Learning of Skid-Steered Kinematic and Dynamic Models for Motion Planning

Camilo Ordonez[a], Nikhil Gupta[a], Brandon Reese[a], Neal Seegmiller[b], Alonzo Kelly[b], Emmanuel G. Collins, Jr.[a]

[a] *Center for Intelligent Systems, Controls and Robotics (CISCOR)*
Department of Mechanical Engineering
Florida A&M University-Florida State University
Tallahassee, Fl 32310
[b] *Robotics Institute*
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Modeling of the motion of a skid-steered robot is challenging since slippage and skidding is inherent to this type of platform and it requires high torques to perform curvilinear motion. If the ground-robot interaction and torque requirements are not captured properly, motion planners will sometimes generate trajectories that are not achievable by the robot. Important motion planning applications that rely heavily in these models, include energy efficient and momentum based planning. However, these models change as the terrain surface varies. To cope with this issue, this paper presents a methodology to perform online learning of such models. It combines detailed slip and terramechanic-based dynamic models of wheel-terrain interaction with online learning via Extended Kalman filtering (to update the kinematic model) and an efficient neural network formulation (to update the dynamic model). The proposed approach experimentally demonstrates the importance of the joint utilization of the learned vehicle models in the context of energy efficient motion planning. In particular, the slip-enhanced kinematic models are used to efficiently provide estimates of robot pose and the dynamic models are

Email addresses: cordonez@fsu.edu (Camilo Ordonez), ng10@my.fsu.edu (Nikhil Gupta), bmr09f@my.fsu.edu (Brandon Reese), n.seegmiller@gmail.com (Neal Seegmiller), alonzo@cmu.edu (Alonzo Kelly), ecollins@eng.fsu.edu (Emmanuel G. Collins, Jr.)

Preprint submitted to Robotics and Autonomous Systems *April 7, 2017*

© 2017. This manuscript version is made available under the Elsevier user license
http://www.elsevier.com/open-access/userlicense/1.0/

employed to generate energy estimates and minimum turn radius contraints.

Keywords: wheel-terrain interaction, online learning, skid-steered robots, energy efficient planning

---

1. Introduction

Skid-steered vehicles are one of the most common types of platforms employed in field robotics due to their mechanical robustness and simplicity. A major challenge with these platforms is the complexity of the wheel-terrain interaction models required to capture the surface friction as the robot performs curvilinear motion. The complexity of these models originates in the fact that the wheels or tracks of a skid-steered platform must slip and/or skid when performing a turning maneuver.

As illustrated in Fig. 1, the slippage observed in these type of vehicles has several implications for motion planning applications: first, it causes the vehicle to severely understeer when compared to a differential drive robot with the same dimensions (i.e., same wheel radius and trackwidth), which induces localization errors and/or obstacle collisions [1]. Second, when the vehicle performs sharp turns, large turning resistance moments appear and the motors are forced to apply high wheel torques [1]. This situation can easily lead to actuator saturation and therefore limit the minimum turn radius achievable on a given surface [2]. It is very important to respect this minimum turn radius constraints as otherwise vehicle commands will be poorly tracked by the low level robot control system. In addition, the high wheel torques required to turn result in increased energy expenditure, which is not desirable for long term missions. An additional complication is the fact that the minimum turn radius and energy expenditure are highly sensitive to the terrain being traversed and the vehicle payload.

In order to do motion planning that respects actuator constraints and accounts for slip and energy consumption, it is necessary to obtain vehicle models that capture these phenomena. In addition, since these robots operate in diverse terrain surfaces, the developed models should be able to adapt in an online fashion. Furthermore, strategies to integrate such models with motion planners need to be devised. This paper focuses on both kinematic model and dynamic model learning and integration of the developed models in energy efficient motion planning applications. If these models are ignored, robot trajectories generated by the motion planner will often be kinematically

or dynamically infeasible and therefore the system will have to use additional resources to perform extensive replanning. Although constant replanning can yield feasible trajectories, these trajectories will usually be far from optimal.

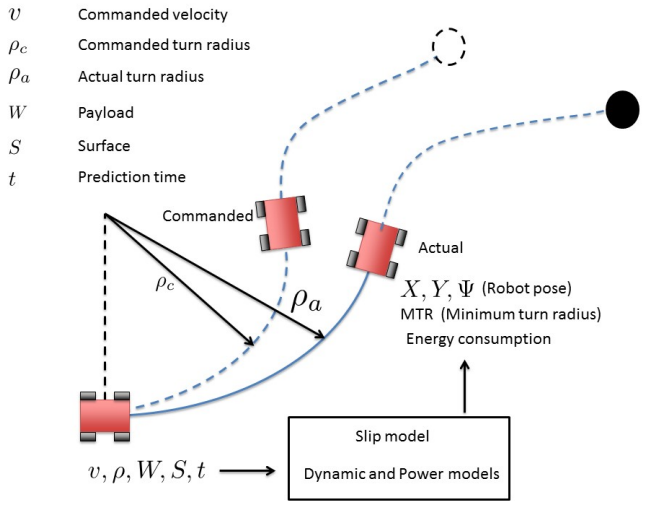


Figure 1: Properly calibrated slip, dynamic, and power models are necessary to predict robot pose (X, Y, Ψ), minimum turn radius MTR, and energy expenditure for a given maneuver, payload, and terrain surface. Notice the difference between the commanded and actual robot trajectories. Through modeling, these discrepancies can be accounted for while planning.

A kinematic model of a skid-steered wheeled vehicle maps the wheel velocities to the vehicle velocities and is an important component in the development of a dynamic model. In contrast to the kinematic models for Ackerman-steered and differential-steered vehicles, the kinematic model of a skid-steered vehicle is dependent on more than the physical dimensions of the vehicle, since it must take into account vehicle sliding and is hence terrain-dependent [3]. In [3, 4], a kinematic model of a skid-steered vehicle was developed by assuming a certain equivalence with a kinematic model of a differential-steered vehicle. This was accomplished by experimentally determining the instantaneous centers of rotation (ICRs) of the sliding velocities of the left and right wheels. An alternative kinematic model that is based on the slip ratios of the wheels has been presented in [5]. This model takes into account the longitudinal slip ratios of the left and right wheels. The difficulty in using this model is the actual detection of slip, which cannot be computed analytically.

Most research work in modeling dynamics of skid-steered platforms has been based on simplified Coulomb friction models [6, 7]. However, as shown in [8], Coulomb friction models fail to capture the relationship between torque and turn radii for general curvilinear motion.

In recent years terramechanic-based dynamic models have been developed for skid-steered vehicles [1, 2, 9]. These models rely on proper characterization of slip to predict tractive forces. A virtue of such models is that they generalize easily to changes in vehicle payload. Different from the work here proposed, [1, 2] did not perform online learning of vehicle models and did not considered curvilinear energy efficient motion planning.

Related research has investigated parameter identification methodologies for some of these models [10, 11, 12, 13]. This includes the online estimation, via a Newton Raphson method, of a lumped parameter terramechanic force-slip model for a tracked vehicle moving on soft ground [10]. However, that particular model assumed straight-line motion. Another approach has considered the estimation of two soil parameters (cohesion and internal friction angles) employed on terramechanic models for deformable terrain [11]. The methodology relied on a recursive least squares approach and was validated in a controlled single wheel setup. Alternative formulations to estimate tractive forces on skid-steered vehicles have relied on assumed vehicle dynamic models and Kalman filtering to estimate random wheel-terrain interaction forces as a function of slip [12]. However, that work did not include a model to predict slip, which makes the results more suitable for control than planning. Another important control result is presented in [14], which proposed an adaptive control algorithm to simultaneously estimate the parameters of a pseudo-static friction model and control a skid-steered robot to follow a desired trajectory. This approach could be run in parallel with the work here proposed and used to trigger our online learning algorithm when it detects a significant change in the estimated friction parameters.

Additional relevant work, which could be used in our proposed methodology to enhance vehicle position estimation, consists in the utilization of inertial measurement units to predict motion of skid-steered robots [15]. More recently, a new methodology named Integration Prediction Error Minimization (IPEM) was developed [13]. This methodology performs online calibration of vehicle kinematic models via Extended Kalman filtering and has been shown capable of calibrating deterministic and stochastic models of the forward, lateral, and angular slips for skid-steered robots. Due to its generality, convenient formulation for performing forward simulations, and

ability to work in realtime, the IPEM algorithm is incorporated as part of the approach presented here. It is important to note that the work of [13] did not learned the vehicle dynamics here presented and did not include the energy efficient motion planning of this work.

