Analytic error variance predictions for planar vehicles

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

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>As Published</td>
<td><a href="http://dx.doi.org/10.1109/ROBOT.2009.5152583">http://dx.doi.org/10.1109/ROBOT.2009.5152583</a></td>
</tr>
<tr>
<td>Publisher</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>Version</td>
<td>Final published version</td>
</tr>
<tr>
<td>Accessed</td>
<td>Sat Dec 08 22:43:46 EST 2018</td>
</tr>
<tr>
<td>Citable Link</td>
<td><a href="http://hdl.handle.net/1721.1/59358">http://hdl.handle.net/1721.1/59358</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>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.</td>
</tr>
<tr>
<td>Detailed Terms</td>
<td></td>
</tr>
</tbody>
</table>
Analytic Error Variance Predictions for Planar Vehicles

Matthew Greytak and Franz Hover

Abstract—Path planning algorithms that incorporate risk and uncertainty need to be able to predict the evolution of path-following error statistics for each candidate plan. We present an analytic method to predict the evolving error statistics of a holonomic vehicle following a reference trajectory in a planar environment. This method is faster than integrating the plant through time or performing a Monte Carlo simulation. It can be applied to systems with external Gaussian disturbances, and it can be extended to handle plant uncertainty through numerical quadrature techniques.

I. INTRODUCTION

Mobile robots are becoming increasingly prevalent in terrestrial, marine, and aerial environments. The robots must be able to plan safe and efficient trajectories through these complex environments. There are many planning algorithms that can find feasible trajectories around fixed or moving obstacles. These techniques include grid search [1], graph search [2], and Rapidly-Exploring Random Trees (RRTs) [3], among others. In many applications it is also necessary to account for external disturbances or model uncertainty when generating plans so that the resulting trajectory will be robust to these effects. To that end, a graph search algorithm has been extended to include the uncertainty associated with each edge of the graph [4]. It is possible to directly include the risk of collisions in the cost function [5], but to do so requires an estimate of the position error variance throughout the motion plan. Another domain where error variance predictions are necessary is Robust Model Predictive Control (RMPC). Conventional MPC is a technique in which various control sequences are considered over a finite time horizon in order to choose the one that results in the best state trajectory. RMPC extends conventional MPC to include process noise and plant uncertainty [6]. RMPC either follows the direct numerical integration approach to propagate error statistics [7] or it uses Monte Carlo simulations sampled from a set of plant uncertainties and disturbances.

Figure 1 presents an experiment [5] showing how error variance predictions can be used. The planner ensures that there are no obstacles near the nominal path in the sections of the path where the predicted error variance is high. In the figure, the measured trajectory of the vehicle mostly remains within one standard deviation of the predicted cross-track error distribution around the reference path.

The planning framework that we consider generates plans by concatenating different maneuvers together [2]. Each maneuver is a finite-duration “trim trajectory,” which is a period of constant reference velocity. Complex plans, such as the one shown in Figure 1, can be generated in this framework.

In this paper we develop an analytic prediction of the state variance due to process noise for a holonomic mobile robot that moves in a plane. This solution is an explicit function of time that does not require numerical integration of the plant, and it can be extended to include plant uncertainty through numerical quadrature techniques. The solution is based on two evaluations of the matrix exponential per maneuver, making it much faster than an equivalent set of Monte Carlo solutions. The vehicle model is introduced in Section II. The error variance evolution for a single maneuver is computed in Section III; through a convenient change in coordinates, the error variance can be written as a Riccati equation with constant coefficients, which can be solved analytically. Section IV describes how to handle underactuated plants and/or plants with parameter uncertainty, as well as transitions between different maneuvers within a complex motion plan. A MATLAB simulation of the vehicle shown in Figure 1 is provided in Section V to verify the results.
II. SYSTEM DESCRIPTION

The techniques presented in this paper apply to any holonomic vehicle operating in a planar environment. First we present the system model, then we describe the linear full-state feedback controller for the system.

A. Vehicle Plant

Following the standard nomenclature for marine vehicles [8], the body-centric translational and rotational velocities are contained in the vector \( \nu = [u, v, r]^T \) where \( u, v \) and \( r \) are the surge, sway and yaw velocities, respectively. The global position and orientation states \( X, Y \) and \( \psi \) are contained in the vector \( \eta = [X, Y, \psi]^T \).

