The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Contingency Planning over Probabilistic Hybrid Obstacle Predictions for Autonomous Road Vehicles

Jason Hardy and Mark Campbell

Abstract— **This paper presents a novel optimization based path planner that can simultaneously plan multiple contingency paths to account for the uncertain actions of dynamic obstacles. This planner addresses the particular problem of collision avoidance for autonomous road vehicles which are required to safely interact with other vehicles with unknown intentions. The presented path planner utilizes an efficient spline based trajectory representation and fast but accurate collision probability approximations to enable the simultaneous optimization of multiple contingency paths.**

I. INTRODUCTION

The 2007 DARPA Urban Challenge (DUC) competition demonstrated that current autonomous vehicles are adept at identifying obstacles, planning routes, and interacting in controlled environments with well defined rules [1]. It also demonstrated that there remains a large gap in problem solving and decision making capabilities between autonomous vehicles and human drivers. A major reason for this gap is the inability of current perception and planning algorithms to adequately identify and predict obstacle intent [2]. Inferring obstacle intent enables an autonomous vehicle to model the future motion of dynamic obstacles. These obstacle predictions can then be used to improve the safety and robustness of the robot's trajectory planning.

Predicting future obstacle motion is a problem that has been explored in a variety of fields and contexts. Kushleyev and Likhachev [3] perform predictions of dynamic obstacles using constant velocity and constant curvature dynamics models. This approach is computationally efficient but ignores structural information in the environment. Bennewitz et al. [4] model human trajectories through an environment using observed trajectories as training data. This approach can produce detailed probabilistic trajectories but requires retraining for each new environment and obstacle type. Miura and Shirai [5] infer the potential paths an obstacle might take using a tangent graph of the environment and a 1D model of velocity variance along each potential path. Hwang and Seah [6] present an algorithm for inferring the intent of other airplanes based on likely flight plans, for probabilistically modeling their future motion, and for utilizing this information to detect potential future conflicts. During the DUC, the Tartan Racing team showed that it is possible to deduce a small set of short term goal hypotheses for dynamic obstacles based on the surrounding road configuration [7]. The simulations presented in [7] and the success of Tartan Racing's BOSS robot in the DUC make a compelling case for the inclusion of inferred obstacle intent in the planning process.

The approach presented in this paper is to form goal hypotheses for each dynamic obstacle as in [7] and predict these obstacles' state distributions forward in time using a probabilistic motion model. Based on the work and lessons learned from Team Cornell's entry in the 2007 DUC [8] [9], a new path optimization algorithm is proposed which intelligently incorporates probabilistic obstacle predictions to enable safe, efficient trajectory generation in dynamic environments. The computational efficiency required for solving this problem is achieved by 1) framing the planning problem as a constrained numerical optimization problem, 2) solving the planning problem over a limited planning horizon, 3) adopting an efficient spline based trajectory representation, and 4) using fast but accurate collision probability approximations. Spline based trajectory representations have been used before, such as in [10]; here they are employed in a unique way to enable multiple contingency trajectories to be efficiently optimized in unison. Also, while limiting the planning horizon prevents true global planning, it is an effective navigation strategy when used in conjunction with a higher level global route planner [8].

Department of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY, USA jsh256@cornell.edu

II. OBSTACLE PREDICTION

It is assumed in this paper that the problem of identifying unique dynamic obstacles, tracking these obstacles between sensor measurements, and estimating the state of these obstacles has been resolved to a sufficient degree using an external tracking algorithm, such that current Gaussian state estimates ((μ^i, Σ^i) for the i^{th} of n_o dynamic obstacles) are known at the beginning of each planning cycle. Each of these obstacle estimates is then used as an initial distribution for the obstacle prediction algorithm. A set of $\{n_g\}^i$ goal states, $g^i_{1:n_g}$, is inferred for each obstacle based on a known topological representation of the local road network. For example, a dynamic obstacle approaching a four way intersection might have $g \in \{\text{stop}, \text{go straight}, \text{turn left}, \text{turn right}\}$. For multiple dynamic obstacles, G_j represents the j^{th} set of possible goal combinations for the n_o obstacles, $G_j = \{g^1_{(.)}, \ldots, g^{n_o}_{(.)}\}_j$, where $\{g^i_{(.)}\}_j$ is the inferred goal state included in G_j for obstacle i. A total of $n_s = \prod_{i=1}^{n_o}\{n_g\}^i$ goal sets are available.

The output of the obstacle prediction algorithm, o_j, is a sequence of predicted obstacle state distributions for n_o dynamic obstacles over N timesteps for a given goal state

set, G_j:

$$o_j = \{p(G_j), (\hat{\mu}_j^1, \hat{\Sigma}_j^1)_{k:k+N}, \ldots, (\hat{\mu}_j^{n_o}, \hat{\Sigma}_j^{n_o})_{k:k+N}\} \quad (1)$$

where $p(G_j)$ is the probability of G_j being correct.

