) KF is simply a recursive estimation algorithm that provides minimum mean squared estimates (MMSE) of the states of a linear system utilizing knowledge about the process and measurement dynamics, process and measurement noise statistics subject to Gaussian assumptions and initial condition information. When these assumptions are satisfied, the estimates provided by the Kalman filter are *optimal*. The extension of the linear Kalman filtering ideas to a non-linear system is termed extended Kalman filtering.

The Extended Kalman Filter (EKF) is a linear estimator for a non-linear system obtained by *linearization* of the nonlinear state and observation equations. For any non-linear system, the EKF is the best linear unbiased estimator with respect to minimum mean squared error criteria. Within the on-road driving hierarchy, short-term prediction of objects moving at variable speeds and at given look-ahead time instants are predicted using the EKF. More information about this approach can be found in [7].

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. More information about this approach is included in the follow section.

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.

Both of these prediction methods have been implemented in two different simulation environments. The EKF approach has been implemented in the OneSaf testbed (www.onesaf.com). OneSaf is a composable, next generation computer generated forces (CGF) that represents a full range of operations, systems, and control process from individual combatant and platform to battalion level, with a variable level of fidelity. OneSaf is able to represent moving objects and provide the object's location and velocity at any point in time, through custom-developed Application Programmers' Interface (API) calls. A user-interface was built on top of OneSaf to display the predicted locations of the moving objects.

In Figure 1, the triangle represents the moving object whose future location is to be predicted. The large circle in front of the triangle is the area in which we are 99 % confident that the object will be in two seconds and the small shaded circle is the area in which we are 50 % confident that the object will be in two seconds. For our implementation, we found that the EKF provided reasonable predictions within a two second horizon. A horizon greater than two seconds introduced too much uncertainty to be useful for our autonomous driving scenarios.

The situation-based probabilistic prediction approach has been implemented in the AutoSim simulation package developed by Advanced Technology Research Corporation[i]. AutoSim is a high-fidelity simulation tool which models details about road networks, including individual lanes, lane markings, intersections, legal intersection traversibility, etc. Using this package, we have simulated typical traffic situations (e.g., multiple cars negotiating around obstacles in the roadway, bi-directional opposing traffic, etc.) and have predicted the future location of individual vehicles on the roadway based upon the prediction of where other vehicles are expected to be.

At the point this paper was written, we have simulated a handful of 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 rightmost lane, 5) rapidly accelerating or decelerating, and 6) changing lanes where double yellow lines in the road exist, among other costs.

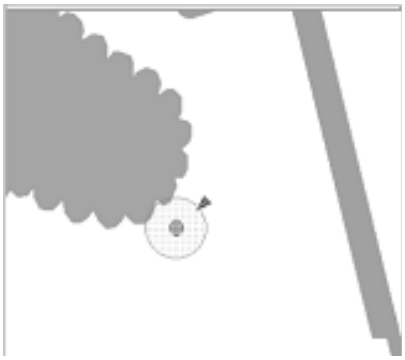


Figure 1: Short-term Prediction.

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 personality 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. A snapshot of the implementation is shown in Figure 2 (a) and (b).

4. APPLYING THE PRIDE FRAMEWORK TO TRAFFIC SIMULATION

Although the PRIDE framework was originally developed to inform a planner about the future position of moving

