MIT Open Access Articles

Motion planning with an analytic risk cost for holonomic vehicles

The MIT Faculty has made this article openly available. **Please share** how this access benefits you. Your story matters.

Citation: Greytak, M., and F. Hover. "Motion planning with an analytic risk cost for holonomic vehicles." Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on. 2009. 5655-5660. © Copyright 2010 IEEE

As Published: <http://dx.doi.org/10.1109/CDC.2009.5399943>

Publisher: Institute of Electrical and Electronics Engineers

Persistent URL: <http://hdl.handle.net/1721.1/59359>

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Terms of Use: Article is made available in accordance with the publisher's policy and may be subject to US copyright law. Please refer to the publisher's site for terms of use.

Motion Planning with an Analytic Risk Cost for Holonomic Vehicles

Matthew Greytak and Franz Hover

Abstract—We present a robust motion planning algorithm for a holonomic mobile robot that incorporates the risk of collisions directly in the cost function. This deterministic algorithm uses analytic predictions of the path-following error statistics to quickly calculate the collision risk. The A* search algorithm is used to find motion plans that optimally balance the plan duration with the collision risk, and these plans have a higher probability of success than plans that do not consider the collision risk. We present experimental results for an autonomous marine vehicle.

I. INTRODUCTION

For air, sea, and ground vehicles that navigate autonomously, robust and efficient motion planning is essential in order for the vehicle to complete its mission successfully. The planner must generate plans that are nominally collision-free, but the plans must also have a high probability of success if the vehicle is perturbed from the plan by external disturbances. In this paper we develop a motion planner for holonomic mobile robots that balances efficiency and collision risk to generate robust motion plans. Our specific application is the navigation of underactuated autonomous harbor patrol vessels in a dynamic environment.

A mobile robot may not be able to follow an arbitrary trajectory due to dynamic and kinematic constraints. Kinodynamic planners generate trajectories that are based on the known capabilities of the vehicle to ensure that each trajectory is feasible given the vehicle constraints. Some kinodynamic techniques are rapidly-exploring random trees (RRTs) [1], probabilistic roadmaps (PRMs) [2], and mixed-integer linear programming (MILP) solvers [3]. These planners can efficiently generate motion plans through a cluttered, time-varying environment. However, in most cases these planners do not consider uncertainties that may push the vehicle off the nominal trajectory while executing the plan.

There are several ways to incorporate uncertainty into the planning process. In the simplest approach, a heuristic measure of the uncertainty of each maneuver is added to the planner's cost function so that higher-risk plans are discouraged [4]. The disadvantage of this method is that it does not capture the propagation of path-following error throughout the plan. Another approach is to use a MILP particle method to predict the effects of external disturbances to guarantee a certain probability of mission success [5]. This is a probabilistic approach whose accuracy scales with the number of sample trajectories; the computational cost of accurately evaluating the collision probability is high. A

two-stage approach can be used to constrain the configuration space so that all plans in the reduced space have a guaranteed minimum probability of success [7]. If the Markov assumption is valid for a mobile robot system, then a PRM can be combined with a Markov decision process to find a trajectory that has a high probability of success [6]. Another method is to analytically and deterministically predict the error statistics throughout each plan and incorporate the predicted collision probability directly into the cost function of the planner [8]. This paper is an extension of [8] using a more efficient planner and a more accurate collision probability calculation. We consider only the uncertainty acting on the plant from Gaussian disturbances, and we assume a perfect measurement of the vehicle state and the locations of the obstacles in the environment. We do not require a static map, as long as the locations of the obstacles can be predicted accurately throughout the duration of the motion plan.

In our proposed planner we represent motion plans using a framework based on the Maneuver Automaton (MA) [9]. Plans are built using maneuvers from a library of speed control setpoints and waypoints. The A* search algorithm is used to find optimal plans within the motion planning framework based on a cost function that combines the plan duration with the predicted collision probability. We use an analytic approximation of the collision probability based on the vehicle dynamic model and the plan characteristics.

In Section II we introduce the vehicle's equations of motion, control system, and analytic path-following error predictions. In Section III we describe the framework used to describe motion plans. The planning algorithm and its risk-based cost function are described in Section IV. Finally, we present experimental results using this planner with a 1.25-meter autonomous marine vehicle in Section V.