The motion of each obstacle is predicted forward in time using a probabilistic motion model. For linear motion models, this prediction can be performed using the prediction step of the standard Kalman filter. For more general nonlinear, non-differentiable motion models, this prediction can be performed using an algorithm such as the Unscented Transform [11].

At the start of each new planning cycle updated state estimates for each obstacle, $\{(\mu^1, \Sigma^1)_{k+1}, \ldots, (\mu^{n_o}, \Sigma^{n_o})_{k+1}\}$, are obtained from the tracking algorithm. These new state estimates are used to update the probability of each goal set, $p(G_j)$:

$$p(G_j)_{k+1} = \frac{p(G_j)_k \prod_{i=1}^{n_o} p((\mu^i, \Sigma^i)_{k+1} | (\hat{\mu}_j^i, \hat{\Sigma}_j^i)_{k+1})}{\sum_{l=1}^{n_s} p(G_l)_k \prod_{i=1}^{n_o} p((\mu^i, \Sigma^i)_{k+1} | (\hat{\mu}_l^i, \hat{\Sigma}_l^i)_{k+1})} \quad (2)$$

where $p((\mu^i, \Sigma^i)_{k+1} | (\hat{\mu}_j^i, \hat{\Sigma}_j^i)_{k+1})$ is the innovation likelihood of the updated state estimate for obstacle i, $(\mu^i, \Sigma^i)_{k+1}$, given the predicted state distribution from the previous timestep, $(\hat{\mu}_j^i, \hat{\Sigma}_j^i)_{k+1}$, using the goal specified in goal set G_j.

III. PATH OPTIMIZATION

The path planning problem is formulated here as a general optimization problem:

$$\mathbf{h}_{\text{opt}} = \min_{\mathbf{h}} J(\mathbf{h}) \quad (3)$$
$$C(\mathbf{h}) < 0$$

where $\mathbf{h}$ is a parameter vector that defines a trajectory or set of contingency trajectories for the robot and $J(\mathbf{h})$ is a predefined cost function over which the path is optimized. The constraint function, $C(\mathbf{h})$, places hard constraints on the range of possible values that $\mathbf{h}$ can take.

In order to accurately handle the hybrid nature of the obstacle predictions, it is necessary to plan a separate contingency path for each obstacle goal set, G_j. However, since the robot cannot follow multiple contingency paths at the same time, the initial segment of each contingency path is constrained to be the same.

This is preferable to alternate approaches, such as planning a single path that attempts to avoid all obstacle predictions simultaneously, or taking a weighted combination of independent contingency paths. Planning a single path is overcautious and ignores the fact that the obstacle goal sets are mutually exclusive, while taking a weighted combination offers no guarantees on the safety of the combined path. Sharing the initial segment of each contingency path gives the robot a deterministic path to execute at the current time step while maintaining the independence of future contingencies.

A. Trajectory Representation

To make the optimization problem defined in Equation 3 efficient enough for realtime replanning, a path representation is used that is both expressive in its ability to cover the search space and compact in its dimensionality. Cubic splines are chosen for this purpose since two splines, $y(t)$ and $x(t)$, define a continuous representation of the robot's position, (x, y), and its derivatives, (v^x, v^y, a^x, a^y), as a function of time, t, using only a small set of control points, h, as variables. The optimization vector, $\mathbf{h}$, is a set of parameters defining n_s contingency trajectories with a shared initial segment:

$$\mathbf{h} = \{h_1^x, h_{2:n}^{1,x}, ..., h_{2:n}^{n_s,x}, h_1^y, h_{2:n}^{1,y}, ..., h_{2:n}^{n_s,y}\} \quad (4)$$

where n is the number of cubic line segments in each contingency path. The shared initial segment constraint is enforced by defining the initial segment as an independent cubic line segment. The initial conditions of the robot define the boundary conditions for one side of this initial segment and a zero-curvature end condition and the height of the first control point, $h_1^{(\cdot)}$, define the remaining boundary conditions. The first control point, $h_1^{(\cdot)}$, represents the end of the initial shared segment and a separate cubic spline is defined for each contingency path starting from this point. Figure 1 depicts

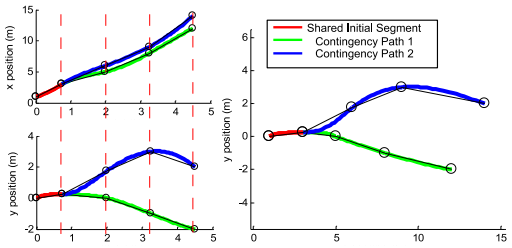


Fig. 1. Left: depictions of the temporal $x(t)$ and $y(t)$ path representations. Right: a visualization of the path representations in physical space.

a cubic spline representation for multiple contingency paths. To reduce dimensionality, the control points on the spline are constrained to lie on fixed time intervals, as indicated by the dashed vertical lines. Using this cubic spline definition, the robot's position and its derivatives at a given time t depend linearly on the control point heights $\mathbf{h}$.