In this work a novel method to adaptively identify the friction component of the dynamic model of skid-steered vehicles is proposed. The approach combines terramechanic-based and data driven models as a way to exploit the benefits of both modeling methodologies. In particular, terramechanic-based models are used here to provide sound initial conditions to data driven models. On the one hand, terramechanic models offer good generalization to changes in physical quantities such as the vehicle payload and dimensions of the tire contact patch. Also, they allow extrapolation of torque predictions with respect to vehicle commands such as turn radii and velocities. However, these models assume very specific formulations depending on the type of soil and tires. For example very different parameter sets are required to model wheel terrain interaction on soft vs. hard ground [9]. In addition, they tend to be too complex to use for online learning. On the other hand, data driven approaches are not bound to a particular terramechanic model, and if properly formulated, they offer the possibility of efficient adaptation, which is a desirable feature for long and changing field applications. Examples of expected changes during a robot mission include change in terrain type (concrete to asphalt or to grass) and surface condition (dry to wet). A disadvantage of data driven formulations is that in general they are only valid if used with the robot operating within conditions similar to those employed during learning.

Besides the learning of terrain dependent models, an important contribution of the paper is the development and experimental verification of a methodology that combines, through motion planning (e.g., energy efficient motion planning), the calibrated slip-enhanced kinematic and dynamic models. While in principle it would be possible to use the robot dynamics to predict robot position over time, here we take a simpler and more computationally efficient approach. As depicted in Fig. 2 both models have well defined tasks within the motion planning framework. On the one hand, the velocity driven slip model is integrated forward in time and provides the motion planner with efficient and accurate estimates of robot position. On the other hand, the robot dynamic model interacts with the motion planner in two different ways. First, it provides estimates of cost of traveling (energy: time integral of power requirement) by computing wheel torques and power

requirements as a function of terrain type and robot commands (velocity and turn radius). Second, it generates estimates of the minimum turn radius (MTR) that is achievable on a given terrain before an actuator saturates and unpredictable robot motion occurs.

The remainder of the paper is structured as follows: Section 2 describes the experimental platform, its nominal kinematic, slip-enhanced, and dynamic models. Section 3 details the proposed online learning approach. Section 4 presents experimental results together with a demonstration of the utilization of the learned models in an energy efficient motion planning application. Finally, Section 5 contains concluding remarks and directions for future work.

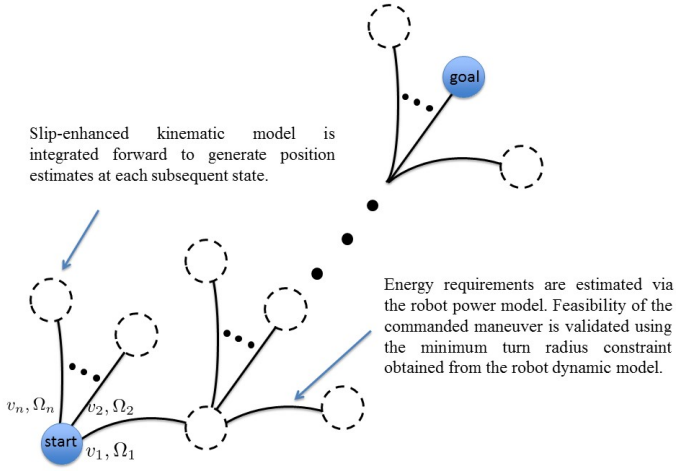


Figure 2: Integration of slip-enhanced kinematic and dynamic models in energy efficient motion planning. The slip model generates localization prediction and the dynamic model generates energy estimates and minimum turn radius constraints. Refer to Section 4.3 for more details about Sampling Based Model Predictive Optimization (SBMPO) and the computation of power estimates.

2. Experimental Platform, Kinematic and Dynamic Model

This Section presents a brief description of the experimental platform and summarizes the terramechanic model for general curvilinear motion.

The FSU Bot, shown in Fig. 3, is a skid-steered robotic platform powered by two mechanically coupled DC motors per side. The robot is controlled by a

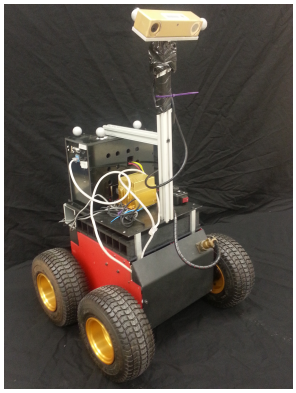


Figure 3: The FSU Bot.

QNX realtime operating system at a rate of $1KHz$. Its sensor suite includes an IMU NAV440 and a visual odometry system, which runs at $10Hz$ and was developed at the Jet Propulsion Laboratory. The robot's top speed is $1m/s$ and the maximum torque per side is controlled via software and was set to $4.6Nm$. Table 1 lists the main parameters of the vehicle, motor, motor controller, and a nominal vinyl surface.

2.1. Nominal Kinematic Model

In this research, the nominal kinematic model for a skid-steered robot neglects slippage and is assumed to be the same as a differential drive robot with the same track width. In other words, the nominal kinematic model assumes that no slip is required for turning. Following [16], this model is given by

$$\begin{bmatrix} v_y \\ \Omega_z \end{bmatrix} = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ -\frac{r}{B} & \frac{r}{B} \end{bmatrix} \begin{bmatrix} \omega_l \\ \omega_r \end{bmatrix}, \tag{1}$$

where v_y and Ω_z are the forward and angular velocities of the vehicle and ω_l and ω_r are the angular velocities of the wheels corresponding to the left and right sides of the vehicle. The left wheels become the inner wheels for a CCW (counter clockwise) turn and the outer ones for a CW (clockwise) turn. B corresponds to the vehicle track width, and r is the wheel radius. Section 2.2 discusses the slip calibration of the nominal vehicle kinematics. This enhances the model to generate estimates of forward, lateral, and angular robot velocities provided its commanded velocities.

Table 1: Parameters of the FSU Bot

Vehicle		
Weight (N)	W	175.52
Track width (m)	B	0.39
Wheel base (m)	L	0.27
Contact patch length (m)	p_l	0.023
Contact patch width (m)	b	0.028
CG offset in x direction (m)	C_x	-0.0014
CG offset in y direction (m)	C_y	0.0030
CG height (m)	h	0.1208
Radius of tire (m)	r	0.1075
Motor		
Torque constant (Nm/A)	K_t	0.023
Speed constant (rad/sV)	K_n	43.478
Gear ratio	g_r	49.8
Efficiency	η	0.76
Motor controller		
Supply voltage (V)	V_{cc}	16.8
Maximum pulse width modulation	PWM_{max}	0.9
Voltage drop (V)	V_{drop}	2
Vinyl Surface		
Coefficient of friction	μ_o	0.3119
Coefficient of friction	μ_i	0.2405
Shear deformation modulus (m)	K	0.0004

Two different friction coefficients μ_o and μ_i are employed for the inner and outer sides of the vehicle to capture asymmetries between the left and right wheels.

2.2. Calibration of Vehicle Kinematics

Here, the vehicle slip-enhanced kinematics are approximated by

$$\begin{aligned} v_y &= v_c + \delta v_y, \\ v_x &= \delta v_x, \\ \Omega &= \Omega_c + \delta\Omega, \end{aligned} \tag{2}$$

where v_y, v_x, and Ω are the longitudinal, lateral, and angular velocities. Similarly, v_c and Ω_c are the robot commanded longitudinal and angular velocities. The terms δv_y, δv_x, and $\delta\Omega$ correspond to longitudinal, lateral, and angular slip velocities, which are in turn modeled as polynomial functions of the vehicle commands and are given by

$$\begin{aligned} \delta v_y &= a_{11} v_c + a_{12}\Omega_c + a_{13} v_c\Omega_c, \\ \delta v_x &= a_{21} v_c + a_{22}\Omega_c + a_{23} v_c\Omega_c, \\ \delta\Omega &= a_{31} v_c + a_{32}\Omega_c + a_{33} v_c\Omega_c, \end{aligned} \tag{3}$$

where the coefficients a_{ij} are calibrated online and in real time using the IPEM algorithm as the robot is commanded with random velocities and turn radii.

The coefficients a_{ij} where selected to capture the dependence of slip on the commanded robot velocities (forward and angular). In addition, slip depends on contact forces such as the centripetal force ($\frac{v_c}{\rho} = v_c\Omega_c$) so this term is included by using coefficient a_{13}, a_{23}, and a_{33}. Experimental evidence presented in [13] has extensively validated this slip formulation. In this paper, we focus mainly on level ground. However, work by some of the coauthor [13] has examined different formulations of the slip model. Including one that can handle significant robot attitude variation coming from steep slopes.

It is relevant to note that for skid-steered robots the most critical term of the slip-calibrated model (2) is the angular slip $\delta\Omega$ as it accounts for the severe understeering effect observed in these types of platforms.

It is relevant to mention that the slip calibrated estimates of angular and forward velocities are then used to obtain the turn radius, which is a required input to the dynamic model of section 2.3.