II. VEHICLE DESCRIPTION AND CONTROL SYSTEM

In this section we describe the equations of motion for a holonomic vehicle and its control system. The control system is defined by a library of setpoints, each with its own control law. For planar vehicles perturbed by Gaussian disturbances it is possible to analytically predict the evolution of the error statistics when following each controller setpoint. These statistics are necessary for calculating the collision risk in Section IV.

A. Vehicle Equations of Motion

The vehicle state $\mathbf{x} = [\boldsymbol{\nu}^T, \boldsymbol{\eta}^T]^T$ is a vector containing the body-reference velocities $\boldsymbol{\nu}$ and the global position and orientation $\boldsymbol{\eta}$. For a planar vehicle, $\boldsymbol{\nu} = [u, v, r]^T$ and

M. Greytak and F. Hover are with the Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA. mgreytak@mit.edu, hover@mit.edu.

$\eta = [X, Y, \psi]^T$, where u and v are the surge and sway velocities and r is the yaw rate. The global position of the center of mass of the vehicle is (X, Y) and the heading angle is ψ . The vector of control inputs is τ . The damping and Coriolis matrix is $\mathbf{a}(\nu)$ and the matrix of control gains is $\mathbf{b}$. Process noise disturbances $\mathbf{w}_\nu$, sampled from a zero-mean multinormal distribution with a covariance matrix $\mathbf{W}_\nu$, perturb the system. The equation of motion for the velocities is:

$$\dot{\nu} = \mathbf{a}(\nu)\nu + \mathbf{b}\tau + \mathbf{w}_\nu. \quad (1)$$

$\mathbf{J}(\eta)$ is a rotation matrix from body-reference coordinates to the global coordinate system. The position states evolve as follows:

$$\dot{\eta} = \mathbf{J}(\eta)\nu. \quad (2)$$

The control law for τ is designed to make the state $\mathbf{x}$ track a reference state $\mathbf{r} = [\nu_r^T, \eta_r^T]^T$. The reference state itself must satisfy (2), so $\dot{\eta}_r = \mathbf{J}(\eta_r)\nu_r$. A natural choice for the control law has a feedforward term that multiplies the reference state and a feedback term that multiplies the vehicle state.

$$\tau = \mathbf{K}_r \mathbf{r} - \mathbf{K}_x \mathbf{x} \quad (3)$$

B. Speed Controller Setpoints

The control system can drive the vehicle to any velocity setpoint in the operating envelope of the vehicle. For use in the motion planning framework, described in Section III, it is necessary to choose a finite set of velocity setpoints. The capability of the planner increases with the number of setpoints, but so does the computational cost of finding an optimal plan. If the vehicle is underactuated, then the setpoints must be achievable by the vehicle in the steady state. For example, a speed controller for an underactuated marine vehicle can independently regulate the surge velocity u and the yaw rate r , but not the sway velocity v . In this paper we consider a set of nine setpoints that form a grid in the (u, r) space, as shown in (4), using a nominal forward speed U and a nominal yaw rate R .

$$\begin{bmatrix} (U, R) & (U, 0) & (U, -R) \\ (0, R) & (0, 0) & (0, -R) \\ (-U, R) & (-U, 0) & (-U, -R) \end{bmatrix} \Leftrightarrow \begin{bmatrix} \mathbf{a} & \mathbf{b} & \mathbf{c} \\ \mathbf{d} & \mathbf{e} & \mathbf{f} \\ \mathbf{g} & \mathbf{h} & \mathbf{i} \end{bmatrix} \quad (4)$$

The letter codes $\mathbf{a}$ through $\mathbf{i}$ in (4) are identification codes for each setpoint; these are used in Section III to efficiently represent a motion plan.

