2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

Robust Motion Planning Methodology for Autonomous Tracked Vehicles in Rough Environment Using Online Slip Estimation

Sang Uk Lee   Karl Iagnemma

Abstract— **This paper presents a robust motion planning methodology for autonomous tracked vehicles navigating in a rough and unknown environment. Two fields of study are dealt with in this paper: motion planning and slip estimation. For the motion planner, the CC-RRT* algorithm is combined with LQG-MP. The motion planner uses a chance-constrained approach and considers the role of compensator in the planning step to provide a robust yet non-conservative planner. For the slip estimator, a stable yet practical online approach known as IPEM is used. IPEM compares the integrated prediction with the measurement to calculate appropriate parameters. The methodology performs online slip estimation and re-planning iteratively. This guarantees the safe travel of the vehicle even when there is an unexpected terrain change that can be fatal. The simulation result shows that the iterative estimation and re-planning plays a significant role in ensuring the safety of the vehicle.**

I. Introduction

Motion planning of autonomous vehicles in rough off-road environments is a challenging task. Numerous factors affect the vehicle system as process noises. Vehicle slip due to vehicle-terrain interaction affects the state of the vehicle significantly. Thus, this phenomenon should be considered with great care, especially when aiming for robust planning. In addition, vehicle-terrain interaction varies greatly across different terrain types [1]. Thus, when we travel through an unknown environment that consists of multiple soil types, we need to consider each soil type separately [2]. Finally, the problem becomes extremely complicated when there is no prior information about the terrain that the vehicle must travel through.

This problem can be resolved through a robust motion planning scheme that uses online slip estimation. The slip estimator calculates the slip property of the soil that the vehicle is currently operating on in an online fashion. Then, the robust motion planner plans the trajectory for the vehicle iteratively based on the most recent slip property calculated by the slip estimator.

The chance-constrained method is a great robust planning strategy that calculates the probability of collision by considering the vehicle state as a random variable rather than as a deterministic one. In addition, RRT* is an incremental sampling-based motion planning algorithm that has been proven very effective. It yields the optimal trajectory for a given cost function [3], even for nonholonomic systems such as the tracked vehicles considered in this paper [4]. However, it involves solving the two-point boundary value problem (TPBVP), which can be demanding [5]. An approximate measure has been introduced to bypass this difficulty [6]. CC-RRT* combines the chance-constrained method with RRT* [7]. It yields a robust and optimal trajectory for the given planning problem. CC-RRT* is usually implemented in an open loop fashion, and the planned trajectory is thus more conservative than necessary. This issue can be resolved using LQG-MP (Linear Quadratic Gaussian Motion Planning) [8], which considers the role of compensator in the planning process to avoid being overly conservative. The resulting algorithm is a non-conservative robust optimal motion planner. The planner was originally introduced in [9], and the current paper is an extension of this earlier work, adding estimation step in order to deal with unknown environment.

The slip estimator needs to calculate the slip property of the unknown terrain. The integrated prediction error minimization (IPEM) method is an effective estimation algorithm [10]. For a given differential system with parameters, IPEM calculates the parameter that minimizes the deviation of prediction from measurement. In addition, it calculates the probabilistic distribution of the minimized deviation, which can be used in the chance constrained motion planning step. To briefly explain the approach, IPEM calculates the predicted state after a certain time using the given differential system equation, and compares it with the measurement at that time for estimation. The fact that it uses integrated prediction rather than derivative directly makes the algorithm more stable and accurate.

This paper provides the overall framework of the motion planning methodology for tracked vehicles operating in a rough environment with unknown slip property. The environment is assumed to be flat and to consist of multiple soil types. A robust motion planner runs iteratively based on the slip property estimate calculated by the online slip estimator. The CC-RRT* algorithm for the nonholonomic vehicle model, combined with the LQG-MP framework, is used as a robust optimal motion planner. The IPEM approach is used to estimate the slip property online. To the authors' knowledge, this paper is the first to use existing work to demonstrate a practical robust optimal motion planning methodology for autonomous vehicles operating in rough unknown environment scenarios. The methodology applied to tracked vehicles can be applied to a wide range of vehicle models, which is another advantage of this paper.

This paper is organized as follows. Section II provides an overview of the iterative planning and estimation methodology. Basic information about the algorithms used is presented in Section III. This includes an explanation about CC-RRT* for nonholonomic systems combined with the LQG-MP method, as well as further details about the IPEM method. The application of the algorithms to the tracked

The authors are with the Robotic Mobility Group, Massachusetts Institute of Technology, Cambridge, MA 02139, USA (e-mail : sangukbo@mit.edu and kdi@mit.edu).

vehicle system is explained in Section IV. This includes an explanation of the tracked vehicle model used for the simulation and how it is applied to the algorithms in Section III. Section V presents simulation results. Finally, the paper is concluded in Section VI.