B. Collision Probability

The problem of calculating collision probabilities with uncertain dynamic obstacles has been studied in a range of fields including air traffic control [12] and satellite collision avoidance [13]. For autonomous road vehicles, the collision probability calculation is especially challenging because road vehicles are expected to interact closely with dynamic obstacles, such as driving in adjacent lanes. This limits the level of over approximation that is acceptable.

1) Circular Approximation: The true instantaneous probability of collision between the robot and a predicted obstacle is calculated by integrating the covariance of the obstacle

position distribution, P_{obst}, over the combined body, CB, of the obstacle and robot shapes. This CB is formed by sweeping the shape of the obstacle around the perimeter of the robot shape and recording the outline traced by its origin. The collision probability calculation can be simplified by approximating the shape of both the obstacle and the robot as circles. The CB in this case is simply a circle with radius $r_{\text{CB}} = r_{\text{robot}} + r_{\text{obst}}$. The probability of the obstacle being in a collision state with the robot is equivalent to the probability of a point sample from the obstacle's position distribution lying within the CB centered at the robot's relative location.

To quickly approximate the instantaneous collision probability between two circular bodies, a rotation transformation, R, is applied that aligns the major and minor axes of the obstacle position covariance ellipse with the coordinate axes of the reference frame:

$$\begin{aligned} P_{\text{obst}} &= \begin{bmatrix} a & b \\ b & c \end{bmatrix} \\ \phi &= \frac{1}{2} \cdot \tan\left(\frac{2b}{c-a}\right) \\ R &= \begin{bmatrix} \cos(-\phi) & \sin(-\phi) \\ -\sin(-\phi) & \cos(-\phi) \end{bmatrix} \end{aligned} \tag{5}$$

where ϕ is the angle by which P_{obst} is rotated in the current coordinate system. This alignment allows the 2D Gaussian obstacle position distribution to be decoupled into the product of uncorrelated 1D Gaussian distributions along the coordinate axes:

$$\begin{aligned} P^R_{\text{obst}} &= \begin{bmatrix} (\sigma^x_{\text{obst}})^2 & 0 \\ 0 & (\sigma^y_{\text{obst}})^2 \end{bmatrix} \\ \mathcal{N}(\mathbf{x}^R_{\text{obst}}, P^R_{\text{obst}}) &= \mathcal{N}(x^R_{\text{obst}}, (\sigma^x_{\text{obst}})^2)\mathcal{N}(y^R_{\text{obst}}, (\sigma^y_{\text{obst}})^2) \end{aligned} \tag{6}$$

where P^R_{obst} and $\mathbf{x}^R_{\text{obst}}$ are the obstacle covariance matrix and position vector transformed by the R.

As proposed in [6], decoupling the obstacle position distribution allows the probability of any rectangular region aligned with the transformed coordinate frame to be integrated over using the 1D Gaussian cumulative density function $\Psi(z; \mu, \sigma)$. Since the CB in this case is circular, its shape is unaffected by the coordinate transformation R and can be over approximated using a square bounding box. The collision probability approximation is then computed as:

$$p^{\text{circ}}_{\text{coll}}(\mathbf{x}_{\text{robot}}, \mathbf{x}_{\text{obst}}, P_{\text{obst}}) = \int_{\text{CB}} \mathcal{N}(\Delta\mathbf{x}^R, P^R_{\text{obst}}) \tag{7}$$

$$\begin{aligned} p^{\text{circ}}_{\text{coll}} = (&\Psi(\Delta x^R + r_{CB}; 0, \sigma_x) - \Psi(\Delta x^R - r_{CB}; 0, \sigma_x) \\ &\cdot(\Psi(\Delta y^R + r_{CB}; 0, \sigma_y) - \Psi(\Delta y^R - r_{CB}; 0, \sigma_y) \end{aligned}$$

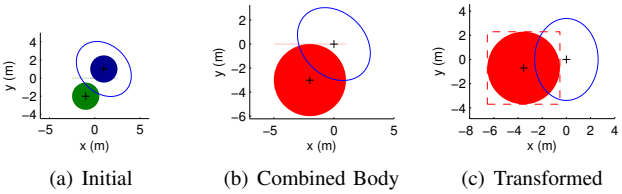
where $\Delta\mathbf{x}^R = \mathbf{x}^R_{\text{obst}} - \mathbf{x}^R_{\text{robot}}$, $\Delta x^R = x^R_{\text{obst}} - x^R_{\text{robot}}$, and $\Delta y^R = y^R_{\text{obst}} - y^R_{\text{robot}}$. The steps of the circular collision probability approximation are depicted in Figure 2.