C. Position Control

A trajectory defined by a reference velocity ν_r has an associated reference position $\eta_r(t)$ based on the initial position and orientation of the vehicle. Using $\eta_r(t)$ we can add position feedback to the controller. For underactuated vehicles, position feedback may not be possible for all controller setpoints; for example, with an underactuated planar vehicle it is possible to regulate position only when the reference forward speed is nonzero. (When the controller setpoint is the origin, $\mathbf{e}$ in (4), this is known as underactuated

point stabilization [10], and it cannot be solved with a fixed feedback law.) Therefore we can only add position feedback to certain control setpoints for underactuated vehicles. In this paper we apply position feedback only to the setpoints for which the surge reference velocity is greater than zero: $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ in (4). In the general case, each controller setpoint can have an independent controller $\{\mathbf{K}_r, \mathbf{K}_x\}$.

D. Analytic Mean and Variance Predictions

Combining the system model (1-2) with the control law (3) for a particular controller setpoint (4) results in a nonlinear closed-loop system. The tracking error while following a setpoint is $\mathbf{e} = [\nu_e^T, \eta_e^T]^T = \mathbf{x} - \mathbf{r}$. For velocity control we expect the velocity error ν_e to be driven to zero (or, in the case of an underactuated vehicle, one or more components may converge to a non-zero steady-state value), and if position control is used then we expect the entire error vector $\mathbf{e}$ to converge. The basic statistics of the error vector are the mean $\bar{\mathbf{e}}(t) = E[\mathbf{e}(t)]$ and variance $\Sigma(t) = E[(\mathbf{e}(t) - \bar{\mathbf{e}}(t))(\mathbf{e}(t) - \bar{\mathbf{e}}(t))^T]$, where the expectations are taken over all possible noise processes $\mathbf{w}_\nu(t)$. For a planar holonomic vehicle, we can predict the mean and variance analytically if the velocity dynamics (1) and the controller (3) can be linearized around each controller setpoint; this analytic solution is derived in [11]. For fully-actuated vehicles the mean error is always zero if the initial error is zero, but for underactuated vehicles the mean error evolves through time. These error statistics are key to predicting the collision risk associated with motion plans.

III. MANEUVER AUTOMATON FRAMEWORK WITH WAYPOINTS

The motion planning framework used in this paper is based on the Maneuver Automaton (MA) framework [9]. In the MA framework, maneuvers comprise *trims*, which are periods of constant reference velocity, and *motion primitives*, which are state transitions from one trim to another. Motion plans are then formed by concatenating maneuvers together. A simple MA system is shown in Figure 1.

In this paper we forego motion primitives and allow trim maneuvers to be concatenated directly together. Without motion primitives the motion plan will have discontinuities in the reference velocities at the trim transitions, but the transients resulting from these discontinuities are handled by the analytic mean error solution described in the previous section [11]. By tracking the mean error throughout the motion plan, the framework becomes much simpler. Each motion plan can be represented as a string of controller setpoints with a corresponding vector of durations for each trim. For example, a motion plan with the setpoint sequence $\mathbf{a}\mathbf{c}\mathbf{b}$ is shown in Figure 2. The predicted vehicle trajectory does not exactly match the reference due to the setpoint transitions and underactuation, but the errors can be predicted analytically.

This motion planning framework reduces the planning domain from the entire free space to the subset reachable by concatenating discrete maneuvers together. This limits the

Fig. 1. A simple Maneuver Automaton system with four trim trajectories. Motion primitives (the straight arrows) are maneuvers that transition the vehicle from one trim to another. Clockwise from the top left, these trims correspond to the setpoints a, b, c, and e from (4).

Fig. 2. The motion plan acb (black) with the predicted mean and standard deviation of the path-following error (blue dashed) for an underactuated vehicle.

domain of the state space reachable by the vehicle, but it makes the planning process tractable.

A. Fixed-duration Trims

There are two main pieces of data associated with each motion plan: the setpoint sequence and the duration of each trim. Once the sequence has been set, the durations can be manipulated to generate different motion plans. If the trims to be manipulated correspond to straight-line motion only, then finding the optimal durations (given a cost function) is a linear programming problem [9]. To find the optimal durations for all types of trims, one must solve a nonlinear programming problem. For computational simplicity we use fixed-duration trims; that is, each maneuver in the motion plan lasts a fixed amount of time (five seconds in the examples in this paper). As the trim duration decreases, the reachable set of the state space increases, but the typical number of maneuvers in a motion plan increases. The cost of the search procedure for finding the optimal motion plan increases with the number of maneuvers in the plan, so the trim duration must be chosen to balance the plan resolution with the computational cost of the search. The trims are stored in a trim library $\mathcal{T}$, as in (4), and the corresponding maneuvers m_t with a fixed duration are stored in a maneuver library $\mathcal{M}$. Each maneuver is stored with its net position and orientation change, which can be computed analytically given the reference velocity during the maneuver.