The control vector \( \tau \) is the same size as \( \nu \) in a fully-actuated plant. External disturbances (process noise) also act on the system. The noise vector \( \nu \) is shown below. The damping matrix \( a(\nu) \) may be a function of velocity, and in this section \( b \) is assumed to have full rank, i.e. the vehicle is fully-actuated.

\[
\dot{\nu} = a(\nu)\nu + b\tau + \nu
\]

\( J(\psi) \) is a rotation matrix from body-fixed coordinates to global coordinates. Using this rotation matrix, the body-frame velocities \( \nu \) are mapped to the global position state evolution.

\[
\dot{\eta} = J(\psi)\nu \quad \text{where} \quad J(\psi) = \begin{bmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix}
\]

Equations (1-2) are combined in a nonlinear state space equation to describe the evolution of the full state vector \( x = [\nu^T, \eta^T]^T \).

\[
\dot{x} = A(x)x + B\tau + w, \quad \text{where} \quad A(x) = \begin{bmatrix} a(\nu) & 0 \\ J(\psi) & 0 \end{bmatrix}, \quad B = \begin{bmatrix} b \\ 0 \end{bmatrix}, \quad w = \begin{bmatrix} \nu \nu \end{bmatrix}
\]

B. Linear Full-State Feedback Controller

The control task is to use the control input \( \tau \) to cause the full state vector \( x \) to track a reference \( r = [\nu_r^T, \eta_r^T]^T \). Because the full state vector contains both velocities and positions, it is assumed that the reference is self-consistent:

\[
\dot{\eta}_r = J(\psi_r)\nu_r
\]

For simplicity we assume that \( \nu_r = 0 \), that is, each reference trajectory is a constant-velocity maneuver [2]. Transitions between different maneuvers within a motion plan are discussed in Section IV.

The linear controller has a feedforward term \( K_r \) that acts on the reference and a feedback term \( K_x \) that acts on the state vector.

\[
\tau = -K_x x + K_r r
\]

The feedback matrix \( K_x \) has a part that operates on the velocity vector \( \nu \) and a part that operates on the position vector \( \eta \). The feedforward matrix \( K_r \) has a similar structure.

\[
K_x = \begin{bmatrix} K_\nu & K_\eta \end{bmatrix} \quad (7)
\]

\[
K_r = \begin{bmatrix} K_\nu - b^{-1}a(\nu) & K_\eta \end{bmatrix} \quad (8)
\]

C. Closed-loop System

Putting the control law (6) into the state space system (3) results in the following nonlinear closed-loop system,

\[
\dot{x} = A_{cl}(x)x + BK_r(\nu)r + w, \quad \text{where} \quad A_{cl}(x) = A(x) - BK_x = \begin{bmatrix} a(\nu) - bK_\nu & -bK_\eta \\ J(\psi) & 0 \end{bmatrix} \quad (10)
\]

The error vector \( e = x - r = [\nu_e^T, \eta_e^T]^T \) represents the deviation of the state vector from the reference. Two important statistics used later are the mean error \( \bar{e} \equiv E[e] \) and the error covariance \( \Sigma \equiv E[(e - \bar{e})(e - \bar{e})^T] \).

While \( \tau \) is a function of the global position state \( \eta \), the position feedback control law is usually defined in reference-local coordinates. When the control law is expanded using the control matrix definitions (7-8), the resulting control law is:

\[
\tau = -K_\nu \nu_e - K_\eta \eta_e - b^{-1}a(\nu)\nu_r \quad (11)
\]

Ideally the position error term \( K_\eta \eta_e \) is not a function of the reference position or orientation. The position error in reference-local coordinates is \( \eta_{\bar{r}} = J(\psi_r)^T \eta_e \). The local version of the position error feedback matrix, designed around \( \psi = 0 \), is \( K_{\eta_0} \). The global position error feedback matrix is calculated from \( K_{\eta_0} \) below.

\[
K_\eta \eta_e = K_{\eta_0} \eta_{\bar{r}} = K_{\eta_0} J(\psi_r)^T \eta_e \quad K_\eta = K_{\eta_0} J(\psi_r)^T \quad (12)
\]

In the control implementation, the feedback matrix \( K_{\eta_0} \) is designed once for the \( \psi = 0 \) system and is rotated to the global coordinate system using (12) as needed during real-time operations.