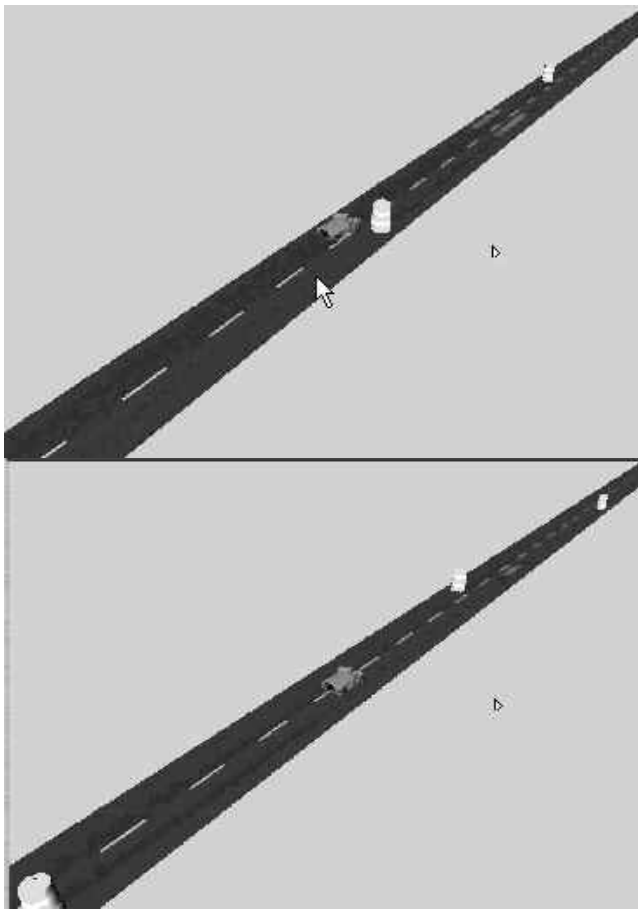


Figure 2: Situation-based probablistic prediction. (a) above and (b) below show a vehicle performing a passing operation around stationary obstacles.

objects for the purpose of path planning and collision avoidance, we have found that the same set of algorithms could be applied to simulating traffic patterns during on-road driving. More specifically, we applied the situation recognition and probabilistic algorithms to determine the likely actions that a vehicle in the environment would take when confronted with a specific situation, and then command that vehicle to perform that action. By doing this with multiple vehicles, we are able to simulate fairly sophisticated traffic situations in which vehicles behave in a way that is very similar to how a human would behave. Vehicles will slow down and/or pass when approaching a stopped or slow object in their lane, they will typically only change lanes when the next lane is clear and they are going slower than desired, they will keep a safe follow distance, etc. By providing realistic simulations of traffic situations, we are able to test the autonomous vehicle during realistic on-road driving situations, without having to place the vehicle on a potentially dangerous city street or highway.

The vision is that we will use these algorithms originally in simulated environments (such as the OneSaf and AutoSim simulation environments discussed above) to test out the planning algorithms in the presence of moving objects. Then, when the NIST On-Road Driving Arenas (which are described in another paper in this conference), are completed, these algorithms will be used to control "environmental" vehicles in the arena to simulate on-road traffic.

The remainder of this section describes the details of how the situation recognition and probabilistic algorithms are used to simulate on-road traffic.

The basic assumption behind this situation-based probabilistic prediction approach is a driver's behavior can be quantified using costs. In general, a driver will prefer an action that minimizes its cost. With this assumption, the cost can be converted to probabilities, where the higher the cost, the lower the probability that the driver will execute that action.

4.1. POSSIBLE VEHICLE ACTIONS

For the purpose of the algorithms, we have discretized the possible actions that a vehicle can make at any given time. Note that for this exercise, we have not accounted for intersections. All of the examples occur on a continuous stretch of roadway.

A vehicle can execute two types of actions. The first type of action is regarding its velocity, namely, quick acceleration (QA), slow acceleration (SA), constant velocity (CV), slow deceleration (SD) and quick deceleration (QD).

The second type of action is regarding lane changing, namely, changing to the left lane (CL), staying in the same lane (SL) and changing to the right lane (CR). These are shown in Figure 3.

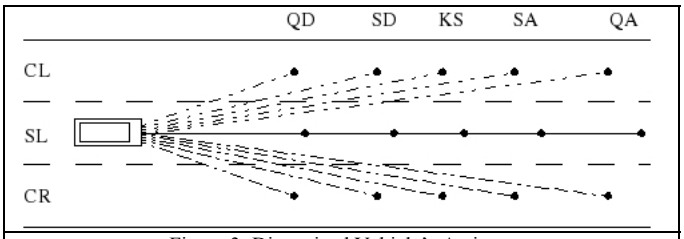


Figure 3: Discretized Vehicle's Actions.

As shown, at any point in time, the vehicle can have up to 15 possible future actions.

4.2. THE PRIDE ALGORITHMS

In this section, we will use the following scenario as shown in Figure 4 to explain the concepts in the algorithms. This scenario is composed by three vehicles, two (A and B) on the same lane (L1) and another one (C) on the opposite lane (L2) and a static obstacle (D) on the lane L1.