Fig. 3. Waypoints (large circles) are automatically placed off the corners of the obstacles with a suitable buffer distance. The motion plan $awaw$ uses one waypoint on the way to the goal, and the goal itself is treated as a waypoint.

B. Waypoint Maneuvers

Waypoints are valuable navigational tools; they represent locations in the configuration space that are known to be safe, and they can be used to construct high-level motion plans. They can be automatically generated in the configuration space, either from a Voronoi graph or by simply placing waypoints a certain distance off the corners of obstacles. In this paper we use the latter approach with a buffer distance equal to the width of the vehicle. Additionally, we place a waypoint at the goal. The waypoints are stored in the waypoint library $\mathcal{W}$.

The disadvantage of using fixed-duration trims in a motion plan is that it takes many maneuvers concatenated together to span a large open region of the configuration space. As described above, shorter plans reduce the computational cost of the planner. For each waypoint we add a maneuver m_w to the maneuver library that drives the vehicle to the waypoint at a constant forward speed. It is necessary for the reference position and orientation to be continuous throughout the motion plan; therefore, a turning trim of an appropriate duration must be added to the plan before each waypoint maneuver. Figure 3 shows a motion plan, $awaw$ involving two waypoint maneuvers. The durations of the two a trims are calculated to align the trajectory with each waypoint.

C. Termination of Maneuvers

During the execution of the plan, it is important to know when the vehicle has completed a maneuver so that it can start the next maneuver. Adhering to the planned maneuver durations is often not the ideal policy. For example, when driving to a waypoint, the maneuver can be considered to be over when the vehicle crosses a line perpendicular to the motion plan at the waypoint. If the vehicle is lagging behind the reference, the controller lets the vehicle catch up before continuing the plan, and if the vehicle is ahead of the reference then it is not forced to wait at the end of the maneuver before continuing. In both cases, the along-track position at the start of the next maneuver is known exactly and the corresponding components of the error vector $\bar{e}$ and variance Σ are exactly zero. The error statistics prediction algorithm takes this property into account. When turning in place (maneuvers d and f in (4)), the maneuver is terminated

when the vehicle heading matches the reference heading at the end of the maneuver. Similar termination criteria can be used for the other maneuvers and trims. These termination conditions add robustness to the motion plan, which is reflected in the evolution of the mean and variance.

IV. A* PLANNING ALGORITHM

The motion planning framework described in the previous section is used to represent motion plans, but it does not generate them. To find the optimal motion plan we use the A* planning algorithm [12]. Optimality refers to minimizing a cost function such as the one described below. The A* algorithm expands a search tree from the initial vehicle state toward the goal configuration. This is known as kinodynamic planning, because each branch of the tree is a feasible motion plan that captures the vehicle dynamics and kinematics. While other tree expansion algorithms such as RRTs could be used instead, we choose A* for its optimality guarantees and determinism. The following sections describe the cost function, the heuristic used to guide the search, and the implementation of the search algorithm.

A. Cost Function

The planner uses a cost function g to compare different motion plans in the search tree. The cost function incorporates both the duration of the plan, T , and the risk associated with the plan. The quantitative risk is the probability of hitting an obstacle during the execution of the plan, P_{hit} . To balance the competing objectives of time and risk, we use the following nonlinear cost function:

$$g = \frac{T}{1 - P_{hit}}. \quad (5)$$

The cost function (5) strongly discourages plans with a low probability of success: $g \rightarrow \infty$ as $P_{hit} \rightarrow 1$. If the vehicle must restart the mission when it hits an obstacle, then (5) is the expected time required to complete the mission. For the planner to run efficiently, we must be able to compute g very quickly. The plan duration is trivial to compute for a given motion plan, but the collision probability is not. Monte Carlo simulations are too computationally expensive to be used in a real-time planner. The following section describes an analytic approximation to the collision probability.