II. Problem Statement and Iterative Methodology Overview

Let the state space be $X \subset R^n$ and the input space be $U \subset R^m$. The vehicle model can be formulated in the following general form:

$$\dot{z} = f(z, u, p, w) \quad (1)$$

where $z \in X$ is the state vector, $u \in U$ is the (nominal) input vector, p is the parameter used to compensate for the slip, and w is the white process noise with zero mean. p and Σ_w, the covariance matrix of w, vary with respect to the terrain type and they are calculated by the slip estimator. In this paper, p and Σ_w will be referred to as slip estimate. p and w are often considered together with input u to form the following equation:

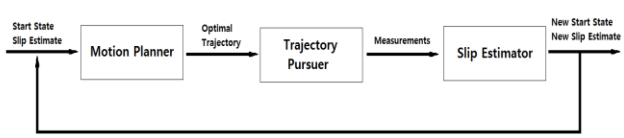
$$\dot{z} = f(z, u_a(p, w)) \quad (2)$$

where u_a is the augmented input.

The vehicle should reach the goal region X_{goal}, starting from the initial state z_{init} and following the optimal and probabilistically feasible trajectory calculated by the motion planner. Probabilistically feasible trajectory means that, given the obstacle state X_{obs}, and the collision free state defined as $X_{free} = X \backslash X_{obs}$, every point in the trajectory given as $\tau : [0, 1] \rightarrow X_{free}$ must always have a smaller probability of collision with obstacles than some threshold value δ, that is, $\mathrm{P}(\tau(s) \subset X_{obs}) < \delta, \forall t, (0 \le s \le 1)$. The optimal path means that the trajectory must be the optimal one with respect to the specified cost function.

Figure 1 and Algorithm 1 show an overview of the entire iterative methodology.

Figure 1. Diagram of overall framework



Algorithm 1: Overall Framework

```
Initialize starting state and slip estimate;
while (vehicle not in X_goal)
    MotionPlanner → Optimal traj.;
    TrajPursuit → Measurements;
    SlipEstimator → Slip estimate;
    New start state; New slip estimate;
return
```

The framework consists of three components: i) planner, ii) trajectory tracker, and iii) slip estimator. Given initial state and slip estimate, the planner finds the optimal trajectory assuming that the entire environment has a homogeneous slip property. The obtained optimal trajectory is passed to the trajectory tracker. The tracker follows the trajectory and meanwhile gathers sensor measurements. For tracked vehicles, these would be GPS and odometry measurements. The slip estimator calculates the new slip estimate based on the measurements and feeds it back to the motion planner along with the new start state. The planner re-plans the optimal trajectory based on the new slip estimate, and the process repeats itself until the vehicle reaches the goal region.

III. Algorithm Overview

A. Motion Planner

The motion planner in this paper combines several state-of-the-art planning methods, [6], [7], and [8]. CC-RRT* [7] is a robust incremental sampling-based motion planning algorithm that yields the optimal and probabilistically feasible trajectory. A brief pseudo code is provided in Algorithm 2.

Algorithm 2: MotionPlanner($z_{start}, \Sigma_{start}, p_{est}, \Sigma_w$)

```
1  V ← {z_start}; E ← ∅;
2  for i = 1, ⋯ , n do
3      z_rand ← Sample(i);
4      Connect to ProbFeas nearest node →
       (z_nearest, u_nearest, Traj_nearest);
5      X_near ← NearVertices(Tree, z_rand);
6      Choose ProbFeas and min cost parent node
       among X_near → (z_min, u_min, Traj_min);
7      if z_rand reduces cost to any nodes in
         X_near and ProbFeas
8          Rewire and Repropagate;
9  return Tree
```

ProbFeas in Lines 4, 5, and 6 checks if the node and trajectory are probabilistically feasible. The algorithm assumes that the state vector is a Gaussian random variable, that is, $z_t \sim N(\hat{z}_t, P_{x_t})$, and all obstacles are convex polyhedrons. The vehicle is inside the polyhedron (and thus colliding with the obstacle) when $\wedge_{i=1}^{n_j} a_{ij}{}^T(z_t - c_{ij}) < 0$ holds, where i stands for i^{th} face of the polyhedron, j stands for j^{th} obstacle, and n_j is the number of faces in the j^{th} polyhedron. The probability of collision with the j^{th} obstacle can be calculated as follows.

$$\mathrm{P}\left(\wedge_{i=1}^{n_j} a_{ij}{}^T\left(z_t - c_{ij}\right) < 0\right) \le \mathrm{P}\left(a_{ij}{}^T\left(z_t - c_{ij}\right) < 0\right) = \Delta_{ijt} = \frac{1}{2}\left(1 - \mathrm{erf}\left(\frac{a_{ij}{}^T(\hat{z}_t - c_{ij})}{\sqrt{2 a_{ij}{}^T(P_{x_t}) a_{ij}}}\right)\right), \forall i \quad (3)$$

$\Delta_{jt} = \min_{i=1,\cdots,n_j} \Delta_{ijt}$ can serve as the upper bound for the probability of collision with the j^{th} obstacle. The ProbFeas function returns "safe" if $\sum_{j=1}^{n_o} \Delta_{jt} < \delta$ for some threshold δ, where n_o is the number of obstacles.