2) Rectangular Approximation: The circular approximation provides a fast, orientation independent collision probability approximation, but additional accuracy is required for enforcing safety constraints. A more accurate approximation can be achieved by using rectangular robot and obstacle shapes. This complicates the probability calculation because the obstacle's orientation is included as an additional uncertain variable. Assuming the obstacle's orientation is an independent 1D Gaussian distribution, $\gamma \sim \mathcal{N}(\mu_\gamma, \sigma^2_\gamma)$, which is equivalent to ignoring the correlation between the orientation and position elements of the obstacle's state error covariance, an upper bound on the total collision probability can be found by taking a sum over discrete obstacle orientation ranges:

$$\begin{aligned} &p^{\text{rect}}_{\text{coll}}(\mathbf{x}_{\text{robot}}, \mathbf{x}_{\text{obst}}, P_{\text{obst}}, \mu_\gamma, \sigma_\gamma) \leq \\ &\quad (1 - \sum_{l=1}^{n_\gamma} p(\gamma^{l:l+1}_{\text{obst}}))p^{\text{circ}}_{\text{coll}}(\mathbf{x}_{\text{robot}}, \mathbf{x}_{\text{obst}}, P_{\text{obst}}) \\ &\quad + \sum_{l=1}^{n_\gamma} p(\gamma^{l:l+1}_{\text{obst}})p^{\text{rect}}_{\text{coll}}(\mathbf{x}_{\text{robot}}, \mathbf{x}_{\text{obst}}, P_{\text{obst}}, \gamma^{l:l+1}_{\text{obst}}) \end{aligned} \tag{8}$$

where n_γ is the number of discrete angle ranges used to approximate the obstacle orientation distribution, $p(\gamma^{l:l+1}_{\text{obst}})$ is the probability of the obstacle orientation lying between γ^l and γ^{l+1}, and $p^{\text{circ}}_{\text{coll}}(\mathbf{x}_{\text{robot}}, \mathbf{x}_{\text{obst}}, P_{\text{obst}})$ uses the furthest point from the center of the back axle as the radius for both the robot and obstacle vehicles.

Solving the right hand side of Equation 8 requires the computation of $p^{\text{rect}}_{\text{coll}}(\mathbf{x}_{\text{robot}}, \mathbf{x}_{\text{obst}}, P_{\text{obst}}, \gamma^{l:l+1}_{\text{obst}})$, which is the probability of collision between the robot and the obstacle with a fixed orientation range. For the simpler problem of an obstacle rectangle with a single fixed orientation, a CB is formed by sweeping the oriented obstacle rectangle around the perimeter of the robot rectangle as shown in Figure 3. This CB can then be over approximated using a bounding box. A similar procedure can be applied for an



(a) Initial (b) Combined Body (c) Transformed

Fig. 2. Depiction of approximate collision calculation. a) depicts the initial scene. b) depicts the combined body of the obstacle shape and the vehicle shape. c) depicts the effects of the transformation R and the bounding box approximation.

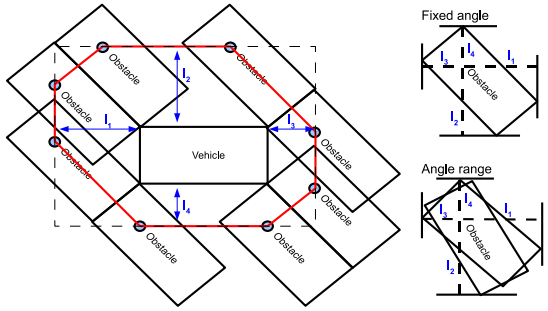


Fig. 3. Formation of a combined body for collision probability between a rectangular robot and a rectangular obstacle.

obstacle rectangle with a fixed orientation range by rotating the obstacle through its range of possible orientations and applying a bounding box to the entire rotation. This bounded obstacle shape can then be swept around the perimeter of the robot rectangle to create a CB.