B. Collision Probability

The collision probability can be approximated analytically from the mean and variance of the path-following error throughout the plan. However, predicting the collision probability from the error statistics is very difficult because the vehicle states are correlated in time, and the probability density function (PDF) for the vehicle position is truncated whenever the vehicle passes an obstacle. To simplify the calculation, we sample the relative obstacle positions at a certain rate, Δt_{obs} . At each sample time t_k we compute the component of the mean error vector $\bar{e}(t_k)$ in the direction of the nearest point on each obstacle, $\bar{y}$, and the corresponding component of the variance, Σ_y , extracted from $\Sigma(t_k)$. We call the separation distance d . If the position of the vehicle is

Fig. 4. The PDF of the vehicle position when the obstacle is 0.1 meters away, with a collision probability of 16.3% (blue), and the normal distribution with the same mean $\bar{y}_+$ and variance Σ_{y+} (red dashed). The loss due to this approximation is the integral of the dashed curve to the right of $y = 0.1$.

normally distributed, then the probability of hitting the j 'th obstacle at t_k is:

$$P_{hit,kj} = \frac{1}{2} \left(1 - \operatorname{erf} \frac{d}{2\Sigma_y} \right). \quad (6)$$

For the vehicle to execute the entire motion plan without hitting an obstacle, it must pass each obstacle without a collision. The probabilities of passing each obstacle multiply together, so the overall collision probability for the plan is:

$$P_{hit} = 1 - \prod_{k=1}^N \prod_{j=1}^{\# obs} (1 - P_{hit,kj}), \quad (7)$$

where N is the total number of sample points t_k used to evaluate the collision probability.

The PDF of the distribution of trajectories that successfully passed an obstacle is a truncated normal distribution, as shown in Figure 4:

$$p(y)_{kj} = \begin{cases} \frac{1}{(1 - P_{hit,kj})\sqrt{2\pi\Sigma_y}} \exp\left(-\frac{(y-\bar{y})^2}{2\Sigma_y}\right) & \text{for } y \leq d \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

We can use the analytic mean and variance equations [11] to propagate the error statistics from t_k to t_{k+1} , but doing so requires a normal distribution for the error state. An approximate normal distribution can be formed using the mean and variance of (8):

$$\bar{y}_+ = \bar{y}(1 - P_{hit,kj}) - \Sigma_y p_d \quad (9)$$

$$\Sigma_{y+} = (\Sigma_y + \bar{y}^2)(1 - P_{hit,kj}) - \Sigma_y(d + \bar{y})p_d - \bar{y}_+^2(1 + P_{hit,kj}) \quad (10)$$

$$\text{where } p_d = \frac{1}{\sqrt{2\pi\Sigma_y}} \exp\left(-\frac{(d - \bar{y})^2}{2\Sigma_y}\right).$$

These new values for the mean and variance of the error are inserted back into $\bar{e}(t_k)$ and $\Sigma(t_k)$ and the error statistics are propagated until the next sample time. The sample time that gives the most accurate collision probability prediction is a function of the vehicle dynamics; it can be determined offline from Monte Carlo simulations.

C. Heuristic

The A* algorithm examines plans in the search tree based on the predicted total cost of the plan. For incomplete plans (those that do not terminate at the goal), we use a heuristic function h to approximate the remaining cost from the end of the plan to the goal. The algorithm can only return the optimal plan if h never overestimates the cost to the goal. For simplicity, we ignore the collision probability in the heuristic and consider instead the minimum time required to arrive at the goal from the end of the plan. If the vehicle has a maximum forward speed U , then h is the minimum collision-free distance to the goal divided by U . To compute the minimum collision-free distance, we construct a visibility graph $\mathcal{D}$ that connects the obstacle corners with straight line segments in the free configuration space. Next we use Dijkstra's algorithm [13] to compute the minimum distance from each obstacle corner to the goal; for each node n , this cost is $C(n)$. These steps are performed offline before the search algorithm executes. Online, to compute h for a motion plan, the planner computes the distance from the end of the plan to each node, c_{kn} , which is infinity if the shortest straight line intersects an obstacle. Finally, the minimum time needed to move from the end of the plan to the goal is:

$$h = \min_{n \in \mathcal{D}} (C_n + c_{kn}) / U \quad (11)$$

D. Search Procedure

The overall search algorithm is listed in Algorithm 1. The planner combines maneuvers and waypoints to build motion plans that extend from the initial position of the vehicle to the goal location. To help the planner find the goal, we place a waypoint at the goal. The planner stores partial plans in a queue $\mathcal{Q}$, sorted by the predicted total cost $f = g + h$. The first plan in the queue (the one with the smallest f value) is expanded by adding fixed-duration trims from the maneuver library $\mathcal{M}$ and waypoint maneuvers toward each waypoint $\mathcal{W}$. Each new plan that does not intersect with an obstacle is added to the queue sorted by f . If the first plan in the queue ends at the goal, then that is guaranteed to be the optimal plan within the motion planning framework.

V. EXPERIMENTAL RESULTS

A series of experiments were performed in the Towing Tank at the Massachusetts Institute of Technology using an underactuated autonomous surface vessel. The vessel and the experiment setup are described below.

A. Underactuated Surface Vessel

The vehicle used in the experiment is a 1.25-meter autonomous model of an icebreaking ship (Figure 5). The vessel is powered by a single azimuthing thruster under the stern which can generate a thrust vector in any direction in the horizontal plane. A state space model for the vehicle dynamics, including the process noise due to waves, was derived from a series of simple system identification tests. The controller is a linear quadratic regulator designed around each control setpoint.

Algorithm 1 A* Algorithm for robust motion planning.

- 1: Define the goal set $\mathcal{G}$ around $\mathbf{x}_{goal}$.
- 2: Load the trim library $\mathcal{T}$ and use it to construct the maneuver library $\mathcal{M}$.
- 3: Create a library of waypoints $\mathcal{W}$ around the obstacles, with one at the goal.
- 4: Create a visibility graph $\mathcal{D}$ around the obstacles.
- 5: Create a plan p containing only the initial state $\mathbf{x}_0$.
- 6: Initialize the search queue $\mathcal{Q}$ with p .
- 7: **while** $|\mathcal{Q}| > 0$ **do**
- 8: Remove the first plan from the queue: $p \leftarrow \mathcal{Q}(0)$.
- 9: **if** p ends in $\mathcal{G}$ **then**
- 10: **return** p with success.
- 11: **for** each maneuver m in $\mathcal{M}$ **do**
- 12: Add m to p : $p' \leftarrow p + m$.
- 13: **if** p' is collision-free **then**
- 14: Compute the cost $g(p')$.
- 15: Compute the predicted cost-to-go $h(p')$ using $\mathcal{D}$.
- 16: Insert p' into $\mathcal{Q}$ sorted by $f(p') = g(p') + h(p')$.
- 17: **for** each waypoint $w \in \mathcal{W}$ **do**
- 18: Calculate the appropriate trim maneuver m_t to align the vehicle with w .
- 19: Add m_t to p : $p' \leftarrow p + m_t$.
- 20: Calculate the maneuver m_w that ends at w .
- 21: Add m_w to p' : $p' \leftarrow p' + m_w$.
- 22: **if** p' is collision-free **then**
- 23: Compute the cost $g(p')$.
- 24: Compute the predicted cost-to-go $h(p')$ using $\mathcal{D}$.
- 25: Insert p' into $\mathcal{Q}$ sorted by $f(p') = g(p') + h(p')$.
- 26: **return** with failure: there is no feasible path to the goal.

Fig. 5. The 1.25-meter underactuated surface vessel used in the experiment.

B. Motion Plan

The planner (Algorithm 1) was used to find a path through the environment shown in Figure 6. The nominal forward speed is $U = 0.15$ m/sec and the nominal yaw rate is $R = 9^\circ/\text{sec}$. The optimal plan $c(cw)c(aw)$ uses two 5-second trim maneuvers c and two waypoint maneuvers cw and aw . The duration of the plan is 32.36 seconds and the collision probability is 2.2%. This plan was found after 28 iterations of the search algorithm in 1.33 seconds on a 2.33 MHz Intel Core 2 Duo processor. If the planner does not consider the collision probability in the cost function, then