The CC-RRT* algorithm maintains the optimality property even for nonlinear dynamic systems. For this, the algorithm needs to solve the two-point boundary value problem (TPBVP), especially for the "rewire" step. In this paper, an approximate method [6] is used to avoid solving TPBVP exactly, under the assumption of constant input along the trajectory. Due to this assumption, exact connection to the endpoint is impossible. To solve this issue, local steering is going to be considered successful if z_{end} (desired endpoint) and z'_{end} (approximate) are close, that is, $\|z_{end} - z'_{end}\| < \varepsilon$ for some specified threshold ε. Compensation for this is applied with the "repropagation" step shown in Algorithm 3. In this step, all children nodes to z_{end} are regenerated from z'_{end} using inputs and time durations saved previously by calling the "ReppropSteer" function.

Algorithm 3: Repropagate(z, z_{new})

```
for all z' ∈ Children(z) do
    z'_new ← ReppropSteer(z_new, Time(z, z'),
                                    Input(z, z'));
    Repropagate(z', z'_new)
    Delete z' in Tree;
    Add z'_new in Tree;
```

We can limit the degree of uncertainty and implement a less conservative planner by considering the role of compensator, using the framework of LQG-MP [8]. The continuous nonlinear system model can be discretized and linearized as follows:

$$
\begin{gathered}
z_{t+1} = A_t z_t + B_t u_t + C_t + G_t w_t \\
y_{t+1} = H_{t+1} z_{t+1} + v_{t+1} \\
\hat{z}_{t+1} = A_t \hat{z}_t + B_t u_t + C_t + L_t (y_{t+1} - \hat{y}_{t+1}) \\
\hat{y}_{t+1} = H_{t+1} \hat{z}_{t+1} \\
\bar{z}_{t+1} = A_t \bar{z}_t + B_t u_{r,t} + C_t \\
\bar{y}_{t+1} = H_{t+1} \bar{z}_{t+1} \\
u_t = u_{r,t} - K_t (\hat{z}_t - \bar{z}_t)
\end{gathered} \tag{4}
$$

where y_t is the measurement, w_t is the process noise ($w_t \sim N(0, P_{w,t})$), v_t is the sensor noise ($v_t \sim N(0, P_{v,t})$), $\hat{z}_t$ is the estimate of z_t, and $\bar{z}_t$ is the reference state at time t. L_t is the optimal Kalman filter gain and K_t is the optimal feedback controller gain. The distribution of z_t, Kalman gain, and controller gain can be obtained as follows. First, let $Z_t = \begin{bmatrix} z_t \\ \hat{z}_t \end{bmatrix}$. Then, manipulating Equation (3), we get:

$$
\begin{gathered}
Z_{t+1} = \bar{A}_t Z_t + \bar{B}_t u_{r,t} + \bar{C}_t + \bar{G}_t W_t \\
\bar{A}_t = \begin{bmatrix} A_t & -B_t K_t \\ L_{t+1} H_{t+1} A_t & (I - L_{t+1} H_{t+1}) A_t - B_t K_t \end{bmatrix} \\
\bar{B}_t = \begin{bmatrix} B_t \\ B_t \end{bmatrix}, \bar{C}_t \text{ is some constant term} \\
\bar{G}_t = \begin{bmatrix} G_t & 0 \\ L_{t+1} H_{t+1} G_t & L_{t+1} \end{bmatrix}, W_t = \begin{bmatrix} w_t \\ v_{t+1} \end{bmatrix}
\end{gathered} \tag{5}
$$

Then, we get state mean ($E(z_t)$) and covariance (P_t) by:

$$
\begin{gathered}
E(Z_{t+1}) = \bar{A}_t E(Z_t) + \bar{B}_t u_{r,t} + \bar{C}_t, E(z_t) \\
= [1^T \ 0^T] E(Z_t) \\
\bar{P}_{t+1} = \bar{A}_t \bar{P}_t \bar{A}_t^{\,T} + \bar{G}_t \bar{P}_{W,t} \bar{G}_t^{\,T}, P_{t+1} = [I \ 0] \bar{P}_{t+1} \begin{bmatrix} I \\ 0 \end{bmatrix} \\
\bar{P}_t : \text{ cov. of } Z_t, P_t : \text{ cov. of } z_t, \bar{P}_{W,t} : \text{ cov. of } W_t
\end{gathered} \tag{6}
$$

The Kalman gain and optimal controller gain can be obtained using the standard LQG algorithm, which is explained in [8] in more detail.