As in the circular approximation, the obstacle position covariance ellipse must be aligned with the coordinate axes in order to decouple the distribution. However, the CB rectangle must also be aligned with the coordinate axes in order to define a tight rectangular integration region. Paielli et al. [12] provides a method for normalizing the error covariance using a linear coordinate transformation:

$$\begin{aligned} W &= (\sqrt{P_{\text{obst}}})^{-1} \\ WP_{\text{obst}}W^T &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned} \tag{9}$$

This transformation allows the obstacle position distribution to be decoupled in any direction, allowing the coordinate frame to be aligned with the transformed CB, defined as CB^W. Applying this coordinate transformation results in a normalized obstacle covariance matrix and a skewed rectangular CB as shown in Figure 4. This skewed rectangle CB

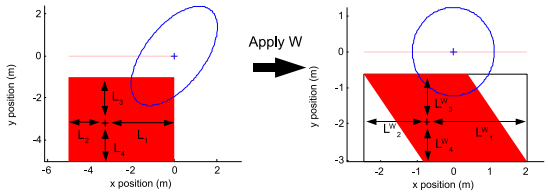


Fig. 4. Coordinate transformation for combined rectangular body.

can be over approximated with another bounding box and an upper bound on the collision probability for the given $\gamma_{\text{obst}}^{l:l+1}$ range can be computed using:

$$\begin{aligned} p_{\text{coll}}^{\text{rect}} \leq (\Psi(\Delta x^W + L_1^W; 0, 1) - \Psi(\Delta x^W - L_2^W; 0, 1) \\ (\Psi(\Delta y^W + L_3^W; 0, 1) - \Psi(\Delta y^W - L_4^W; 0, 1) \end{aligned} \tag{10}$$

where $L_{(\cdot)}$ and $L_{(\cdot)}^W$ are as shown in Figure 4.

This rectangular approximation provides a tighter bound than the circular collision approximation but has a higher computational cost. Figure 5 shows a hypothetical relative trajectory (in a robot centric coordinate frame) for an interaction between the robot vehicle and an obstacle vehicle and provides a comparison for each point on the trajectory between the circular approximation, and two rectangular approximations ($n_\gamma = 1$, $n_\gamma = 5$). For this comparison, Equation 8 is simplified for speed by setting $p_{\text{coll}}^{\text{circ}}(\mathbf{x}_{\text{robot}}, \mathbf{x}_{\text{obst}}, P_{\text{obst}}) = 1$ and $\sum_{l=1}^{n_\gamma} p(\gamma_{\text{obst}}^{l:l+1}) = 0.99$. Figure 5 shows that the rectangular approximation produces a much tighter upper bound than the circular approximation in cases such as point four, where the obstacle vehicle is driving parallel to the robot in an adjacent lane. This is important because it allows for a maximum collision probability threshold that prevents unsafe situations, while still allowing for normal driving behavior. The results also

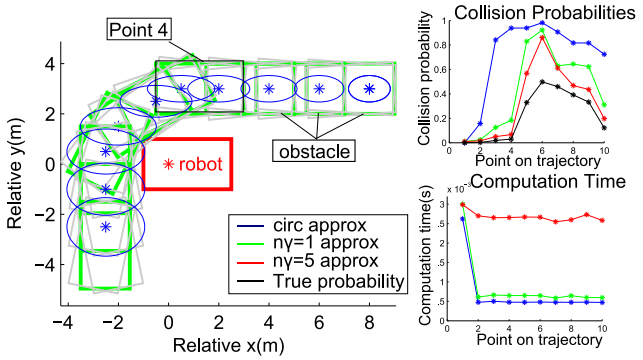


Fig. 5. Comparison of different collision probability approximations.

show that using $n_\gamma = 1$ provides nearly as good of a bound as using $n_\gamma = 5$, but with a computational cost nearly equivalent to that of the circular approximation.

C. Cost Function Definition

The cost function for the path optimization problem is:

$$J(\mathbf{h}) = J_{\text{shared}}(\mathbf{h}, o_{1:n_s}) + \sum_{j=1}^{n_s} p(G_j) J_{\text{conting}}(\mathbf{h}, o_j) \tag{11}$$

where $J_{\text{shared}}(\mathbf{h}, o_{1:n_s})$ is the cost for the shared path segment and $J_{\text{conting}}(\mathbf{h}, o_j)$ are the costs for each contingency path. These costs are defined as a sum over discrete evaluation points; for this paper, Δt_{eval} is set equal to the obstacle prediction time step for convenience. This discrete formulation approximates a continuous integral over the path segments, but allows for simpler pointwise calculations. The cost for the shared segment, $J_{\text{shared}}(\mathbf{h}, o_{1:n_s})$, is:

$$\begin{aligned} J_{\text{shared}}(\mathbf{h}, o_{1:n_s}) = \\ \alpha_a \sum_{k=1}^{n_1} ((a_k^x)^2 + (a_k^y)^2) + \alpha_c \sum_{k=1}^{n_1} (v_k^x a_k^y - v_k^y a_k^x)^2 + \\ \alpha_d \sum_{k=1}^{n_1} f_{\text{prx}}(x_k, y_k) + \alpha_p \sum_{j=1}^{n_s} p(G_j) \sum_{k=1}^{n_1} f_{\text{coll}}^{\text{circ}}(x_k, y_k, o_j) \end{aligned} \tag{12}$$