2.3. Nominal Terramechanic Model

In contrast to dynamic models described in terms of the velocity vector of the vehicle [6, 7], the dynamic models here are described in terms of the

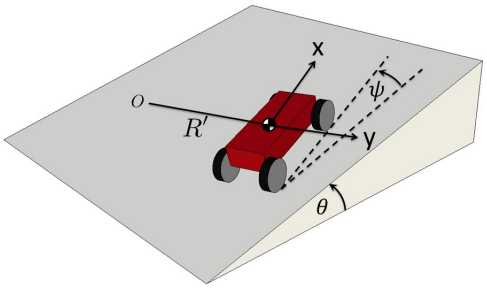


Figure 4: Skid-steered vehicle turning with a radius R' on a hill with slope θ. The heading angle ψ is also illustrated.

angular velocity vector of the wheels. This is because the wheel (specifically, the motor) velocities are actually commanded by the control system; therefore, this model form is particularly beneficial for control and planning.

The dynamic model for a skid-steer vehicle performing a turning maneuver on a slope (depicted in Fig. 4) [2] can be expressed in terms of wheel states and the profile of the local terrain as

$$M\ddot{q}_1 + C(\dot{q}_1, \ddot{q}_1) + G(q_2) = \tau, \tag{4}$$

where $q_1 = [\theta_i \ \theta_o]$ is the angular position of the inner and outer wheels of the vehicle, $\dot{q}_1 = [w_i \ w_o]$ are the wheel velocities, $q_2 = [\theta \ \psi]$ represent the slope of the local terrain and the heading of the vehicle, $\tau = [\tau_i \ \tau_o]$ are the inner and outer motor torques, M is the mass matrix, $C(\cdot)$ is the frictional term and $G(\cdot)$ is the gravitational term. The mass matrix is a function of the vehicle mass and moment of inertia around the vertical axis, which are here assumed known. As shown below, the gravitational term is easily computed as it is only a function of the terrain profile. On the other hand, the frictional term $C(\cdot)$ is difficult to model and is highly dependent on the terrain surface and wheel material. Thus our modeling and learning efforts focus on this term. Even though $C(\cdot)$ is dependent on wheel velocities and accelerations, existing terramechanic-based models assume steady state turns and therefore ignore this dependence [9]. Furthermore, this assumption is justified by the fact that our primary motion planning application is energy efficient planning, which promotes small accelerations [17].

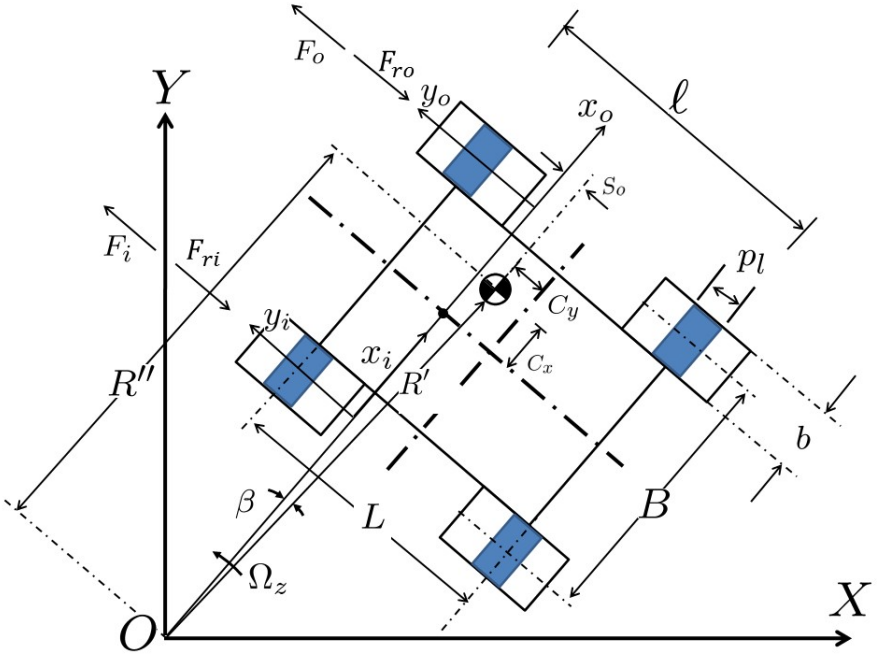


Figure 5: Top view of a skid-steered vehicle performing a steady state turning maneuver with angular velocity Ω_z and turn radius R'. The diagram illustrates the general case with a CG located at (C_x, C_y) and a shifted center of turn by an amount $S_o + C_y$ from the transverse vehicle center line. F_i and F_o are the tractive forces and F_{ri} and F_{ro} are the resistance forces. The contact patches on the ground are represented by the shaded rectangles.

As described in [9], the shear stress τ_{ss} under the contact patch can be computed as a function of the shear displacement ϵ as follows:

$$\tau_{ss} = p\mu(1 - e^{-\epsilon/K}), \tag{5}$$

where p is the normal pressure, μ is the coefficient of friction and K is the shear deformation modulus. Following [2] and refering to Fig. 5, it is possible to estimate the tractive inner force F_i and outer force F_o through integration

of the shear stress over the contact patches as follows:

$$\begin{aligned} F_i = &\int_{n_a-p_l}^{n_a}\int_{-\frac{b}{2}}^{\frac{b}{2}} p_{if}\mu_i(1-e^{-\epsilon_{if}/K})\sin(\pi+\gamma_i)dx_i dy_i \\ &+\int_{n_b-p_l}^{n_b}\int_{-\frac{b}{2}}^{\frac{b}{2}} p_{ir}\mu_i(1-e^{-\epsilon_{ir}/K})\sin(\pi+\gamma_i)dx_i dy_i, \end{aligned} \tag{6}$$

$$\begin{aligned} F_o = &\int_{n_a-p_l}^{n_a}\int_{-\frac{b}{2}}^{\frac{b}{2}} p_{of}\mu_o(1-e^{-\epsilon_{of}/K})\sin(\pi+\gamma_o)dx_o dy_o \\ &+\int_{n_b-p_l}^{n_b}\int_{-\frac{b}{2}}^{\frac{b}{2}} p_{or}\mu_o(1-e^{-\epsilon_{or}/K})\sin(\pi+\gamma_o)dx_o dy_o, \end{aligned} \tag{7}$$

where $n_a = \frac{l}{2} - C_y - S_o$, $R'' = R'cos(\beta)$, $l = 2(\frac{L}{2} + \frac{p_l}{2})$; the terms ϵ_{if}, ϵ_{ir}, ϵ_{of}, and ϵ_{or} are the shear displacements for the inner front, inner rear, outer front, and outer rear wheels at a point (x_i, y_i) on the contact patch; γ_i and γ_o are the angles between the resultant sliding velocities of the inner and outer wheels and the lateral direction of the vehicle. It is important to note that as shown below, the shear displacements and sliding velocities are dependent on slippage, which implies that in order to utilize the terramechanic-based dynamic model, it is necessary to obtain a calibrated slip model. For completeness, here we detail the computation of the shear displacement for the inner-front wheel and the corresponding sliding velocity. For details on the calculations for the other wheels, refer to [2]. The shear displacements for the inner front wheel at a point (x_i, y_i) are given by:

$$\begin{aligned} \epsilon X_{if} = (R'' - \frac{B}{2} - C_x + x_i)&\left\{\cos\left(\frac{(n_a-y_i)\Omega_z}{rw_i}\right)-1\right\} \\ &-y_i\sin\left(\frac{(n_a-y_i)\Omega_z}{rw_i}\right), \end{aligned} \tag{8}$$

$$\begin{aligned} \epsilon Y_{if} = (R'' - \frac{B}{2} - C_x + x_i)&\sin\left(\frac{(n_a-y_i)\Omega_z}{rw_i}\right) - (\frac{-l}{2} - C_y - S_o) \\ &+y_i\cos\left(\frac{(n_a-y_i)\Omega_z}{rw_i}\right), \end{aligned} \tag{9}$$

$$\epsilon_{if} = \sqrt{jX_{if}^2 + jY_{if}^2}, \tag{10}$$

The inner sliding velocity is given by

$$V_{ji} = (V_{jxi}, V_{jyi}), \tag{11}$$

where $V_{jxi} = -y_i\Omega_z$ and $V_{jyi} = (R'' - \frac{B}{2} - C_x + x_i)\Omega_z - rw_i$. Then, the angle γ_i between the resultant sliding velocity of the inner side and the lateral direction of the vehicle can be computed as

$$\gamma_i = \arctan(\frac{V_{jyi}}{V_{jxi}}). \tag{12}$$

It is important to reemphasize that in (8) and (9), w_i represents the wheel velocity as predicted by the inverse of the nominal kinematic model (1). On the contrary, Ω_z represents the actual robot angular velocity, which is predicted by the slip calibrated vehicle kinematic model detailed in section 2.2.