B. Slip Estimator

The IPEM method is used for the slip estimator [10]. The algorithm calculates the parameter estimate of the system model and the distribution related to the system's stochastic property, based on measurements. In this paper, this method is used to calculate the parameter estimate that minimizes the deviation between the state prediction at time t, $z_{pred}(t)$, and measurement received at time t, $z_{pred}(t)$. In addition, the distribution of process noise, which is for the system's stochastic property, is calculated, assuming that it takes some normal distribution, $w \sim N(0, \Sigma_w)$. It is required in order to use the chance constrained method.

$z_{pred}(t)$ is obtained by integrating the system equation from t_0 to t, using the current parameter estimate, p_{est}:

$$z_{pred}(t) = z_0 + \int_{t_0}^{t} f\left(z_{pred}(\tau), u_a(\tau, p_{est}, 0)\right) d\tau \tag{7}$$

where u_a is the augmented input as in Equation (17). Note 0 in $u_a(\tau, p_{est}, 0)$, since the process noise has zero mean.

Let us put $z_{pred}(t) = g(z(t_0), u_a(\cdot, p_{est}, 0))$. This equation is usually a nonlinear equation for vehicle models. The above equation can be linearized as follows.

$$z_{pred}(t) \approx \left.\frac{\partial g}{\partial p}\right|_{SE} p + \cdots \tag{8}$$

We will use the notation SE (which stands for 'slip estimate') to denote $p = p_{est}, w = 0$. Now, we can approximate the new parameter estimate based on measurements received using the following equation:

$$\left.\frac{\partial g}{\partial p}\right|_{SE} \Delta p \approx z_{mea} - z_{pred} \tag{9}$$

However, obtaining the Jacobian matrix in Equation (9), which can be obtained as applying the difference equation $\frac{\partial g}{\partial p_i} \approx \left.\frac{g(z_{pred}(t_0), u_a(\cdot, p + \delta p_i, w)) - g(z_{pred}(t_0), u_a(\cdot, p, w))}{\varepsilon}\right|_{SE}$ for all matrix elements, is computationally demanding. We take the detour and first linearize the differential system equation.

$$
\begin{aligned}
&\delta \dot{z}_{pred}(t) \\
&\approx \left.\frac{\partial f}{\partial z_{pred}}\right|_{SE} \delta z_{pred}(t) + \left.\frac{\partial f}{\partial u_a}\right|_{SE} \delta u_a(t, p, w)|_{SE} \\
&= F \delta z_{est}(t) + G \delta u_a(t, p, w)|_{SE}
\end{aligned} \tag{10}
$$

For this linearized differential system equation, the solution is the following vector convolution integral:

$$
\begin{aligned}
&\delta z_{pred}(t) \\
&= \Phi(t, t_0) \delta z(t_0) + \int_{t_0}^{t} \Gamma(t, \tau) \delta u_a(t, p, w)|_{SE} d\tau
\end{aligned} \tag{11}
$$

Here, each symbol represents the following matrices.

$$
\begin{gathered}
\dot{\Phi}(t, \tau) = F(t) \Phi(t, \tau) \\
\Psi(t, \tau) = \int_{\tau}^{t} F(\zeta) d\zeta \\
\Phi(t, \tau) = e^{\Psi(t, \tau)} \\
\Gamma(t, \tau) = \Phi(t, \tau) G(\tau)
\end{gathered} \tag{12}
$$

We can approximate the Jacobian matrix $\left.\frac{\partial g}{\partial p}\right|_{SE}$ as follows:

$$\left.\frac{\partial g}{\partial p}\right|_{SE} \approx H_{sys} = \int_{t_0}^{t} \Gamma(t,\tau) \left.\frac{du_a(\tau,p,w)}{dp}\right|_{SE} d\tau \qquad (13)$$

In addition, we need the measurement covariance matrix, the covariance matrix of $r(t) = z_{mea}(t) - z_{pred}(t)$.

$$R_{sys} = \Phi(t,t_0)\Sigma_{x,mea}(t_0)\Phi(t,t_0)^T + \int_{t_0}^{t} \Gamma(t,\tau)Q(\tau)\Gamma(t,\tau)^T d\tau + \Sigma_{x,mea}(t) \qquad (14)$$

where $\Sigma_{x,mea}(t)$ is the covariance matrix of $z_{mea}(t)$ (it is easy to think of it as the GPS error covariance matrix) and $Q(\tau)$ is the covariance matrix of $\delta u_a(\tau, p, w)$.