where n_1 is the number of discrete timesteps evaluated over the initial path segment, f_{prx} is a function penalizing proximity to static obstacles and distance from goal, and $f_{\text{coll}}^{\text{circ}}$ is the circular collision probability. Here $f_{\text{coll}}^{\text{circ}}$ is an under approximation using the width of the robot and obstacle as their radii. This allows for close driving behavior but is insufficient on its own for guaranteeing safety. The cost for each contingency path, $J_{\text{conting}}(\mathbf{h}, o_j)$ is:

$$\begin{aligned} J_{\text{conting}}(\mathbf{h}, o_j) = \\ \alpha_a \sum_{k=1}^{n_N} ((a_k^x)^2 + (a_k^y)^2) + \alpha_c \sum_{k=1}^{n_N} (v_k^x a_k^y - v_k^y a_k^x)^2 + \\ \alpha_d \sum_{k=1}^{n_N} f_{\text{prx}}(x_k, y_k) + \alpha_p \sum_{k=1}^{n_N} f_{\text{coll}}^{\text{circ}}(x_k, y_k, o_j) \end{aligned} \tag{13}$$

where n_N represents the number of discrete timesteps evaluated along each contingency path.

The $\alpha_{(\cdot)}$ parameters in this equation are weights that govern the relative importance of each term: α_a penalizes high acceleration, α_c penalizes unnormalized path curvature, α_d penalizes proximity to static obstacles and distance from goal, and α_p penalizes high collision probabilities with dynamic obstacles.

The acceleration and curvature terms are formulated to be convex with respect to position, (x, y), and its derivatives, (v^x, v^y, a^x, a^y). Unnormalized path curvature, $(v_k^x a_k^y - v_k^y a_k^x)^2$, is used to maintain convexity and is equivalent to weighting the path curvature by the robot's velocity. The collision probability term $f_{\text{coll}}^{\text{circ}}$ represents a cost hill and behaves as a convex function away from its cost peak.

The proximity term, $f_{\text{prx}}(x_k, y_k)$, is the sum of a convex distance to goal function, such as the square of the Euclidean distance to goal, and a static obstacle function that is convex with respect to the robot's lateral offset from the center of its obstacle free driving corridor. Typically, this corridor center is simply the center of the road lane. For more general definitions, this corridor center might be based on a Voronoi decomposition or cell decomposition of the robot's static environment.

In order to ensure the optimized path obeys the physical constraints of the robot and stays below a required probability of collision with dynamic obstacles, a set of constraint functions, $C(\mathbf{h})$, are evaluated at each timestep k along the shared initial segment and along each contingency path:

$$-(v_{k-1}^x v_k^x - v_{k-1}^y v_k^y) - \beta_{\max} ||\mathbf{v}_{k-1}|| \cdot ||\mathbf{v}_k|| < 0 \quad (14)$$

$$(v_k^x)^2 + (v_k^y)^2 - v_{\max}^2 < 0 \quad (15)$$

$$f_{\text{static}}(x_k, y_k, \theta_{\text{robot},k}) < 0 \quad (16)$$

$$p(G_j) f_{\text{coll}}^{\text{rect}}(x_k, y_k, \theta_{\text{robot},k}, o_j) - p_{\max} < 0 \quad (17)$$

Equation 14 constrains changes in angle between successive timesteps, where $\beta_{\max}$ controls the maximum allowed change in angle ($\beta_{\max} = 0$ equates to $\Delta\theta_{\max} = 90°$). Equation 15 enforces a maximum velocity constraint. Equation 16 constrains the robot's configuration space to prevent collisions with static obstacle regions. Equation 17 constrains the maximum pointwise collision probability for dynamic obstacles using the rectangular collision probability bound presented in Section III-B.2.

Both the f_{static} and $f_{\text{coll}}^{\text{rect}}$ constraint functions include orientation information, $\theta_{\text{robot},k} = \tan^{-1}(\frac{v_k^y}{v_k^x})$, making them non-convex. Both of these constraints are required to ensure the safety of the final set of contingency paths. These terms could be replaced with circular over approximations; however, this would prevent normal driving behavior such as passing in adjacent road lanes. The approach proposed in this paper is to include these non-convex constraint terms with the assumption that they won't have a significant impact on the optimization algorithm's convergence. A similar numerical optimization based planner used in Cornell's DUC robot included several non-convex constraint terms and was able to perform without any significant convergence problems in the DUC [8].

IV. SIMULATED SCENARIO

A simple scenario was simulated to evaluate the performance of the proposed path optimization algorithm. This scenario consists of a robot traveling down a two lane road, which encounters another vehicle traveling in the opposite direction; the other vehicle either turns in front of the robot or continues in its current lane. Each obstacle goal, (Turn, Go Straight), is initialized with equal likelihood in the prediction algorithm. A static version of the path optimization algorithm presented in Section III (without the $f_{\text{coll}}^{\text{circ}}$ and $f_{\text{coll}}^{\text{rect}}$ terms) is used as a probabilistic motion model in the obstacle prediction algorithm. As a baseline comparison, three algorithms were run: 1) the contingency approach proposed in this paper, 2) a single path variation where one contingency path is used that avoids all obstacle predictions, and 3) a static variation that uses a stationary obstacle prediction motion model. All other optimization parameters were identical.