The total resistance term is then given by

$$C(\dot{q_1}) = r \begin{bmatrix} F_i + F_{ri} \\ F_o + F_{ro} \end{bmatrix}, \tag{13}$$

where F_{ri} and F_{ro} represent the combined resistance due to rolling and the drive train friction. Refering to Figs. 4 and 5, the gravitational term can be computed as follows:

$$G(q_2) = \frac{rW \sin\theta \cos\psi}{B} \left[\frac{B}{2} - C_x, \frac{B}{2} + C_x\right]^T, \tag{14}$$

where C_x is the lateral offset of the center of gravity (CG) and W is the vehicle weight.

In this work, a vinyl surface is utilized to obtain the nominal terramechanic-based dynamic model of the vehicle. The identified surface parameters for this surface were $[\mu_o\ \mu_i\ K] = [0.3119\ 0.2405\ 0.0004m]$, where μ_o and μ_i correspond to the outer and inner coefficients of friction and K is the shear deformation modulus. These parameters were determined by conducting a series of experiments in which the robot was commanded to follow constant turn radii in the set $\rho = \{0.2, 0.3, 0.4, 0.5, 1, 1.5, 2, 3, 4, 5, 7, 10, 20, 50, 100, 300, 500,$ $700, 1000\}m$ at a longitudinal velocity of $0.2m/s$. Each run lasted for $8s$ and motor torques were logged at a rate of $1KHz$. The torque measure-

ments were then used to find the model parameters using the nonlinear least squares routine (lsqnonlin) of Matlab. For exhaustive experimental verification of Eq. (4)-(14), refer to [1, 2]. For completeness, Fig. 6, shows a comparison of wheel torque measurements and torque predictions generated by the terramechanic model.

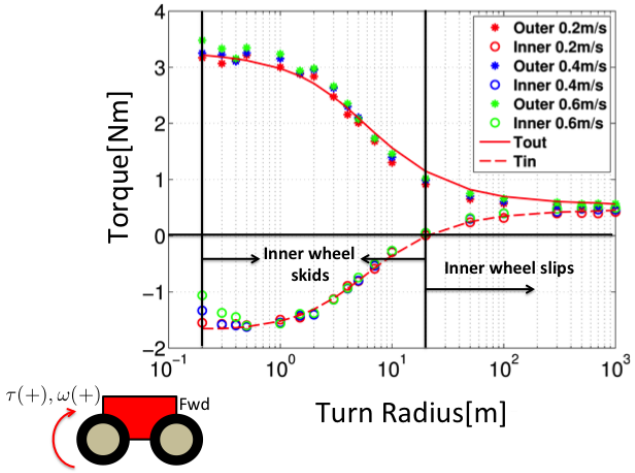


Figure 6: Inner and outer wheel torques for steady state CCW turns. Experimental data was obtained at 0.2, 0.4 and 0.6m/s. The torque predictions (solid and dashed lines) are generated for 0.2m/s. Notice the different regime of operations (skid/slip) for the inner side wheel as a function of the turn radius. For each vehicle wheel, if the wheel linear velocity computed using the angular velocity of the wheel is larger than the actual linear velocity of the wheel, slipping occurs, while if the computed wheel velocity is smaller than the actual linear velocity, skidding occurs.

Assuming steady-state turns, the nominal dynamic model (4) becomes $C(\dot{q}_1) + G(q_2) = \tau$. This model can be used to predict torques as a function of vehicle turn radius and speed. In order to evaluate the frictional term, the first step is to pass the vehicle commands through the slip-calibrated kinematic model (see section 2.2) to obtain an estimate of the actual robot velocities. Using a previously calibrated kinematic model on the vinyl surface and the surface parameters listed above, it is possible to generate the torque vs. turn radius predictions presented in Fig. 6. From this graph the following observations can be made: 1) torque demands are significantly higher for sharper turns and approach a value necessary to overcome the rolling resistance as the turn radius increases; 2) the outer wheel always slips (i.e., traction is always applied.). However, as shown in Fig. 6, the inner wheel

operates in different regimes depending on the turn radius. In particular, it skids for radii greater than half the track width (0.245m) and less than 20m (negative torque and positive wheel velocity). It slips for radii greater than 20m where the torque becomes positive and the wheel velocity is positive. For radii smaller than half the track width (not shown in the figure), the inner wheels slip (negative torque and negative wheel velocity). See Fig. 6 for a sign convention for torques and wheel velocities. A third important observation is that the experimental torques do not vary significantly with speed except for very sharp turns. Therefore, we make the assumption that for the range of speeds being considered in this work, the wheel torques are primarily a function of turn radius. Similar observations regarding the low dependence of torques on certain speed ranges have been reported on tank sized, tracked vehicles [9].

As detailed in Section 4.3, the dynamic model is required to estimate power consumption and generate energy predictions as a function of commanded turn radius and robot velocity. Figure 7, shows power predictions corresponding to curvilinear motion on a vinyl surface at speeds of $0.2m/s$ and $0.6m/s$.

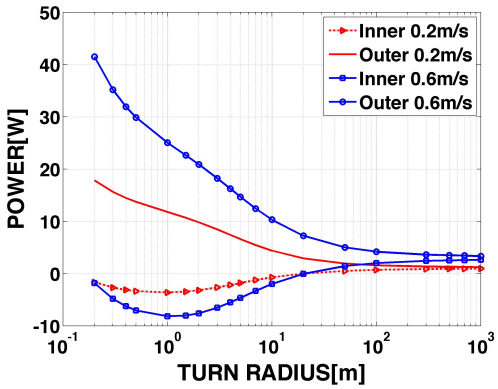


Figure 7: Inner and outer wheel power for steady state CCW turns on vinyl surface. Predictions are generated using robot linear speeds of $0.2m/s$ and $0.6m/s$. Negative power corresponds to a regime of operation where the inner side motors act as generators, but this energy is not captured in the experimental aparatus used in this research.

3. Online Learning Methodology

The overall methodology is illustrated in the block diagram of Fig. 8. In summary, the approach starts by querying terramechanic-based and slip models from a previously traversed surface for a set of initial conditions (i.e., torque predictions for various turn radii). These predictions are used to train a neural network in an offline manner to capture the terrain dependent term $C(\dot{q}_1)$ in (4). Then, when the robot traverses an unknown surface, the IPEM algorithm is used to calibrate the robot kinematic model (i.e., learn a slip model) and the MRAN algorithm (see Section 3.1) is used to adapt the neural network. As described in section 4.3, the calibrated models provide key predictions required by the motion planner.

It is important to emphasize that learning employed by this methodology does not rely on good initial conditions and they are employed only when available. As shown later in section 4.2, the initial conditions can differ significantly from the actual terrain being navigated. Therefore, the approach works on unknown surfaces. However, the methodology does rely on the employment of diagnostic trajectories to ensure that the dynamics of the system are properly excited. In a field ready robotic system, the diagnostic trajectories would be triggered by a perception system when a surface perceptually dissimilar to the surface being navigated is encountered. Since all the algorithms used are recursive, real-time learning can be triggered at any point in the robot mission to update the enhanced kinematic and dynamic models

Notice in Fig. 8 that the approach includes an additional module that performs offline nonlinear least squares optimization to map the learned torque vs. turn radius relationship and slip model into an equivalent terramechanic-based model. This module only runs when it is expected that a terramechanic-based model will offer good predictions on the surface that has been recently navigated by the robot. For example, the module is run when there is a priori knowledge indicating that the robot is navigating on a hard surface where the wheel-terrain interaction can be characterized via the coefficient of friction and the shear deformation modulus. This mapping has the benefit that the terramechanic-based model can generalize to changes in vehicle payload. If there is not a priori knowledge of the terrain being navigated, the offline optimization is not run and the robot relies on the slip model and the torque vs. turn radius relationship learned by the neural network.