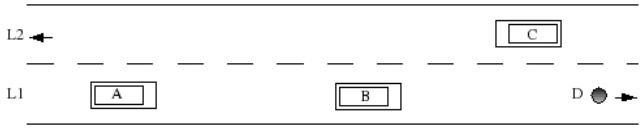
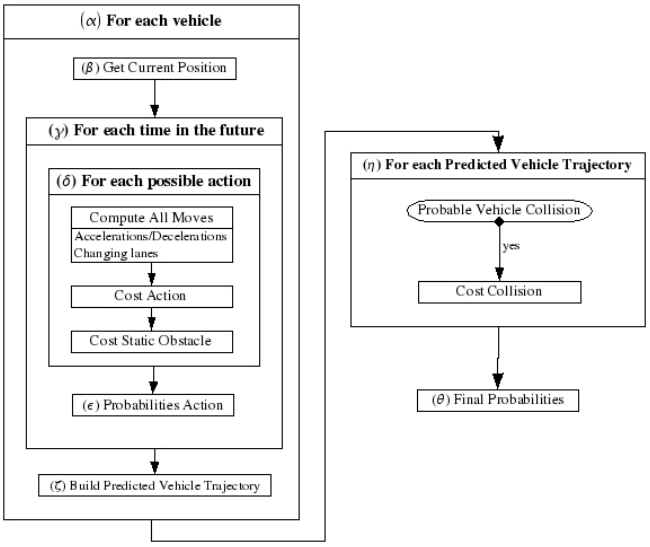


Figure 4: Scenario

Figure 5 shows the overall process for the algorithms (a) graphically and (b) in pseudo-code. The algorithm proceeds as follows:

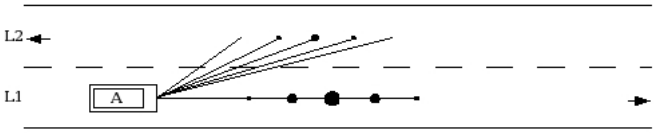
1. For each vehicle on the road (α), the algorithm gets the current position and velocity from external programs/sensors (β).
2. For each time in the future and for each starting position (χ), the algorithm creates a set of next possible positions in the future and assigns a cost to each of them corresponding to the action performed and the state occupied. At the end of the first prediction, each ending position is set at the starting positions at time *t* are used as "starting position" to build the next set of future position for time t+1. This loop (δ) is performed t_{final} iterations, where t_{final} is the predetermined time in the future that we wish to predict.
3. Using the costs found in step 2, the algorithm computes the probability for each movements of the vehicle (ε). In Figure 6, ten possible positions are shown for the vehicle (a) A, (b) B, and (c) C at time t_{final}.
4. The width of each dot in Figure 6 represents the relative probability of each action with respect to the others, where the bigger the dot, the higher the probability.




```
Begin
  Loop
    For each vehicle (α)
      Get Current Position (β)
      For each time in the future (χ)
        For each possible actions (δ)
          Compute All Moves
          Calculate Cost Action
          Calculate Cost Static Obstacle
        End for
        Calculate Probabilities Action (ε)
      End for
      Build Predicted Vehicle Trajectory (ξ)
    End for
    For each Predicted Vehicle Trajectory (η)
      If Probable Vehicle Collision
        Then Calculate Cost Collision
      End if
    End for
    Calculate Final Probabilities (θ)
  End Loop
End.
```

Figure 5 (a) (above) schematic of overall process, (b) (below) pseudo code

5. The algorithm then builds the Predicted Vehicle Trajectories (ξ) which will be used to evaluate the possibly of colliding with another vehicle. The Predicted Vehicle Trajectory notion is explained later in this document.
6. For each pair of Predicted Vehicle Trajectory (η), the algorithm checks if there is a probable collision and assigns a cost to the collision.
7. In the example, for the vehicles A and C, Figure 7 shows that one possible trajectory for A intersects a possible trajectory for C, thus resulting in a collision.



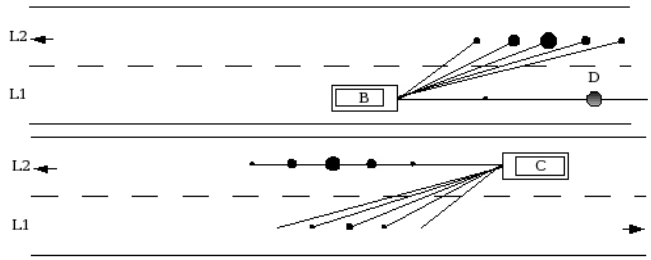


Figure 6: Predicted Positions of (a) (top) Vehicle A, (b) (middle) Vehicle B, and (c) (bottom) Vehicle C.