III. ERROR STATISTICS EVOLUTION

In this section we derive the mean and variance of the error \( \bar{e} \) during a maneuver. First we compute the error dynamics based on the plant and controller in Section II, then we derive the statistics in global coordinates and reference-local coordinates.

A. Error Dynamics

The evolution of \( \bar{e} \) is shown below.

\[
\dot{\bar{e}} = \dot{x} - \dot{r} = A_{cl}(x)x + BK_r(\nu)r + w - \dot{r} = A_{cl}(x)e + A_r(\psi)r - \dot{r} + w \quad (13)
\]

The matrix \( A_r(\psi) \) used in (13) is:

\[
A_r(\psi) \equiv A_{cl}(x) + BK_r(\nu) = \begin{bmatrix} 0 & 0 \\ J(\psi) & 0 \end{bmatrix} \quad (14)
\]
The second and third terms of (13) can be expanded based on the self-consistency of\( r \) (5).
\[
A_r(\psi) r - \dot{r} = \begin{bmatrix} 0 \\ J(\psi) \nu_r \end{bmatrix} - \begin{bmatrix} \nu_r = 0 \\ \eta_r = J(\psi_r) \nu_r \end{bmatrix}
\]
(15)

To simplify (15), consider the rate of change of\( J(\psi) \):
\[
\dot{J}(\psi) = S(r)J(\psi) \]
where
\[
S(r) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}
\]
(16)

Using this notation and assuming that the heading error\( \psi_e \) is small, the 3 \times 3 lower submatrix of (15) is:
\[
\begin{align*}
(J(\psi) - I)J(\psi_r)\nu_r &= S_r J(\psi_r)\nu_r \\
&= \begin{bmatrix} 0 & 0 & S_r J(\psi_r)\nu_r \end{bmatrix} \eta_e
\end{align*}
\]
(17)

Defining a new 3 \times 3 matrix\( A_{22} \):
\[
A_{22}(r) \equiv \begin{bmatrix} 0 & 0 & S_r J(\psi_r)\nu_r \end{bmatrix},
\]
(19)
equation (15) reduces to the convenient expression shown below.
\[
A_r(\psi) r - \dot{r} = \begin{bmatrix} 0 & 0 \\ 0 & A_{22}(r) \end{bmatrix} e
\]
(20)

Inserting this new matrix into (13) results in a new closed-loop state matrix\( A_{\text{clr}} \) that incorporates the non-zero reference.
\[
A_{\text{clr}}(x, r) = \begin{bmatrix} a(\nu) - bK_r & -bK_\eta \\ J(\psi) & A_{22}(r) \end{bmatrix}
\]
(21)

Using (21), the error dynamics of the closed loop system are now:
\[
\dot{e} = A_{\text{clr}}(x, r)e + w
\]
(22)

Equation (22) includes the nonlinear dynamics of\( a(\nu) \), but also nonlinear time-varying kinematics involving\( \psi_r(t) \). The kinematic nonlinearities can be eliminated by changing to a reference-local coordinate system.

B. Error Dynamics in Reference-Local Coordinates

The error vector\( e \) contains the reference-local velocity error\( \nu_e \) and the global position error\( \eta_e \). The reference-local error vector\( e_0 \) is defined using the block matrix\( J_r \).
\[
e_0 = \begin{bmatrix} \nu_e \\ \eta_e \end{bmatrix} = J_r^T e \quad \text{where} \quad J_r \equiv \begin{bmatrix} I & 0 \\ 0 & J(\psi_r) \end{bmatrix}
\]
(23)

The reference-local error variance\( \Sigma_0 \) is defined through a similar transformation.
\[
\Sigma_0 \equiv E[(e_0 - e_0)e_0 - e_0)^T] = J_r^T \Sigma J_r
\]
(24)