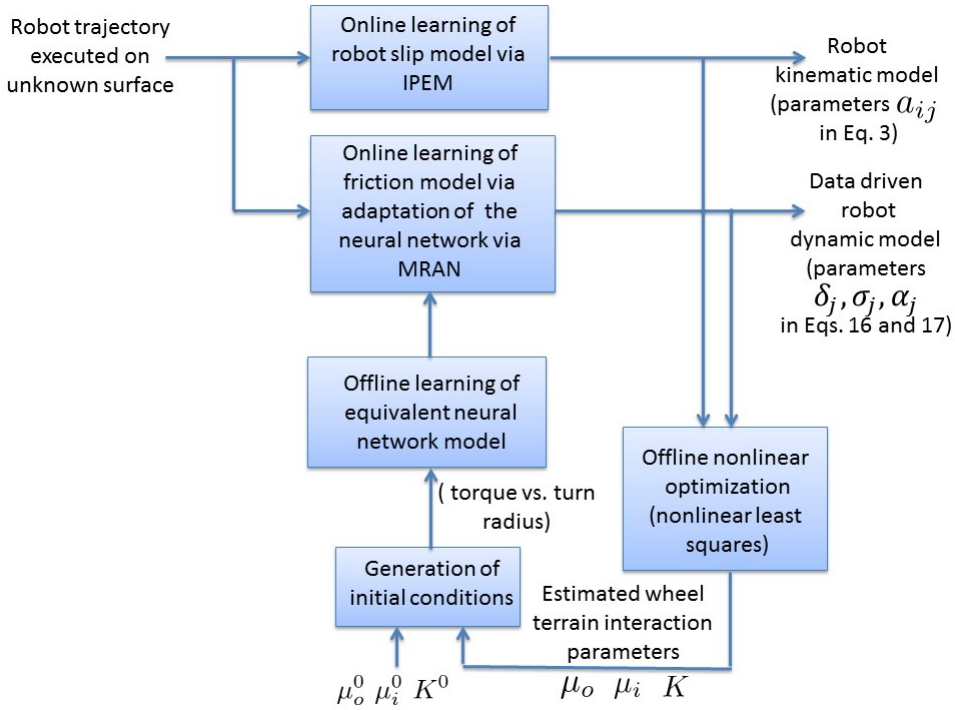


Figure 8: Schematic of the proposed learning approach. The methodology adapts slip and friction models in an online fashion. These models are then employed in motion planning applications.

3.1. Calibration of the Wheel-terrain Interaction Frictional Term

The terrain dependent term $C(\dot{q}_1)$ in (4) is here represented via two Radial Basis Function (RBF) neural networks (one network per robot side). This type of network was selected motivated by [18, 19, 20], which showed that RBFs can accurately and efficiently represent highly nonlinear processes. In this work we employ the Minimal Resource Allocation Network (MRAN) algorithm to online adapt the two neural networks.

The MRAN algorithm proposed by [21] has two main advantages over other neural network training algorithms. First, it allows the network to automatically change its size (increase and decrease) to accommodate changes in the system being learned. In addition, the utilization of an Extended Kalman Filter (EKF) enables the network adaptation to be conducted in an online manner. Also, as discussed in [18], MRAN results in faster convergence rates when compared to back propagation. As shown in Section 4,

the two neural networks capture the terrain dependent relationship between torque and turn radius.

A Radial Basis Function Network represents a function $f(x)$ as the result of some nonlinear mapping of the input state vector x. In general, x is the vector of the past n outputs and m inputs given by

$$x = \left[y_{i-1}^T, y_{i-2}^T, ..., y_{i-n}^T, u_{i-1}^T, u_{i-2}^T, ..., u_{i-m}^T\right]^T, \tag{15}$$

and $f(x)$ is the current output prediction given by

$$f(x) = \hat{y} = \alpha_0 + \sum_{j=1}^{N} \alpha_j \phi_j(x), \tag{16}$$

where each function ϕ_j is radially symmetric about some unique centroid δ_j. The RBF is a single layer neural network structure as depicted in Fig. 9. In MRAN, each ϕ_j is a Gaussian function,

$$\phi_j = \exp\left(-\frac{\|x_k - \delta_j\|^2}{\sigma_j^2}\right), \tag{17}$$

with unique unique widths σ_j. As expressed by (16), the output of the RBF network is a linear combination of multiple Gaussian functions. Each set of parameters $\delta_j, \sigma_j, \alpha_j$ constitutes a hidden unit.

In this research, the inputs to the neural networks are chosen to be the commanded turn radius ($x = \rho$) and the outputs are the predicted wheel torques. One could also use as inputs the robot wheel velocities and accelerations. However, it was found that for the energy efficient planning applications considered in Section 4.3 this is not necessary because this application promotes smooth robot motion (i.e., motion with limited wheel accelerations). In addition, as shown in Section 2.3, wheel torques do not vary significantly with speed for the range of velocities being considered in this work.

As illustrated in Fig. 10, the MRAN algorithm begins with an empty network ($N = 0$) and uses prediction error criteria at each time step k to determine whether an additional hidden unit is necessary to represent the system. The three error criteria are

$$e_1 = \|y_k - \hat{y_k}\|, \tag{18}$$

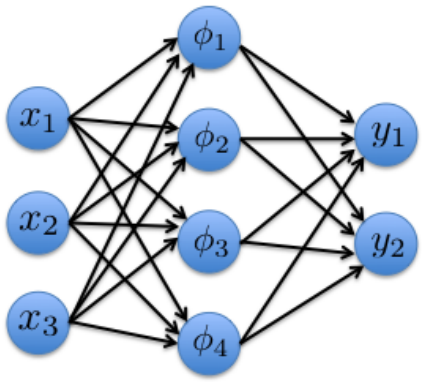


Figure 9: The Structure of a Radial Basis Neural Network with four hidden units. Each hidden unit ϕ_j evaluates a Gaussian function centered at δ_j that has dimension equal to the input vector x. Each output y_i is an affine function of the ϕ_j. For the approximation of the frictional model developed in this work, the input to the network is given by the commanded vehicle turn radius ($x = \rho$), and the outputs are the inner (τ_i) and outer (τ_o) torques respectively.

$$e_2 = \sqrt{\sum_{i=k-M+1}^{k} \frac{\|y_i - \hat{y}_i\|^2}{M}}, \tag{19}$$

$$e_3 = \|x_k - \delta^*\|, \tag{20}$$

where e_1 represents the instantaneous error between the predicted and measured outputs, e_2 is an RMS error over a window of size M, and e_3 is the Euclidean distance from the current state to the nearest basis vector (δ^*). A new hidden unit is added only when

$$e_1 > E_1, \ e_2 > E_2, \ \text{and} \ e_3 > E_3, \tag{21}$$

where E_1, E_2, and E_3 are tuning thresholds. When (21) is satisfied, a new unit is added with parameters $(\alpha_{N+1}, \delta_{N+1}, \sigma_{N+1})$ chosen to cancel out the instantaneous error. In particular,

$$\alpha_{N+1} = e_k, \delta_{N+1} = x_k, \sigma_{N+1} = \kappa\|x_k - \delta^*\|, \tag{22}$$

where κ is a basis function overlap parameter. If a hidden unit is not added

during a time step, the vector of model parameters

$$w = \left[\alpha_o^T, \alpha_1^T, \delta_1^T, \sigma_1, ..., \alpha_N^T, \delta_N^T, \sigma_N\right]^T \tag{23}$$

is updated using the Extended Kalman Filter (EKF) procedure detailed in [21].

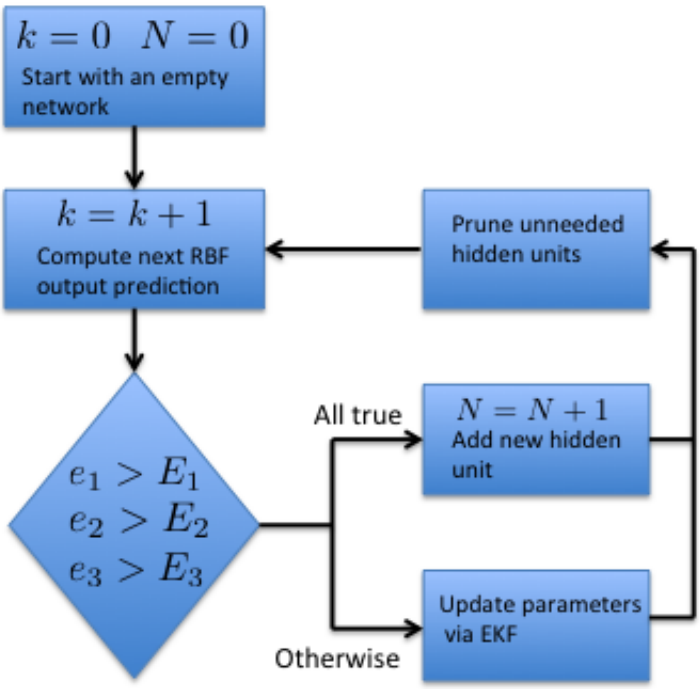


Figure 10: High level flowchart of the MRAN algorithm. The algorithm learns the suitable number of hidden units required to represent the system and sequentially adapts the parameters of those hidden units.

If large attitude motion is to be expected, the approach here proposed can be extended by including the forward and lateral components of the gravity vector. These implies that the slip formulation would have terms depending on the forward and lateral components of gravity. These two components would also be used as extra inputs to MRAN. The overall online learning approach would stay the same.