8. The probabilities (θ) based on the costs of each action are recalculated based upon the collision information. Figure 8 shows the final vehicle probabilities for Vehicles A, B, and C. Notice how the size of the dots have changed from Figure 5, thus accounting for the cost due to possible collisions.

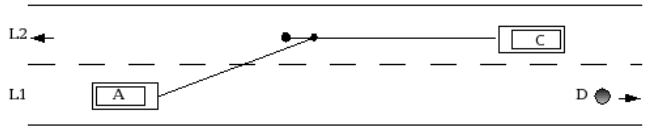


Figure 7: Possible Collision Between A and C

At the end of the main loop, the path with the highest probability for each vehicle represents the most likely location where the vehicles will be in t_{final} in the future.

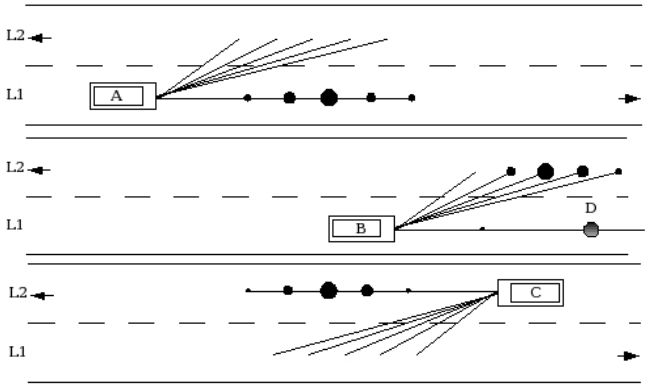


Figure 8: Final Vehicle Probabilities for (a) Vehicle A, (b) Vehicle B, and (c) Vehicle C.

4.3. COST MODEL AND PROBABILITY

The moving object prediction (MOP) algorithms can be separated in two parts, the first one is the creation of a set a predicted positions for each vehicle and the second is the evaluation of the interaction between each vehicle on the road. Every evaluation is based on costs that are converted to probabilities.

The Cost Model (CM) assigns danger ratings (cost) to each action that a vehicle can perform and the states that it

occupies after performing that action. This exact cost assigned to each action and state depends upon the aggressivity of the driver.

The cost of an action is the sum of the costs that are encountered by performing that action, which could include a cost for changing lanes, a cost for accelerating, a cost for going over the speed limit, etc. The cost of the state is described in the next section. This approach for building costs is based upon work performed in [2].

4.4. PREDICTED VEHICLE TRAJECTORY

The Predicted Vehicle Trajectory (PVT) represents the possible movements of a vehicle throughout the time period being analyzed. The PVT is a representation of the trajectory, as shown in Figure 9.

Figure 9: Predicted Vehicle Trajectory.

The PVT is built with the origin position $\{x_{IP}, y_{IP}, t_{IP} = 0\}$ at time = 0 and the predicted position $\{x_{PP}, y_{PP}, t_{PP} = t_{final}\}$ where t_{final} is the predetermined time in the future for the prediction process. It also contains the action-cost and action-probability information.

The PVT is used to determine possible collisions. Because the PVT represents a trajectory of one vehicle (origin to predicted), we can determine possible collision between any two vehicles by determining if two PVTs cross.

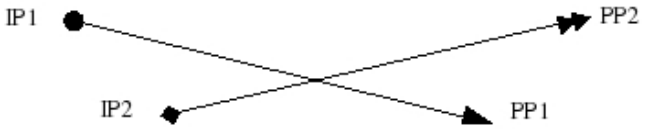


Figure 10: Two PVTs Crossing.

When two PVTs are crossing each other (Figure 10), it is important to know where are they crossing. This information can be obtained by using a parametrization of each PVT.

The parametrization:

$$\begin{aligned} x_1(t_1) &= x_{PP1}t_1 + x_{IP1}(1-t_1) \\ y_1(t_1) &= y_{PP1}t_1 + y_{IP1}(1-t_1) \end{aligned} \text{ where } t_1 \in [0,1]$$

$$\begin{aligned} x_2(t_2) &= x_{PP2}t_2 + x_{IP2}(1-t_2) \\ y_2(t_2) &= y_{PP2}t_2 + y_{IP2}(1-t_2) \end{aligned} \text{ where } t_2 \in [0,1]$$

where t_1 and t_2 are the parameters of each PVT.