The evolution of\( e_0 \) is derived below:
\[
\dot{e}_0 = J_r^T \dot{e} + J_r^T e
= J_r^T A_{\text{clr}}(x, r)e + J_r^T w + J_r^T S_r^T e
= J_r^T \left( A_{\text{clr}}(x, r) + S_r^T \right) J_r e_0 + w
= A_0 e_0 + w
\]
(25)

where\( S_r \equiv \begin{bmatrix} 0 & 0 \\ 0 & S(r) \end{bmatrix} \) and
\[
A_0 = J_r^T \left( A_{\text{clr}}(x, r) + S_r^T \right) J_r
\]
(27)

Equation (27) defines three new submatrices.\( A_{11}(\nu) \) is the closed-loop state matrix for the velocities.\( A_{12} \) uses the locally-defined position feedback matrix, and is therefore independent of\( J(\psi_r) \).
\[
A_{12} = -bK_r J(\psi_r) = -bK_{\eta}0
\]
(28)

\( S_r(\nu, \eta) \) simply reduces to a matrix of reference velocities.
\[
S_r(\nu, \eta) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}
\]
(29)

If the velocity errors are small and\( a(\nu) = a(\nu_r) \), then from the definitions in (26-29), we see that\( A_0 \) is a constant matrix for each maneuver defined by a constant reference velocity; this means the reference-local closed-loop system (25) can be solved analytically to compute the error statistics.

C. Mean and Variance Evolution

The mean value of (25) can be solved analytically to find the mean reference-local error throughout the maneuver.
\[
\frac{\partial \bar{e}_0}{\partial \bar{e}_0} = A_0 \bar{e}_0
\]
(30)

The mean error vector in the original coordinates is computed using the transformation\( \bar{e}(t) = J_r(t) \bar{e}_0(t) \). Next we solve for the covariance matrix of the reference-local error. Using the noise variance matrix\( W \),
\[
W \equiv E[ww^T] = \begin{bmatrix} W_\nu & 0 \\ 0 & 0 \end{bmatrix}
\]
(31)
equation of the mean of (25) is the linear variance equation [9] for the system shown in (25).
\[
\Sigma_0 = A_0 \Sigma_0 + \Sigma_0 A_0^T + W
\]
(32)

The reference-local error variance evolution equation (32) is a Riccati equation with constant coefficients. This equation can be solved analytically using the method described in [10]. First a block matrix\( M \) is constructed using the coefficient matrices of (32).
\[
M = \begin{bmatrix} -A_0^T & 0 \\ W & A_0 \end{bmatrix}
\]
(33)

Next a block matrix\( Y \) is defined using two square matrices\( Y_1 \) and\( Y_2 \), each matching the size of\( \Sigma_0 \).
\[
Y = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}
\]
(34)

Equation (32) is equivalent to the first-order differential equation\( \dot{Y} = MY \) with the initial conditions shown in
This differential equation is solved using the matrix exponential.
\[
Y(t) = e^{Mt}Y(0) \\
0(t) = Y_2(t)Y_1(t)^{-1}
\]  
(35)

Once the system is in the steady state, the inversion of \( Y_1 \) may result in numerical instability. In that case, the steady-state solution for \( \Sigma_0 \) can be found by solving the Lyapunov equation. The variance of the original error vector \( e \) is found using the transformation \( \Sigma(t) = J_r(t)\Sigma_0(t)J_r(t)^T \).

IV. SPECIAL CASES AND EXTENSIONS

The analytic solutions (30) and (35) are valid for a fully-actuated vehicle for which the plant parameters \( a(\nu) \) and \( b \) are known exactly. This section extends the solution to underactuated systems with imperfect knowledge of the plant. We also describe how to handle transitions between different maneuvers within a motion plan.

A. Underactuated Vehicles

For many vehicles, the number of independently actuated channels is less than the number of degrees of freedom. For example, marine vehicles are often controlled by two inputs, the throttle and the rudder angle, which are insufficient to independently control the three degrees of freedom surge, sway and yaw.