3.2. Mapping from Data Driven Model and Terramechanic-based Model

As mentioned before, in situations in which there exists a terramechanic-based model that characterizes the wheel-terrain interaction on the surface

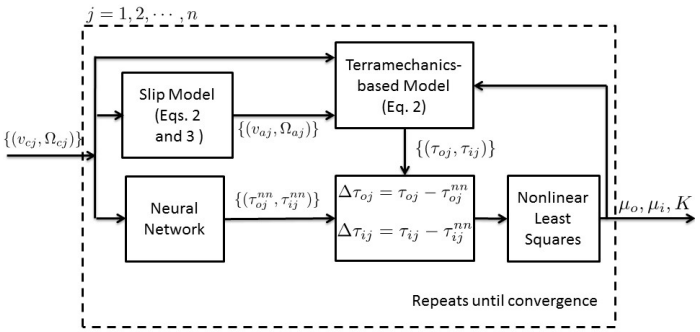


Figure 11: Offline optimization to map the torque vs. turn radius relationship learned via the neural network to an equivalent terramechanic-based model. In this figure v and Ω are the robot forward and angular velocities, subscript o refers to outer robot side, i to inner side, c to command, and a to actual. Superscript nn represents the neural network, τ torque, and index $j = 1, 2, \cdots n$ corresponds to the number of robot velocities samples used in the optimization process.

being navigated, it is advisable to perform a mapping from the learned torque vs. turn radius relationship (neural network model) and an equivalent terramechanic-based model. The primary advantage of this mapping consists in the generalization power of the terramechanic-based model to changes in vehicle payload.

As detailed in Fig. 11 the mapping is performed by using vehicle commands v_c, Ω_c. These velocity commands are passed through the slip model to estimate the actual robot velocities. Then, given a set of assumed surface parameters $[\mu_o, \mu_i, K]$, torque predictions for the outer and inner sides $[\tau_o, \tau_i]$ are generated with the terramechanics-based model. In parallel, the torque vs. turn radius relationship learned via the neural network is employed to generate its own torque predictions $[\tau_o^{nn}, \tau_i^{nn}]$. The predictions from these two models are then used to find the surface parameters that minimize the sum of the squared errors over N predictions. The optimization can be expressed as follows:

$$\min_{\mu_o, \mu_i, K} \sum_{j=1}^{N} [(\Delta^{(j)}\tau_o)^2 + (\Delta^{(j)}\tau_i)^2], \tag{24}$$

where $\Delta\tau_o = \tau_o - \tau_o^{nn}$, $\Delta\tau_i = \tau_i - \tau_i^{nn}$, and N represents the number of predictions. Since in this work, we focus on constant linear speed motion, a vehicle command set with $N = 8$ elements was chosen as follows: $v_c = 0.6m/s$ and $\Omega_c = \{0.24, 0.17, 0.08, 0.05, 0.01, 0.004, 0.002, 0.001\}rad/s$. The optimization is performed using the Matlab routine *fminsearch*.

3.3. Computational Characteristics of the Algorithms

In this section we discuss the memory requirements, complexity, and optimality of the different algorithms that make up the online learning approach.

3.3.1. MRAN

MRANs complexity is proportional to the number of parameters being updated z. The algorithm is $O(z^2)$ where $z = 2 + 4h$, with h being the number of hidden neurons. In MRAN h changes dynamically. It increases when new neurons are added and decreases whenever pruning takes place. In our application, we have specified a maximum value of $h = 30$.

Since the number of parameters being updated changes, the memory requirements are variable. A total of $z = 2 + 4h$ parameters will need to be stored in memory. In the worst case $h = 30$ and $z = 122$.

Due to the linearization employed by the EKF within MRAN, optimality guarantees are not available. In terms of convergence, we have found empirically that under the assumption of relatively homogeneous terrains, the algorithm is able to learn a model that accurately captures the torque vs turn radius curves. Convergence is however dependent on the type of diagnostic trajectories, which is why this work employs a rich diagnostic trajectory that excites the system dynamics.

3.3.2. IPEM

IPEM's complexity is $O(m^2n^2)$,where m is the size of the measurements and n is the size of the state being estimated. In our application $m = n = 3$.

The IPEM implementation used here is very memory efficient as it only requires storage of 6 matrices employed by the EKF. The largest of which is 9×9.

Similiarly as MRAN, the linearization used by the EKF within IPEM implies that optimality guarantees are not available. In terms of convergence, [13] has shown empirically convergence times of 1.5 minutes when starting from a fully uncalibrated model (all slip parameters set to zero). Notice that in practice this time can be significantly reduced when the parameters are initialized using values from a previous calibration experiment.

3.3.3. SBMPO

As with most graph search algorithms [22], the computational complexity of SBMPO scales with the number of expanded nodes. In the worst case, the SBMPO graph grows exponentially, i.e. $(O(B)^N)$, where the B is the branchout factor, and N is the prediction horizon. In practice, however, the use of a suitable heuristic allows the average complexity to tend towards the best case, which has linear execution time $O(N)$. This tendency holds because while searching for an optimal trajectory, SBMPO expands nodes in a best-first sense, selecting at each iteration the most promising node according to a heuristic, which approximates the cost of pursuing a trajectory through that node.

The SBMPO search requires memory in proportion with the number of expanded nodes; therefore the memory used will scale proportionally with the computational cost above.

As an A^*-type algorithm, the optimality of the result is guaranteed as long as the heuristic function is optimistic (i.e. never exceeds the true minimum cost to reach the goal state). In this research the heuristic based on straight-line energy cost is assumed to be optimistic, yielding optimal SBMPO searches.

The implementation of SBMPO used in this research is not guaranteed to converge, however SBMPO may be formulated as an incremental search algorithm, which guarantees an initial suboptimal result, and is refined and further optimized in subsequent epochs. An example of this incremental implementation is the Anytime A^* implementation given in [23]

4. Experimental Results

In the following experiments the robot was commanded with a trajectory that starts with straight motion and progressively decreases the turn radius in a spiral-type manner. The diagnostic trajectory comprises robot commands with varying linear speeds in the set $v_c = \{0.2, 0.4, 0.6\}m/s$ and alternating CCW and CW turn radii in the set $\rho = \{749.89, -421.69, 237.13, -133.35, 79.43, -50.12, 31.62, -19.95, 12.58, -9.33, 8.12, -7.07, 6.15, -5.35, 4.45, -3.54, 2.81, -2.23, 1.77, -1.41, 1.11, -0.88, 0.70, -0.56\}m$. Each robot command was maintained constant for a duration of $5s$. The objective of selecting such a trajectory is to obtain a rich set of operating conditions to excite the system dynamics and properly characterize the torque vs. turn radius relationship and at the same time allow for calibration of the vehicle kinematics.

Table 2: Kinematic model slip parameters [$\times 10^{-3}$]

Surface	a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	a_{31}	a_{32}	a_{33}
Asphalt	15	-0.4	-10	7	15	-7	5	-405	119
Concrete	9	12	-28	6	-25	68	0.3	-394	88
Grass	15	6.9	-19	9	9.4	14	0.01	-337	33

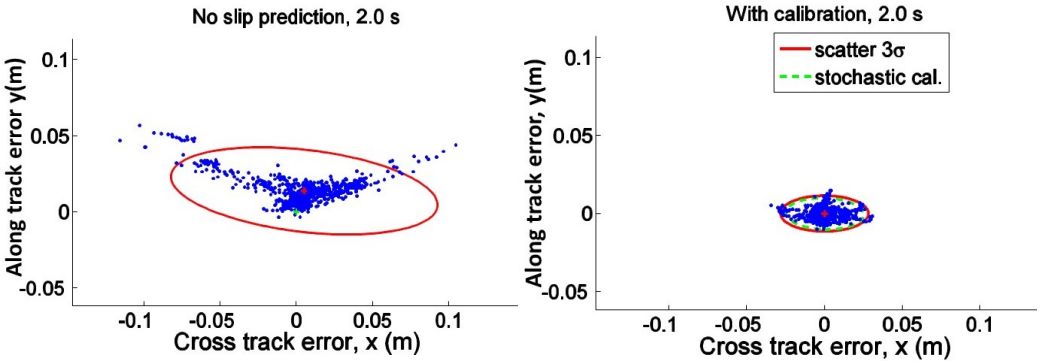
In all experiments, wheel torques and wheel odometry were sampled at a rate of $1KHz$. Before feeding the torques to the learning module, they are compensated for the gravity component using (14). Pose data required by the IPEM algorithm was obtained using visual odometry running at a rate of $10Hz$.

4.1. Offline Initialization Process

A terramechanic-based and slip model are used here to generate sound initial conditions for the neural network. The training process utilizes the dynamic model (4) to generate 30 triplets of outer and inner wheel torques and commanded turn radius (τ_o, τ_i, ρ). The turn radii are selected such that sharp, medium, and shallow turns are included. In order to generate the torque predictions (τ_o, τ_i) corresponding to each of the selected turn radii, the term $M\ddot{q}_1$ in (4) was ignored. This approximation is performed because as discussed previously, energy efficient planning promotes motion with limited wheel accelerations. This data was then presented sequentially to the MRAN algorithm. In this work, model parameters from a lab's vinyl surface were employed. However, we expect that in the field, the robot will perform the initialization based on previously characterized terrains that appear perceptually similar to the terrain being traversed.