We can obtain the parameter value using the following Kalman filter framework iteratively:

$$\begin{aligned}
p_{est,k|k-1} &= p_{est,k-1|k-1} \\
\Sigma_{p_{est},k|k-1} &= \Sigma_{p_{est},k-1|k-1} + \Delta_k \\
r_k &= z_{pred,k} - z_{mea,k} \\
S_k &= H_{sys,k}\Sigma_{p_{est},k|k-1}{H_{sys,k}}^T + R_{sys,k} \\
K_{e,k} &= \Sigma_{p_{est},k|k-1}{H_{sys,k}}^T{S_k}^{-1} \\
p_{est,k|k} &= p_{est,k|k-1} + K_{e,k}r_k \\
\Sigma_{p_{est},k|k} &= (I - K_{e,k}H_{sys,k})\Sigma_{p_{est},k|k-1}
\end{aligned} \qquad (15)$$

where k stands for k^{th} iteration. $\Sigma_{p_{est}}$ is the covariance matrix of the parameter estimate and Δ_k is the covariance matrix that governs how much weight we put on the most recent measurement. The larger that Δ_k is, the more weight we put on the recent measurement.

We can also estimate for Σ_w, assuming that it is a constant, although it can vary according to terrain type. This will not be discussed in detail due to limited space, but more information can be found in [10].

IV. SAMPLING-BASED MOTION PLANNING STRATEGY FOR OFF-ROAD TRACKED VEHICLES UNDER SLIP

A. *Tracked Vehicle Model and Cost Function*

In this paper, the following simple kinematic differential drive with augmented input is of interest. Figure 2 shows the diagram of the vehicle.

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}(\beta + w_\beta)\frac{v_r+v_l}{2}cos\theta \\ (\beta + w_\beta)\frac{v_r+v_l}{2}sin\theta \\ (\alpha + w_\alpha)\frac{v_r-v_l}{b}\end{bmatrix} = \begin{bmatrix}\frac{v_{r,a}+v_{l,a}}{2}cos\theta \\ \frac{v_{r,a}+v_{l,a}}{2}sin\theta \\ \frac{v_{r,a}-v_{l,a}}{b}\end{bmatrix} \qquad (16)$$

where $z = [x, y, \theta]^T$ is the state vector, $u = [v_r, v_l]^T$ is the nominal input vector applied to the system, and $u_p = [v_{r,a} \;\; v_{l,a}]^T$ is the augmented input vector. $p = [\alpha \;\; \beta]^T$ is the parameter vector and $w = [w_\alpha \;\; w_\beta]^T$ is the process noise vector which takes a normal distribution, $w \sim N(0, \Sigma_w)$ where Σ_w is a constant matrix. u and u_a satisfy the following equation:

$$u_a = \begin{bmatrix}\frac{(\alpha+w_\alpha)+(\beta+w_\beta)}{2} & -\frac{(\alpha+w_\alpha)-(\beta+w_\beta)}{2} \\ -\frac{(\alpha+w_\alpha)-(\beta+w_\beta)}{2} & \frac{(\alpha+w_\alpha)+(\beta+w_\beta)}{2}\end{bmatrix} u = T_a u \qquad (17)$$

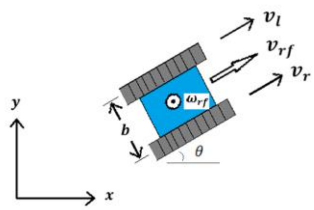
The vehicle model can be considered as the effective wheel base model with an additional parameter β. That is, α compensates for angular slip and β compensates for longitudinal slip.

The cost function to be optimized for the motion planner is as follows:

$$J(X, u) = t_f + \int_0^{t_f} u(t)^T R u(t) dt \qquad (18)$$

where t_f is the final time and R is the cost parameter matrix.

Figure 2. Diagram of simple kinematic differential drive



B. *Implementation of the Motion Planner*

As stated above, the motion planner is the combination of three state-of-the-art works, [6], [7], and [8]. The CC-RRT* will be implemented using Equation (3) to check the probabilistic feasibility of the trajectory.

In addition, the approximation method for solving TPBVP is going to be implemented as follows. Let us consider a case where we are steering the vehicle from z_1 to z_2, satisfying only the (x_2, y_2) coordinate of z_2 under constant input. With the system equation from Equation (16), the trajectory becomes a circular arc:

$$(x(t) - x_1 + rsin\theta_1)^2 + (y(t) - y_1 - rcos\theta_1)^2 = r^2 \qquad (19)$$

where $r = \frac{b\beta(v_r+v_l)}{2\alpha(v_r-v_l)}$ is the radius of the circular arc. The radius can also be obtained purely geometrically and it specifies the velocity ratio of the right and left tracks. We find the velocity and the final time that satisfies the geometric constraint and minimizes the cost function $J(X, u) = t_f + u^T R u \times t_f$. The problem becomes the optimization on polynomial with respect to t_f. The difference between θ_2 and $\theta(t_f)$ can be resolved using the repropagation method discussed in Section III-a.