The earlier assumption of full actuation was used in (8) to define the feedforward gain \( K_r \). When the vehicle is fully-actuated, the inverse of the input matrix, \( b^{-1} \), is defined; for underactuated vehicles it is not, either because it is singular or because it is non-square. One workaround is to use the pseudoinverse \( b^* \), but the control designer may choose to apply feedforward to certain states and neglect others entirely. For generality we define \( b^{(inv)} \) which is used in (36) to construct \( K_r(\nu) \).

\[
K_r(\nu) = \begin{bmatrix} K_\nu - b^{(inv)}a(\nu) & K_{\eta} \end{bmatrix}
\]  
(36)

For a fully actuated system, \( b^{(inv)} = b^{-1} \) for exact feedforward. For underactuated systems, the size of \( b^{(inv)} \) is equal to the size of \( b^T \).

Continuing the analysis, we update the matrix \( A_r(\nu) \) in (14):

\[
A_r(x) = \begin{bmatrix} a(\nu) - bb^{(inv)}a(\nu) & 0 \\ J(\psi) & 0 \end{bmatrix}
\]  
(37)

Using the result in (37), the quantity \( A_r(x)r - \dot{r} \) becomes:

\[
A_r(x)r - \dot{r} = \begin{bmatrix} (a(\nu) - bb^{(inv)}a(\nu))\nu_r \\ A_{22}(r)\eta_r \end{bmatrix}
\]  
(38)

The lower half of the matrix in (38) is accounted for in \( A_{clr} \) and \( A_0 \). The remainder is defined as \( d(\nu_\tau) \), which captures the tracking error due to underactuation. As before, we assume that the velocity errors are small.

\[
d(\nu_\tau) = \begin{bmatrix} (a(\nu_\tau) - bb^{(inv)}a(\nu_\tau))\nu_\tau \\ 0 \end{bmatrix}
\]  
(39)

The evolution equations for the original error vector and the locally-defined error vector are listed in (40).

\[
\dot{e} = A_{clr}(x,r)e + d(\nu_\tau) + w \\
\dot{e}_0 = A_0e_0 + d(\nu_\tau) + w 
\]  
(40)

In (40), \( d(\nu_\tau) \) acts as a constant forcing term for the maneuver. The updated mean error solution (30) is:

\[
\bar{e}_0(t) = e^{A_{\nu_\tau}t}\bar{e}_0(0) + A_0^{-1}(e^{A_{\nu_\tau}t} - I)d(\nu_\tau)
\]  
(41)

The error variances \( \Sigma \) and \( \Sigma_0 \) are unaffected by the constant term \( d(\nu_\tau) \). In summary, when (41) replaces (30), the analytic solution can be applied to underactuated vehicles as well as fully-actuated vehicles.

B. Parameter Error

The solution presented in (35) only accounts for process noise. However, another source of error variance is plant uncertainty. If a controller is designed around an assumed plant \((\bar{a}, \bar{b})\) which differs from the actual plant \((a, b)\), then nonzero reference velocities will cause state errors. We assume that the distribution of the estimated parameter values is known; for example, they may follow a normal distribution around the best guess of the values with a known variance. Using numerical quadrature techniques [11], the state variance resulting from these errors can be determined. First we must modify the analytic solution to account for an imperfect knowledge of the parameter values.

Suppose the control designer believes the plant model contains the parameters \( \hat{a}(\nu) \) and \( \hat{b} \) instead of the true values \( a(\nu) \) and \( b \). The controller is designed based on the estimated plant, so the feedforward matrix \( K_r(\nu) \) in (36) is:

\[
K_r(\nu) = \begin{bmatrix} K_\nu - \hat{b}^{(inv)}\hat{a}(\nu) & K_{\eta} \end{bmatrix}
\]  
(42)

Propagating this change, we update the vector \( d(\nu_\tau) \):

\[
d(\nu_\tau) = \begin{bmatrix} (a(\nu_\tau) - \hat{b}^{(inv)}\hat{a}(\nu_\tau))\nu_\tau \\ 0 \end{bmatrix}
\]  
(43)

The forcing term \( d(\nu_\tau) \) affects the mean error \( \bar{e}_0 \). The error variance is also affected because the matrix \( A_0 \) contains the controller gains \( K_x \) designed around \((\bar{a}, \bar{b})\).