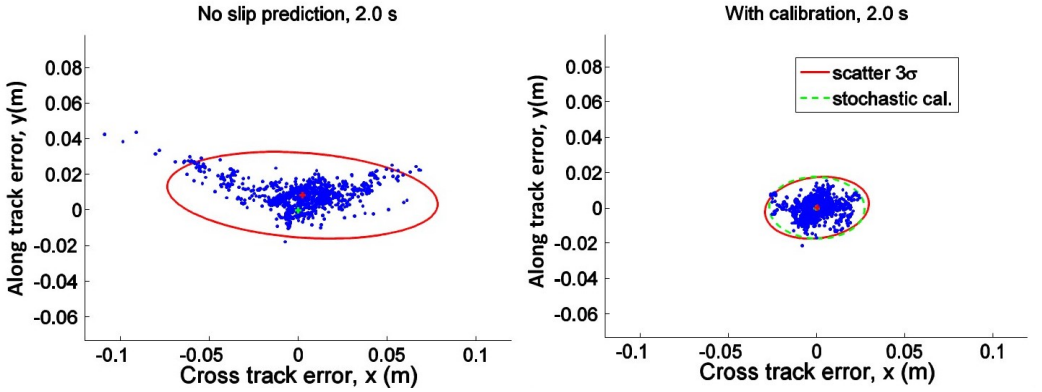
4.2. Online learning

The same diagnostic trajectory as the one described above was commanded on asphalt, concrete, and grass. The calibrated slip models for these surfaces are summarized in Table 2, where all the coefficients for the models are provided. In addition, Figs. 12 to 14 compare along track and cross track errors corresponding to the trajectories for both uncalibrated and calibrated cases. These errors are computed as the difference between the pose predicted by the kinematic model and the pose measurements obtained via



(a) without calibration. (b) with calibration

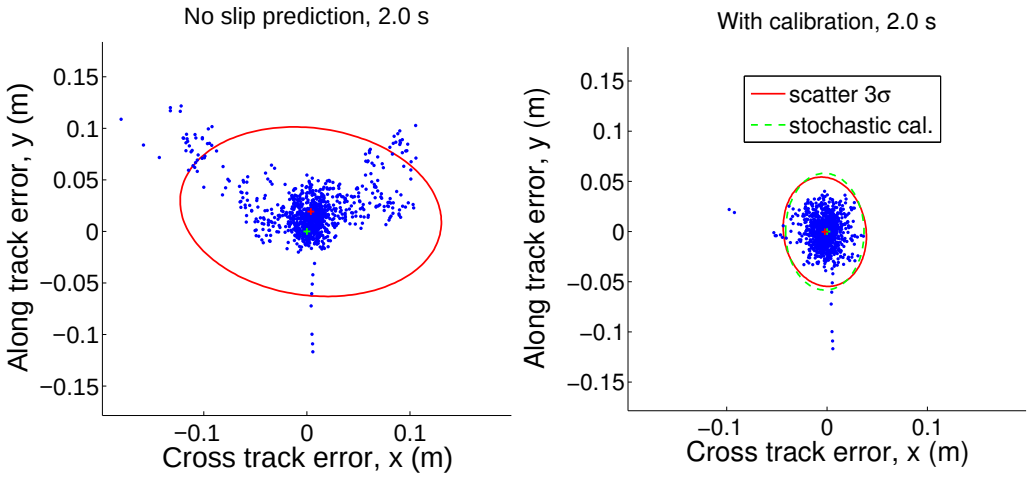
Figure 12: Calibration of slip model on an asphalt surface.



(a) without calibration. (b) with calibration.

Figure 13: Calibration of slip model on a concrete surface.

visual odometry over a prediction horizon of $2s$. Each point in Figs. 12 to 14 corresponds to the cross track and along track errors over a $2s$ prediction horizon. Due to the non-uniformity in length and density of grass, it is expect that as shown in Fig. 14, the variability of the errors is larger for grass than for the other two terrains. The figures also show scatter three sigma ellipses of error points as a solid red line. The stochastic model calibrated by IPEM also predicts a unique pose error covariance for each path segment. The dashed green ellipses in Figs. 12 to 14 correspond to the average of these predictions. Notice the good correspondance with the three sigma ellipses in the calibrated cases.



(a) without calibration.

(b) with calibration.

Figure 14: Calibration of slip model on a grass surface.

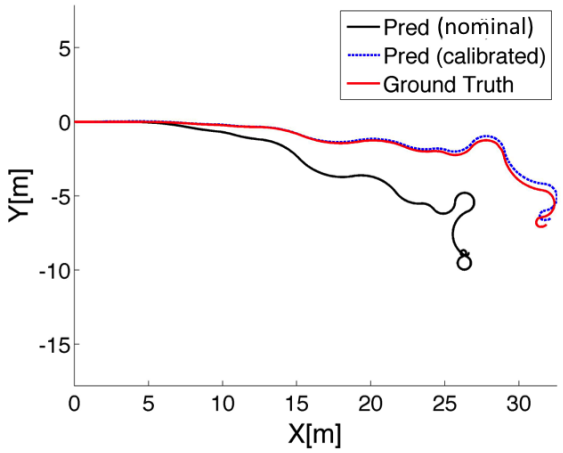


Figure 15: Comparison of pose predictions generated by the nominal (differential drive, no slip) kinematic model and the slip calibrated model. The run was performed on an asphalt surface and the ground truth was obtained using visual odometry.

To better illustrate the effect of the calibrated slip model, consider Fig. 15 which compares predicted trajectories assuming the nominal (no slip) vehicle kinematic model and the slip calibrated kinematic model. Notice the good correspondence between the calibrated model predictions and the ground

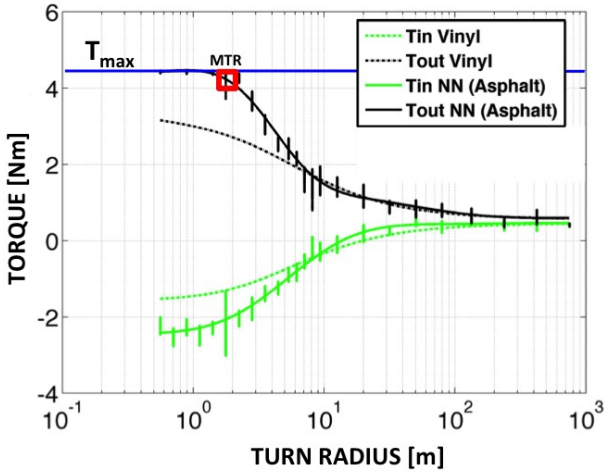


Figure 16: Online learning of the frictional term on asphalt. The initial conditions where generated using the terramechanic model corresponding to a vinyl surface. The vertical lines correspond to measured torques. Notice that the learned model predicts a MTR of 1.8m whereas the vinyl model does not have a MTR restriction.

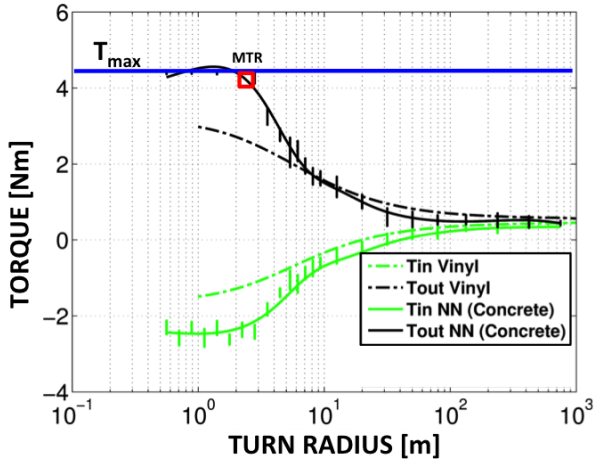


Figure 17: Online learning of the frictional term on concrete. The initial conditions where generated using the terramechanic model corresponding to a vinyl surface. The vertical lines correspond to measured torques. Notice that the learned model predicts a MTR of 2.4m whereas the vinyl model does not have a minimum turn radius restriction.

truth collected via visual odometry.

The resultant calibrated frictional terms are summarized in Figs. 16 to

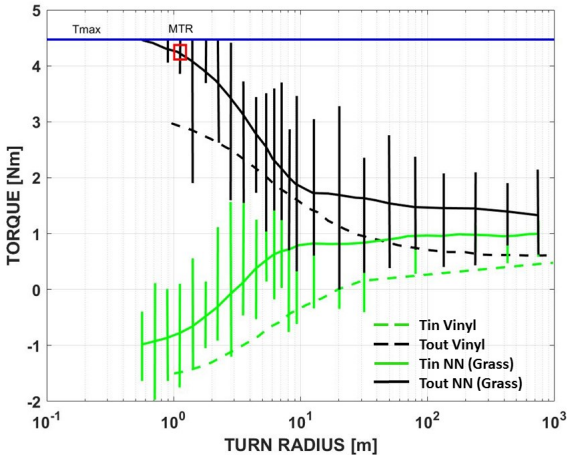


Figure 18: Online learning of the frictional term on grass. The initial conditions where generated using the terramechanic model corresponding to a vinyl surface. The vertical lines correspond to measured torques. Notice that the learned model predicts a MTR of 1.2m whereas the vinyl model does not have a MTR restriction.