Finally, the LQG-MP algorithm is also implemented with a slight modification to consider the effect of a compensator. In this paper, the distribution of z_t and Kalman gain, L_t, is obtained as explained in Section III-a. However, in order to avoid solving the matrix Riccati equation each time and reduce the computational load, a suboptimal controller gain is used [11], [12]:

$$K_t = \begin{bmatrix}2\xi\alpha cos\theta_t - \frac{1}{2}\beta v_{rf,t}sin\theta_t & \cdots \\ 2\xi\alpha cos\theta_t + \frac{1}{2}\beta v_{rf,t}sin\theta_t & \end{bmatrix} \qquad (20)$$

$$\begin{matrix} 2\xi\alpha sin\theta_t + \frac{1}{2}\beta v_{rf,t} cos\theta_t & \xi\alpha \\ 2\xi\alpha sin\theta_t - \frac{1}{2}\beta v_{rf,t} cos\theta_t & -\xi\alpha \end{matrix}\Bigg]$$

$$where\ \alpha = \left(\left(\omega_{rf,t}\right)^2 + \beta\left(v_{rf,t}\right)^2\right)^{\frac{1}{2}},$$

$$\beta\ and\ \xi\ are\ tuning\ parameters$$

$$v_{rf,t} = \frac{v_r + v_l}{2}, \omega_{rf,t} = \frac{v_r - v_l}{b}$$

C. Implementation of the Slip Estimator

Now, let us apply the above IPEM method to our model. First, we need the Jacobian matrices $\left.\frac{\partial f}{\partial z_{pred}}\right|_{SE}$ and $\left.\frac{\partial f}{\partial u_a}\right|_{SE}$.

$$F = \left.\frac{\partial f}{\partial z_{pred}}\right|_{SE} = \begin{bmatrix} 0 & 0 & -\dot{y}_{pred} \\ 0 & 0 & \dot{x}_{pred} \\ 0 & 0 & 0 \end{bmatrix}$$

$$G = \left.\frac{\partial f}{\partial u_a}\right|_{SE} = \begin{bmatrix} \frac{1}{2} cos\theta_{pred} & \frac{1}{2} cos\theta_{pred} \\ \frac{1}{2} sin\theta_{pred} & \frac{1}{2} sin\theta_{pred} \\ \frac{1}{b} & -\frac{1}{b} \end{bmatrix} \tag{21}$$

We can calculate $\Phi(t,\tau)$ easily:

$$\Phi(t,\tau) = I + \Psi(t,\tau) = \begin{bmatrix} 1 & 0 & -\Delta y_{pred} \\ 0 & 1 & \Delta x_{pred} \\ 0 & 0 & 1 \end{bmatrix} \tag{22}$$

where $\Delta x_{pred} = x_{pred}(t) - x_{pred}(\tau)$. Using the above,

$$\Gamma(t,\tau) = \begin{bmatrix} \frac{1}{2} cos\theta_{pred} - \frac{\Delta y_{pred}}{b} & \frac{1}{2} cos\theta_{pred} + \frac{\Delta y_{pred}}{b} \\ \frac{1}{2} sin\theta_{pred} + \frac{\Delta x_{pred}}{b} & \frac{1}{2} sin\theta_{pred} - \frac{\Delta x_{pred}}{b} \\ \frac{1}{b} & -\frac{1}{b} \end{bmatrix} \tag{23}$$

Thus,

$$\Gamma(t,\tau) \left.\frac{du_a(\tau,p,w)}{dp}\right|_{SE}$$

$$= \begin{bmatrix} -\frac{\Delta y_{pred}}{b}(v_r - v_l) & \frac{1}{2}(v_r + v_l) cos\theta_{pred} \\ \frac{\Delta x_{pred}}{b}(v_r - v_l) & \frac{1}{2}(v_r + v_l) sin\theta_{pred} \\ \frac{(v_r - v_l)}{b} & 0 \end{bmatrix} \tag{24}$$

We get $H_{sys} = \int_{t_0}^{t} \Gamma(t,\tau) \left.\frac{du_a(\tau,p,w)}{dp}\right|_{SE} d\tau$ by integrating Equation (24). Calculation is simplified due to piecewise constant input assumption. When we obtain R_{sys}, we assume that Σ_w is a constant matrix. The following holds in that case:

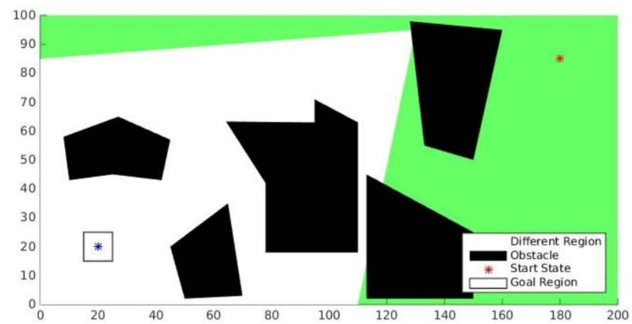
$$Q(\tau) = T_u \Sigma_w T_u^{\ T} \tag{25}$$

where $T_u = \begin{bmatrix} \frac{v_r - v_l}{2} & \frac{v_r + v_l}{2} \\ -\frac{v_r - v_l}{2} & \frac{v_r + v_l}{2} \end{bmatrix}$.