Quadrature techniques can be more efficient than Monte Carlo simulations to compute the statistics of state errors due to parameter uncertainty, especially if the parameters follow a known distribution. Using the results developed in the previous sections, each sample point already incorporates the process noise and requires only an analytic calculation rather than a simulation. To implement the quadrature, each sample point is a vector of parameter values. These values are used to form \( \hat{a}(\nu) \) and \( \hat{b} \). The resulting error variance matrices \( \Sigma_0 \) are summed according to the quadrature’s weighting rule. The aggregate error variance matrix contains the effects of process noise as well as parameter error.

C. Concatenating Multiple Maneuvers

The analysis so far has been restricted to maneuvers with a constant reference velocity \( \nu_\tau = 0 \). As multiple maneuvers are concatenated in a motion plan, the error statistics propagate through the plan. At the transitions from one maneuver to the next, the initial conditions must be compatible. The vehicle state is constant from the moment before \((-)\) to the moment after \((+)\) the transition, as is the
position reference. The reference-local error initial conditions are derived below:
\[ \mathbf{x} = r_− + e_− = r_+ + e_+ \]
\[ e_+ = e_− − (r_+ − r_−) \]
\[ e_{0+} = e_{0−} − (r_+ − r_−) \] (44)

Equation (44) is used to construct the initial condition in the mean error prediction equations (30 and 41). Next consider the error variance before and after the transition, shown below. Note that \( \mathbf{J}_{r+} = \mathbf{J}_{r−} = \mathbf{J}_r \).
\[ \mathbf{Σ}_{0−} = \mathbf{J}_r^T \mathbf{E} \left[ (\mathbf{x} − r_− − \bar{e}_−)(\mathbf{x} − r_− − \bar{e}_−)^T \right] \mathbf{J}_r \]
\[ \mathbf{Σ}_{0+} = \mathbf{J}_r^T \mathbf{E} \left[ (\mathbf{x} − r_+ − \bar{e}_+)(\mathbf{x} − r_+ − \bar{e}_+)^T \right] \mathbf{J}_r \] (45)

Substituting the result from (44), the values in the expectations of \( Σ_{0+} \) and \( Σ_{0−} \) are the same, meaning the variance matrix does not change across the transition.

V. SIMULATION RESULTS

The analytic variance solution was verified in a MATLAB simulation of the underactuated marine vehicle shown in Figure 1. A simplified model for this vehicle, obtained through step response tests in the MIT Towing Tank, is provided below. This dynamic model is designed around a zero forward speed and it assumes fore-aft symmetry, resulting in a constant diagonal \( a(\nu) \) matrix. Position units are meters and angular units are radians.
\[
\mathbf{a} = \begin{bmatrix}
-0.0357 & 0 & 0 \\
0 & -0.3 & 0 \\
0 & 0 & -0.286
\end{bmatrix}
\]
\[
\mathbf{b} = \begin{bmatrix}
-0.0636 & 0 \\
0 & 0.05 \\
0 & -0.156
\end{bmatrix}
\]
\[
\mathbf{W}_\nu = 10^{-3} \mathbf{I}_{3\times3}
\] (46)

The feedback matrix \( \mathbf{K}_{x0} \) is a linear quadratic regulator designed around a forward speed of 0.3 m/sec using a state weighting matrix \( \mathbf{Q} = \mathbf{I}_{6\times6} \) and a control weighting matrix \( \mathbf{R} = 0.01 \mathbf{I}_{2\times2} \). Because the vehicle is underactuated, \( \mathbf{b} \) cannot be inverted. Instead we choose a \( \mathbf{b}^{(inv)} \) matrix that leaves sway unactuated: the first and third columns of \( \mathbf{b}^{(inv)} \) are the inverse of the first and third rows of \( \mathbf{b} \).
\[
\mathbf{b}^{(inv)} = \begin{bmatrix}
15.7 & 0 & 0 \\
0 & 0 & -6.42
\end{bmatrix}
\] (47)