By using the Theorem of Cramer, t_1 and t_2 can be determined:

$$\begin{aligned} x_{PP1}t_1 + x_{IP1}(1-t_1) &= x_{PP2}t_2 + x_{IP2}(1-t_2) \\ y_{PP1}t_1 + y_{IP1}(1-t_1) &= y_{PP2}t_2 + y_{IP2}(1-t_2) \\ (x_{PP1} - x_{IP1})t_1 + (x_{IP2} - x_{PP2})t_2 &= x_{IP2} - x_{IP1} \\ (y_{PP1} - y_{IP1})t_1 + (y_{IP2} - y_{PP2})t_2 &= y_{IP2} - y_{IP1} \end{aligned}$$

$$t_1 = \frac{\begin{vmatrix} x_{IP2} - x_{IP1} & x_{IP2} - x_{PP2} \\ y_{IP2} - y_{IP1} & y_{IP2} - y_{PP2} \end{vmatrix}}{\begin{vmatrix} x_{PP1} - x_{IP1} & x_{IP2} - x_{PP2} \\ y_{PP1} - y_{IP1} & y_{IP2} - y_{PP2} \end{vmatrix}}$$

$$t_2 = \frac{\begin{vmatrix} x_{PP1} - x_{IP1} & x_{IP2} - x_{IP1} \\ y_{PP1} - y_{IP1} & y_{IP2} - y_{IP1} \end{vmatrix}}{\begin{vmatrix} x_{PP1} - x_{IP1} & x_{IP2} - x_{PP2} \\ y_{PP1} - y_{IP1} & y_{IP2} - y_{PP2} \end{vmatrix}}$$

So the two vehicles will cross each other at two different times $t_1 t_{final}$ for the first vehicle and $t_2 t_{final}$ for the second one. For a small difference between the two time, the collision is highly likely. Conversely, if the difference is large collision is improbable.

The inverse of the difference between the two collision times represents the coefficient of the collision cost:

$$CollisionCost = \frac{M}{t_{final}\left|t_1 - t_2\right|}$$

where M is a preset of the Cost Model for collision costs.

5. APPLYING TRAFFIC SIMULATION TO PERFORMANCE METRICS

Now that we've described how we can simulate traffic patterns, we will discuss how this could be used to associate performance metrics to an autonomous vehicle. In evaluating how an autonomous vehicle is performing during on-road driving, we need the ability to test that vehicle in various driving situations. Those situations could be a function of the environment (e.g., winding roads, steep slopes, traffic signals, intersections), weather conditions (e.g., rain, fog, ice on the roadway), and static and dynamic objects in the environment (e.g., traffic barrels, pedestrians, other vehicles). The traffic simulator allows us the ability to dynamically change information about static and dynamic objects in the environment in order to introduce a variety of situations that we can evaluate the autonomous vehicle against.

The traffic simulator allows one to have repeatable, realistic traffic patterns that only vary in response to the autonomous vehicle motions. As such, one would be able to place two different autonomous vehicles in an identical traffic environment to evaluate how each performs. If the two autonomous vehicles behaved in identical fashion, the entire flow of traffic would be identical. Conversely, if the two autonomous vehicles' behaviors differed in any way, the flow of traffic would most likely differ (since other vehicles in the traffic pattern may be reacting to the actions of the autonomous vehicle). Metrics could be assigned to the autonomous vehicle's actions, based on a number of criteria, including proximity to other vehicles, staying within the speed limit, number of lane changes, obeying traffic signs and signals, etc.

The traffic simulator also allows for the ability to vary the aggressivity of drivers on the road. Using this capability, we could have different difficulties of driving scenarios, based, in part, on the aggressivity of the drivers on the road, where the higher the aggressivity, the harder the course. One could even imagine situations where an accident among a pair of traffic vehicles is imminent. The autonomous vehicle could then be evaluated based upon its ability to predict this accident and take precautions in time.