V. Simulation Result

Simulations were conducted in an artificially created environment with several properties. Figure 3 shows the simulated map. There are numerous polygonal obstacles (black polygons) such as sand dunes, ponds, buildings, etc. In addition, the environment is assumed to consist of two terrain types, represented by the white and green (gray) regions in Figure 3. The white region represents a more slippery and unstable terrain (such as an icy region), whereas the green region represents a less slippery and more stable terrain (such as turf). The map size is 200 by 100 meters. The vehicle's starting pose is (180 m, 85 m, π rad) and its goal region is the 10 m × 10 m box around the point (20 m, 20 m).

Figure 3. The map scenario for simulation



The vehicle's trajectory pursuit is also simulated using a closed loop controller. In addition, the vehicle slip is simulated by assigning different parameter values for each terrain. The parameters have mean values that correspond to p from the above context. They also have covariance matrices Σ_w.

Figure 4 shows the simulation result when the slip estimator was not operating. The optimal trajectory was planned using the initial guesses of p and Σ_w. Then, the vehicle tried to follow the trajectory assuming that the initial guesses were correct. The guesses were chosen so that they were close to p and Σ_w of the green region, and the vehicle thus followed the trajectory well within that region. However, when the terrain changed, the vehicle deviated much from the planned trajectory and collided with an obstacle.

Figure 5 shows the result where the slip estimator was operating but the online re-plan was not conducted. The vehicle follows the trajectory much better than it did in Figure 4. This is due to the fact that the closed loop controller could utilize the newly calculated slip estimate. However, since the trajectory was planned using only the initial guesses, the vehicle collided with an obstacle in this scenario as well.

Figure 4. The simulation without slip estimation

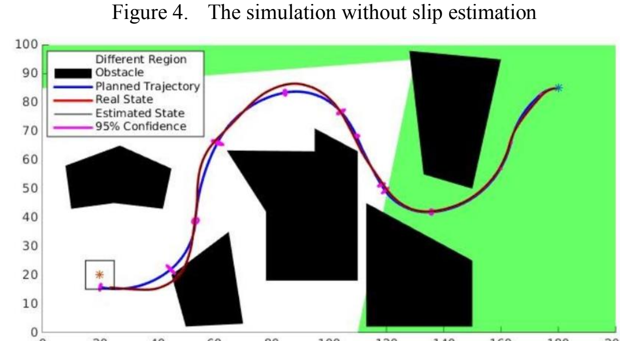


Figure 5. The simulation with slip estimation, but without re-planning

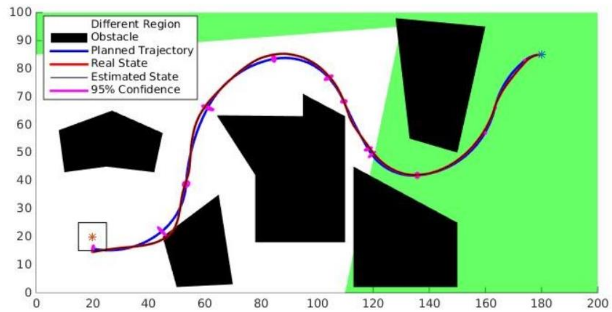


Figure 6 shows result with the entire framework operating. It shows re-planned trajectories for all iterations. The vehicle not only followed the planned trajectory successfully, but it also avoided all the obstacles. This indicates that the re-planning plays a crucial role in the overall process. Figure 7 shows the calculated parameter values (p) for the Figure 5 case (left) and the Figure 6 case (right). The true parameters used were $p_1 = [0.9\ 1.0]^T$ for the green region and $p_2 = [0.5\ 0.8]^T$ for the white region, they are indicated as dashed horizontal lines. The vertical line indicates entering different region. The true parameters were calculated correctly and the terrain change was successfully detected. The high frequency noise is due to the measurement noise and Σ_w. The sum of deviation of vehicle's position from reference trajectory for every 0.3 second was calculated. Table 1 shows the ratio of each case with respect to the iterative method case.