Consider a path planner that compares many different combinations of maneuvers to find the overall plan that best balances the time to the goal and the probability of colliding with obstacles along the way. The planner needs to know the error statistics throughout the plan to be able to compute the collision risk. Figure 2 shows a candidate plan for a mission in which the vehicle must start at the origin and round an obstacle (the vertical line in the figure). The reference trajectory in this example comprises three maneuvers, each lasting 10 seconds. The reference velocities are shown below.
\[
\nu_{r,1} = \begin{bmatrix}
0.3 \\
0 \\
0
\end{bmatrix} \quad \nu_{r,2} = \begin{bmatrix}
0.3 \\
0 \\
0.25
\end{bmatrix} \quad \nu_{r,3} = \begin{bmatrix}
0.3 \\
0 \\
0
\end{bmatrix}
\] (48)

Figure 2 shows one standard deviation of cross-track error for a batch of 100 Monte Carlo evaluation points and from the analytic solution. The standard deviation of the cross-track error is plotted against time in Figure 3. The statistics converge to the analytic prediction as the number of Monte Carlo points increases. Note that the maneuver transitions act as impulses to the mean error.

Based on the error statistics it is possible to determine the probability of successfully passing the obstacle in Figure 2 [5]. Table I presents this probability based on the analytic solution and Monte Carlo simulation using 10, 100 and 1000 evaluation points. For the Monte Carlo simulations this probability is computed both from the mean and variance statistics and from counting the number of trajectories that
TABLE I

<table>
<thead>
<tr>
<th></th>
<th>Analytic</th>
<th>MC 10</th>
<th>MC 100</th>
<th>MC 1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_{\text{pass}}$ (statistics)</td>
<td>0.8890</td>
<td>0.8191</td>
<td>0.9021</td>
<td>0.8636</td>
</tr>
<tr>
<td>Trajectories</td>
<td>8/10</td>
<td>90/100</td>
<td>860/1000</td>
<td>860/1000</td>
</tr>
<tr>
<td>Comp. Time (sec)</td>
<td>0.026</td>
<td>1.74</td>
<td>15.9</td>
<td>158</td>
</tr>
</tbody>
</table>

actually avoided the obstacle. Both results are presented in the table. The risk computed by the Monte Carlo simulation converges slowly to the analytic prediction. The computation time for each method on a 2.33 MHz Intel Core 2 Duo processor is shown in the table. The computation time for the Monte Carlo simulation is a function of the simulation step size, which is 0.1 seconds in this example. The analytic solution can be computed several orders of magnitude faster than a Monte Carlo simulation with similar accuracy.

To test the performance of numerical quadrature in estimating error variance due to plant uncertainty, the plant (46) was modified to include parameter error. The three nonzero values of $b$ were assigned normal distributions around the nominal $b$, with standard deviations equal to 20% of the nominal values. To highlight the effect of the parameter error, the process noise was reduced to $W_{\nu} = 10^{-5} I$. Following the procedure outlined in Section IV, the standard deviation of the cross-track error at the end of the motion plan was computed using a full-grid Hermite quadrature. For a quadrature of order $n$, the full grid contains $n^3$ evaluation points. Figure 4 shows the convergence of the standard deviation of the cross-track error for the quadrature (up to $n = 6$) and the Monte Carlo simulation. The quadrature solution converges in fewer evaluation points ($n = 3$, 27 evaluation points, 0.390 seconds) than the Monte Carlo solution (approximately 300 evaluation points, 65 seconds).

VI. CONCLUSIONS

We have presented an analytic solution to the state error variance due to process noise for a holonomic vehicle that moves in a plane. The solution is an explicit function of time that does not require numerical integration of the plant. Through a transformation to a reference-local coordinate system, the error dynamics are described by a Riccati equation with constant coefficients; this equation has a known analytic solution. The result is extended to handle underactuated vehicles and plant uncertainty.

Motion planning algorithms can be improved by considering how uncertainty due to external disturbances and modeling error affects the likelihood of collisions with obstacles. To do this, the planner must be able to predict the error statistics for each plan. The analytic error variance solution developed in this paper eliminates the need for Monte Carlo simulations when computing the risk associated with a particular motion plan. This results in a greatly reduced computational cost for the planning algorithm. The analytic solution can be applied to any motion planner including RRTs, RMPC, or graph search algorithms.

Presently the solution is limited to vehicles moving in a plane with full-state feedback. We hope to extend the result to 6-degree-of-freedom mobile robots and include an estimator for unmeasured states.

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