6. CONCLUSION

In this paper, we have described the PRIDE framework, which was developed to perform moving object prediction for unmanned ground vehicles. PRIDE is based upon a multi-resolutional, hierarchical approach that incorporates multiple prediction algorithms into a single, unifying framework. The lower levels of the framework (not discussed in detail in the paper) utilize estimation-theoretic short-term predictions while the upper levels utilize a probabilistic prediction approach based on situation recognition with an underlying cost model. We showed the results achieved from applying PRIDE in a simulated environment.

We have then shown how PRIDE can be extended to simulate traffic patterns during on-road navigation. The PRIDE algorithms are used to provide the underlying logic to control vehicles in the environment, thus generating a realistic flow of traffic. We use the prediction algorithms within PRIDE to determine the probability that a vehicle will exhibit a certain behavior given a set of environmental conditions, and then command the vehicle to perform the action which has the greatest probability. By doing this, we are able to create realistic, repeatable, and non-scripted traffic patterns that closely mimic the types of traffic flow expected to be encountered during on-road driving.

We then explore how the PRIDE-based traffic control algorithms can be applied to performance evaluation of autonomous vehicles. Through the use of repeatable and realistic traffic flow simulation, one is able to evaluate the performance of an autonomous vehicle in an on-road driving scenario without the risk involved with introducing the vehicle into a potentially dangerous roadway situation. In addition, by varying a single vehicle's parameters (e.g. aggressivity, speed, location) with the traffic flow, we can show how the entire traffic pattern is affected.

The goal of this paper was to describe how the work to date in moving object prediction could contribute to performance metrics for autonomous systems. Though the algorithms described in this paper have been developed and implemented, there is still much work to be accomplished. For one, the metrics that should be applied when evaluating an autonomous system in the presence of traffic situations being generated by these algorithms need to be determined. This will be the topic of future work and will be realized in the NIST On-Road Driving Arenas mentioned earlier in the paper. Also, these set of algorithms have been shown to work successfully on straight roads, but have not been fully tested on intersections. This will also be a topic of future research.

REFERENCES

1. Albus, J. and et.al., "4D/RCS Version 2.0: A Reference Model Architecture for Unmanned Vehicle Systems," NISTIR 6910, National Institute of Standards and Technology, Gaithersburg, MD, 2002.
2. Balakirsky, S., *A Framework for Planning with Incrementally Created Graphs in Attributed Problem Spaces*, IOS Press, Berlin, 2003.
3. Champion, A., Espie, S., and Auberlet, J., "Behavioral Road Traffic Simulation with ARCHISM," *Proceedings of the Summer Computer Simulation Conference, USA*, 2001.
4. Dickmanns, E. D., "An Expectation-Based Multi-Focal Saccadic (EMS) Vision System for Vehicle Guidance," *Proceedings of the 9th International*

Symposium on Robotics Research (ISRR'99), Salt Lake City, 1999.

5. Espie, S., Saad, F., and Schnetler, B., "Microscopic traffic simulation and driver behaviro modeling: the ARCHISM project," *Proceedings of the Strategic Highway Research Program and Traffic Safety on Two Continents*, Lille, France, 1994.
6. Kalman, R., "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME Journal of Basic Engineering*, Vol. 82, No. Series D, 1960, pp. 35-45.
7. Madhavan, R. and Schlenoff, C., "The Effect of Process Models on Short-term Prediction of Moving Objects for Unmanned Ground Vehicles," *Submitted to the 7th International IEEE Conference on Intelligent Transportation Systems*, Washington DC, 2004.
8. Schlenoff, C., Madhavan, R., and Barbera, T., "A Hierarchical, Multi-Resolutional Moving Object Prediction Approach for Autonomous On-Road Driving," *Proceedings of the 2004 ICRA Conference*, 2004.
9. Scrapper, C., Balakirsky, S., and Weiss, B., "Autonomous Road Driving Arenas for Perfomance Evaluation," *To be published in the Proceedings of the Perfomance Metrics for Autonomous Systems (PerMIS) 2004 workshop*, 2004.
10. Shoemaker, C. and Bornstein, J. A., "Overview of the Demo III UGV Program," *Proceedings of the SPIE Robotic and Semi-Robotic Ground Vehicle Technology Conference*, Vol. 3366, 1998, pp. 202-211.

---

[i] Certain software tools are identified in this paper in order to explain our research. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the software tools identified are necessarily the best available for the purpose.