Fig. 6. The vehicle drives from the initial configuration, as pictured, to the green square. The search tree is shown in gray (top) and the optimal plan is shown in black. The waypoints considered by the planner are shown as large pink circles. The predicted mean and standard deviation for the path-following error are shown as dashed blue lines. Five experimental trajectories are shown in green in the lower figure.

the shortest motion plan drives straight between the two islands with a duration of 31.67 seconds and a collision probability of 11.4%; this plan is 2.1% shorter than the optimal plan but it is over 5 times as likely to hit an obstacle. The optimal plan and the search tree are shown in Figure 6. The search tree is strongly directed toward the goal due to the A* heuristic function h .

C. Experiments

The optimal plan $c(cw)c(aw)$ was executed by the autonomous surface vessel shown in Figure 5 while a wave-maker generated 2.4-Hz waves with a 2-cm wave height and a wavelength of 27 cm moving from left to right in Figure 6. The trajectories from five separate executions of the plan are plotted in the figure. The trajectories mostly remain within one standard deviation of the reference trajectory. Differences in the mean error can be attributed to modeling error and a slight unmodeled drift from left to right due to the net effects of the waves.

VI. CONCLUSIONS AND FUTURE WORK

We have presented a robust motion planning algorithm for holonomic mobile robots that incorporates collision risk directly into the cost function. The motion plans generated by this planner are optimal within the planning domain defined

by the finite maneuver library and the waypoint library. The planner is deterministic, using analytic predictions of the path-following error statistics and the collision probability rather than resorting to particle methods or Monte Carlo simulations. Analytic methods throughout the algorithm allow the planner to be fast enough to be used in real-time.

The only part of the planning algorithm that is not analytic is the calculation of the optimal sample time used to compute the collision probability from the error statistics. We are currently developing an analytic derivation for this optimal sample time based on the vehicle's dynamic model.

The overall algorithm presented in this paper applies to any holonomic vehicle; the only step that requires a planar vehicle is the analytic prediction of the mean and variance of the path-following error. We are currently working to extend this prediction to apply to mobile robots operating in 6 degrees of freedom, as well as non-holonomic robots.

VII. ACKNOWLEDGMENTS

This work was supported by the Office of Naval Research, Grant N00014-02-1-0623, and by a Science, Mathematics and Research for Transformation (SMART) scholarship (M. Greytak).

REFERENCES

- [1] S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report 98-11, Computer Science Dept., Iowa State University, 1998.
- [2] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.
- [3] T. Schouwenaars, B. Mettler, E. Feron, and J. How. Hybrid architecture for full-envelope autonomous rotorcraft guidance. *American Helicopter Society 59th Annual Forum*, 2003.
- [4] T. Schouwenaars, B. Mettler, E. Feron, and J. How. Robust motion planning using a maneuver automation with built-in uncertainties. In *Proceedings of the 2003 American Control Conference*, volume 3, pages 2211–2216, 2003.
- [5] L. Blackmore. A probabilistic particle control approach to optimal, robust predictive control. *Proceedings of the AIAA Guidance, Navigation and Control Conference*, 2006.
- [6] R. Alterovitz, T. Simeon, and K. Goldberg. The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty. In *Proceedings of Robotics: Science and Systems*, 2007.
- [7] M. Ono and B. Williams. Efficient motion planning algorithm for stochastic dynamic systems with constraints on probability of failure. In *23rd AAAI Conference on Artificial Intelligence*, Chicago, IL, 2008.
- [8] M. Greytak and F. Hover. Robust motion planning for marine vehicle navigation. In *Proceedings of the 18th International Offshore and Polar Engineering Conference*, volume 2, pages 399–406, Vancouver, Canada, 2008.
- [9] E. Frazzoli. Maneuver-based motion planning and coordination for single and multiple UAV's. In *AIAA 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles*, Portsmouth, Virginia, 2002.
- [10] M. Greytak and F. Hover. Underactuated point stabilization using predictive models with application to marine vehicles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3756–3761, Nice, France, 2008.
- [11] M. Greytak and F. Hover. Analytic error variance predictions for planar vehicles. In *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, pages 471–476, Kobe, Japan, 2009.
- [12] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.
- [13] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.