Figure 6. The simulation with slip estimation and re-planning

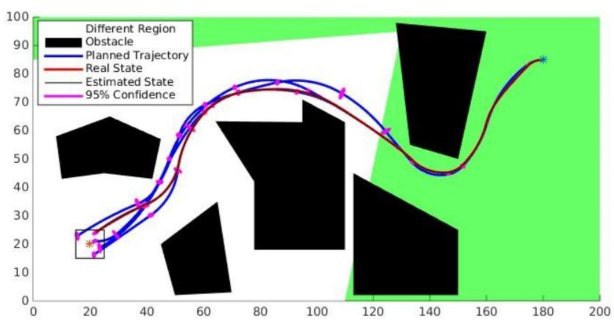


Figure 7. The parameter estimate for Figure 5 (left) and Figure 6 (right)

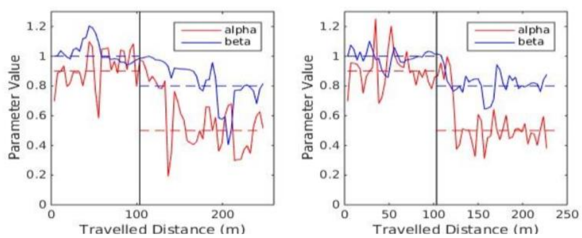


TABLE I. RATIO OF SUM OF VEHICLE'S DEVIATION (WITH RESPECT TO THE ITERATIVE METHOD)

Scenario	No estimation	No re-plan	Iterative method
Ratio of Sum of Dev.	5.6934	4.1405	1

VI. CONCLUSION

This paper demonstrates an iterative methodology that combines motion planning and online slip estimation in the context of robust planning for tracked vehicles operating in a rough and unknown environment. Online slip estimation allows the vehicle to predict any undesirable noise component beforehand and plan and pursue the trajectory accordingly. Specifically, it plays a crucial role when there is an abrupt change in terrain condition. The concept described in this paper can be applied to exploration problems, such as, planetary rovers, where detecting and estimating different terrains in unknown environments and avoiding obstacles at the same time is the major goal.

Future efforts will focus on providing experimental results for the methodology, which has so far only been supported by computer simulations.

ACKNOWLEDGMENT

The research described in this publication was carried out at the Massachusetts Institute of Technology, under the Army Research Project Grant W911NF-13-1-0063 funded by Army Research Laboratory.

REFERENCES

[1] J. Y. Wong, *Theory of Ground Vehicles*, 4th edition, Wiley, 2008.

[2] J. R. Fink and E. A. Stump, "Experimental Analysis of Models for Trajectory Generation on Tracked Vehicles," *IEEE International Conference on Intelligent Robotics and Systems (IROS)*, September 2014, pp. 1970-1977.

[3] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning," *International Journal of Robotics Research*, Vol. 30 No. 7, June, 2011, pp. 846-894.

[4] S. Karaman and E. Frazzoli, "Optimal Kinodynamic Motion Planning using Incremental Sampling-based Methods," *IEEE Conference on Decision and Control (CDC)*, December 2010, Atlanta, GA, USA, pp. 7681-7687.

[5] D. J. Webb and J. van den Berg, "Kinodynamic RRT* : Asymptotically Optimal Motion Planning for Robots with Linear Dynamics," *IEEE International Conference on Robotics and Automation (ICRA)*, May 2013, Karlsruhe, Germany, pp. 5054-5061.

[6] J. H. Jeon, S. Karaman, and E. Frazzoli, "Anytime Computation of Time-Optimal Off-road Vehicle Maneuvers using the RRT*," *IEEE Conference on Decision and Control (CDC)*, December 2011, Orlando, FL, USA, pp. 3276-3282.

[7] B. D. Luders, S. K. Karaman, and J. P. How, "Robust Sampling-based Motion Planning with Asymptotic Optimality Guarantees," *AIAA Guidance, Navigation, and Control Conference (GNC)*, August 2013, Boston, MA, USA, pp. 1-25.

[8] J. van den Berg, P. Abbeel, and K. Goldberg, "LQG-MP: Optimized Path Planning for Robots with Motion Uncertainty and Imperfect State Information," *International Journal of Robotics Research*, Vol. 30 No. 7, June, 2011, pp. 895-913.

[9] S. U. Lee, R. Gonzalez, and K. Iagnemma, "Robust Sampling-based Motion Planning for Autonomous Tracked Vehicles in Deformable High Slip Terrain," *IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, Stockholm, Sweden, Accepted.

[10] N. Seegmiller, F. Rogers-Marcovitz, G. Miller, and A. Kelly, "Vehicle Model Idenfitication by Integrated Prediction Error Minimization," *International Journal of Robotics Research*, Vol. 32, No. 8, July, 2013, pp. 912-931.

[11] R. Gonzalez, F. Rodriguez, and J. L. Guzman, *Autonomous Tracked Robots in Planar Off-road Conditions – Modelling, Localization, and Motion Control*, International Publishing Switzerland, Springer, 2014.

[12] Y. Kanayama, Y. Kimura, F. Miyazaki, T. Noguchi, "A Stable Tracking Control Method for an Autonomous Mobile Robot," *IEEE International Conference on Robotics and Automation*, May 1990, Cincinnati, OH, USA, pp. 384-389.