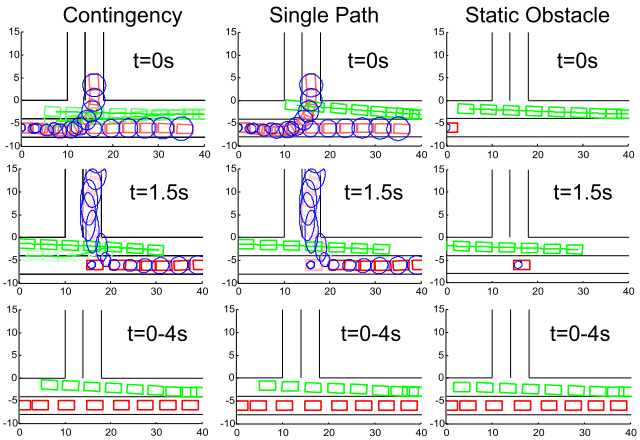


Fig. 6. Closed loop simulations with the obstacle continuing straight.

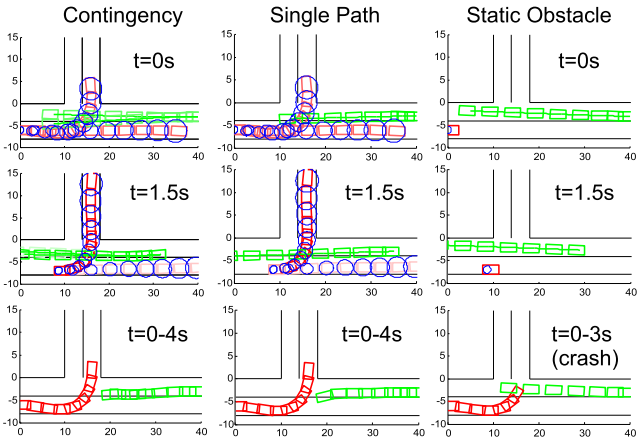


Fig. 7. Closed loop simulations with the obstacle turning.

Figures 6-7 show the simulated obstacle interactions for each of the planning algorithms; Figure 6 shows the case when the other vehicle continues straight and Figure 7 shows the case when the other vehicle turns. The planned motion of the robot (in green) and the predicted motion

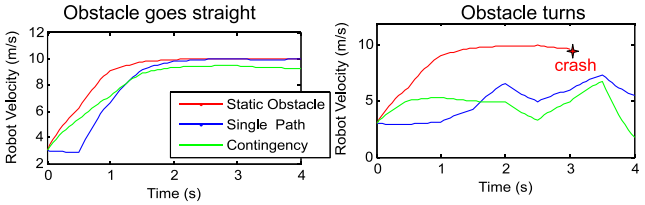


Fig. 8. Velocity profiles of the robot for the closed loop simulations.

of the obstacle (in red) are depicted by colored rectangles at half second intervals. The intensity of the plotted robot and obstacle rectangles correspond to the probability of that obstacle goal or contingency plan. For the trivial case when the obstacle vehicle continues straight, all three algorithms perform reasonably well. As seen in Figure 8, the output of the single path algorithm is a braking action initially because the obstacle vehicle is assumed to be able to both turn and go straight at the same time. The static algorithm accelerates the fastest, while the contingency approach balances speed and safety.

For the case where the obstacle vehicle turns, the single path algorithm again commands a braking action before it actually knows which goal the obstacle vehicle will follow. The static algorithm fails to realize the robot is turning until it is too late and collides with the obstacle at $t = 3s$. The contingency algorithm accelerates at a moderate rate until the other vehicle's intentions are well known, as defined by $p(G_j)$, then slows down to avoid a collision.

This simulation assumes that the obstacle motion model is accurate and the potential obstacle goals are known. To investigate the case when this is not true, a second simulation was run where the obstacle vehicle follows an unknown third goal and stops to wait for the robot. Results for the proposed contingency algorithm are shown in Figure 9. In this case, the obstacle prediction algorithm converges to the closest hypothesis, that the obstacle is turning, and the robot stops to avoid colliding with its false prediction of the obstacle's motion. This simulation highlights the need for acceptable obstacle prediction models for all possible obstacle goals.

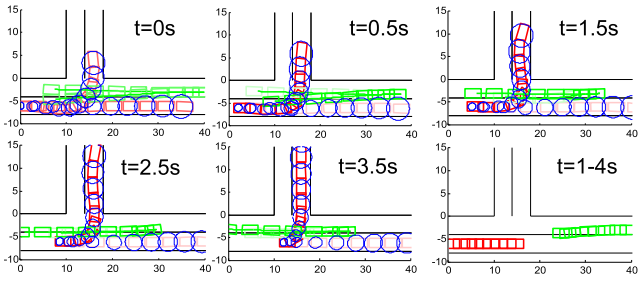


Fig. 9. Closed loop simulation with the obstacle vehicle stopping to wait for the robot.

The average optimization time for the proposed algorithm during these simulations was approximately 2 seconds in Matlab. Implementing this algorithm using a C++ based nonlinear optimization library is expected to improve the algorithm's runtime to at or near realtime performance ($5 - 10Hz$) for a moderate number of contingency plans.

V. CONCLUSIONS

An obstacle prediction and path optimization algorithm is proposed in this paper in order to provide a framework for the inclusion of obstacle goal inference in the motion planning process. The proposed path planning algorithm is able to intelligently compensate for probabilistic dynamic obstacles when provided with accurate obstacle predictions. However, the proposed obstacle prediction algorithm is limited in its ability to identify a complete set of obstacle goal hypotheses. Future research goals include extensive testing in Cornell's DUC simulation environment and additional research into identifying efficient obstacle motion models and determining a more complete set of obstacle goal predictions.

VI. ACKNOWLEDGMENTS

This research is partially supported by ARO Grant #W911NF-09-1-0466, with Dr. Randy Zachery as Program Manager and by the Department of Defense through the NDSEG fellowship program.

REFERENCES

[1] M. Buehler, K. Iagnemma, and S. Singh, *The Darpa Urban Challenge: Autonomous Vehicles in City Traffic*. Springer Verlag, 2010.

[2] L. Fletcher, S. Teller, E. Olson, D. Moore, Y. Kuwata, J. How, J. Leonard, I. Miller, M. Campbell, D. Huttenlocher, *et al.*, "The MIT-Cornell collision and why it happened," *Journal of Field Robotics*, vol. 25, no. 10, pp. 775–807, 2008.

[3] A. Kushleyev and M. Likhachev, "Time-bounded lattice for efficient planning in dynamic environments," in *Proceedings of the 2009 IEEE international conference on Robotics and Automation*. IEEE Press, 2009, pp. 4303–4309.

[4] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, "Learning motion patterns of people for compliant robot motion," *The International Journal of Robotics Research*, vol. 24, no. 1, p. 31, 2005.

[5] J. Miura and Y. Shirai, "Probabilistic uncertainty modeling of obstacle motion for robot motion planning," *Journal of Robotics and Mechatronics*, vol. 14, no. 4, pp. 349–356, 2002.

[6] I. Hwang and C. Seah, "Intent-Based Probabilistic Conflict Detection for the Next Generation Air Transportation System," *Proceedings of the IEEE*, vol. 96, no. 12, pp. 2040–2059, 2008.

[7] D. Ferguson, M. Darms, C. Urmson, and S. Kolski, "Detection, prediction, and avoidance of dynamic obstacles in urban environments," in *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, 2008.

[8] I. Miller, M. Campbell, D. Huttenlocher, F.-R. Kline, A. Nathan, J. C. Sergei Lupashin, B. Schimpf, P. Moran, N. Zych, E. Garcia, M. Kurdziel, and H. Fujishima, "Team cornell's skynet: Robust perception and planning in an urban environment," *Journal of Field Robotics*, vol. 25, no. 8, pp. 493–527, 2008.

[9] J. Hardy, M. Campbell, I. Miller, and B. Schimpf, "Sensitivity analysis of an optimization-based trajectory planner for autonomous vehicles in urban environments," E. M. Carapezza, Ed., vol. 7112, no. 1. SPIE, 2008, p. 711211. [Online]. Available: http://link.aip.org/link/?PSI/7112/711211/1

[10] L. Cremean, T. Foote, J. Gillula, G. Hines, D. Kogan, K. Kriechbaum, J. Lamb, J. Leibs, L. Lindzey, C. Rasmussen, A. Stewart, J. Burdick, and R. Murray, "Alice: An information-rich autonomous vehicle for high-speed desert navigation," *Journal of Field Robotics*, vol. 23, no. 9, pp. 777–810, 2006.

[11] S. Julier and J. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, vol. 3. Citeseer, 1997, p. 26.

[12] R. Paielli, H. Erzberger, and A. R. Center, "Conflict probability estimation for free flight," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 3, pp. 588–596, 1997.

[13] S. Alfano, "Addressing Nonlinear Relative Motion For Spacecraft Collision Probability," *AIAA Paper*, no. 2006-6760, p. 15, 2006.