18. Notice that despite the high variability of the measured torques, the algorithm is able to approximate the average behavior of the torques as a function of turn radius. The learned torque vs. turn radius relationships are also employed to estimate the minimum turn radius constraints. As shown in [2], if MTR constraints are violated, the robot is not able to track the desired commands and behaves erratically.

When the wheels of the outer side to the turn reach torque saturation, the robot lacks control authority and is not able to track the commanded turn radius. In Figs. 16 to 18, raw data of wheel torques is compared to the predicted wheel torques from the nerual network. From the experimental torque data it is possible to determine the actual minimum turn radius on the respective surfaces. It corresponds to the largest turn radius that results in a outer torque that violates the maximum torque line. As shown in Figs. 16 to 18, the neural network predicts torques close to the average torque measurements. Figure 16 shows that for turn radius of less than $1.77m$, the raw torques for the outer wheels are in saturation. Notice the low variability of the measured torques for those turn radii. To be conservative, the neural network returns an MTR selected as the largest turn radius that would result in an outer torque requirement of more than 95% of the maximum available motor torque, which in this case was $1.8m$. A similar analysis can be followed

Table 3: Minimum Turn Radius (MTR) Constraints

Surface	Asphalt	Concrete	Grass
MTR[m]	1.8	2.4	1.2

in Fig. 17, where the outer side motor was saturated for turn radii of less than $2.23m$ and the neural network predicts $2.4m$ as the MTR. Figure 18, which corresponds to grass, shows a much higher variation of measured torques for a given turn radius. This is due to the non-homogeneity of grass. It can be seen that the outer wheel reached saturation even at a turn radius of $2.23m$. However, this was very momentarily and did not affect the turning maneuver of the robot. Turning was affected for turn radii of less than $1.11m$; in this scenario, the neural network predicted an MTR of $1.8m$.

It is key to realize that the MTR constraints are dependent on many factors such as the terrain, the payload, the robot wheels, the pressure in the wheels, and the motors.

The importance of calibrating the the robot dynamic model is illustrated in Fig. 19, which presents a comparison of energy prediction using the nominal dynamic model corresponding to vinyl and the calibrated model on the asphalt surface. The actual energy consumption is also provided for reference. This experiment corresponds to the same vehicle trajectory of Fig. 15.

4.3. Applications of the Learned Models

To illustrate the application and importance of the learning and utilization of the discussed kinematic and dynamic models in motion planning tasks, here we consider energy efficient motion planning under different scenarios. As a motion planning tool, Sampling Based Predictive Optimization (SBMPO) is utilized [24]. This planner is selected as it can efficiently work with kinematic and dynamic models while finding a trajectory from the start to the goal state. Along the planning process, a graph is built through sampling of the control inputs and forward simulation of the system model. Trajectories are generated such that a given cost function is optimized (e.g., energy or time).

The interaction of the planner with the terrain dependent robot models discussed in this paper is depicted in Fig. 2. Notice that during plan-

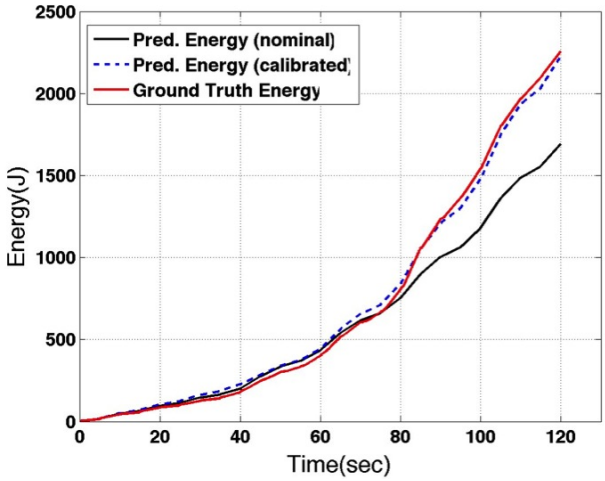


Figure 19: Comparison of energy predictions generated by the nominal (vinyl) and the calibrated dynamic model on asphalt. The run was performed on an asphalt surface.

ning, SBMPO keeps track of the predicted robot position by utilizing the slip-calibrated kinematic model. Similarly, the energy predictions and dynamic constraints (i.e., MTR) required by the planner are generated using the learned frictional models. In this application, the minimum energy heuristic required by the planner is computed as the energy required to reach the goal from the current robot location, assuming straight-line motion.

Provided the learned torque (τ) vs. turn radius (ρ) relationship, energy consumption is estimated through integration of the power consumption $E = \int_{t_0}^{t_f} P dt$ of both the outer and inner robot sides. The power consumption is computed as follows

$$P = \dot{q_1}\tau + R_e i^2, \tag{25}$$

where the motor torques τ are given by (4), q_1 represents the robot wheel velocities, R_e the electrical resistance of the motors and i the motor currents, which is computed as

$$i = \frac{\tau}{K_T g_m \eta_m}, \tag{26}$$

where K_T is the torque constant, g_m is the gear ratio, and η_m is the mechanical efficiency of the motor. For further details on power modeling and tuning of SBMPO, refer to [17]. It is important to clarify that for the sake of computational efficiency the torque vs. turn radius predictions generated by the dynamic model are stored and used within the motion planner in the

form of look up tables.

In all the following scenarios, the robot linear velocity was maintained fixed at $0.6m/s$ and angular velocity was the sampled control input. The MTR constraints are used within SBMPO to limit the range of feasible turn radii. In the current implementation, SBMPO employs 20 samples (obtained via Halton sampling) of the robot angular velocity. The initial robot orientation was set to $0°$ with respect to the horizontal axis. In addition, cones were placed as obstacles and visual odometry was employed to generate ground truth data.

As mentioned above, the modeling approach here presented was conceived to aid in applications such as energy efficient planning where low accelerations are expected. Previous work in the literature for larger (tank size) skid-steered platforms [9] has shown that for significant ranges of speeds sprocket torques do not vary considerably as a function of speed. Therefore, the approach is not expected to deteriorate significantly as a function of vehicle speed.

The first scenario, conducted on an asphalt surface, illustrates the importance of using the calibrated kinematic model. Figure 20 shows the planned and executed energy efficient trajectories when the planner uses the nominal (i.e., a differential) kinematic model. It is clear that the robot diverged significantly from the planned path. On the contrary, Fig. 21 shows the same scenario but with the slip-calibrated robot model corresponding to the actual asphalt surface. Notice the good correspondence between the planned and executed robot paths. In both cases, the identified power model corresponding to the asphalt surface was employed.

The second scenario considers the following two situations. In Fig. 22 the robot utilizes the slip model corresponding to an asphalt surface but the actual surface is concrete. Notice the wider turn executed by the robot due to the usage of the erroneous slip model. On the contrary, Fig. 23 shows results when the robot utilizes the proper slip model for the concrete surface. In both cases, the robot employed the identified concrete power model.

The third scenario shows the benefit of employing the terramechanic model to provide generalization to payload changes. In this case the neural network model was adapted online for asphalt with the platform having a payload of $0kg$. The calibrated models were then used to obtain an equivalent terramechanics-based model with the procedure detailed in Section 3.2. The optimization converged within $0.59s$ and yielded the optimized surface parameters $[\mu_{out}\ \mu_{in}\ K] = [1.209\ 0.795\ 0.0016m]$. Once these parameters

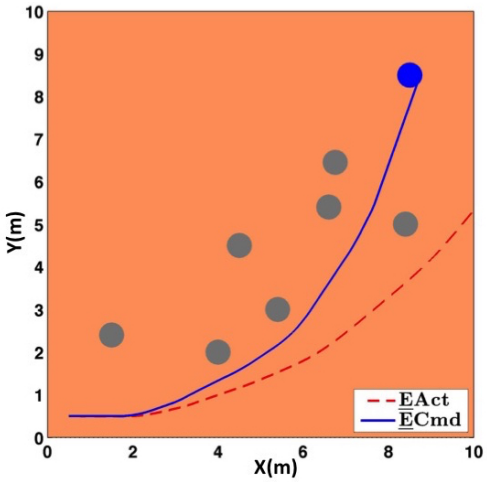


Figure 20: Experimental Scenario 1: robot on asphalt using the nominal (differential drive) kinematic model. Notice the large discrepancy between the planned and executed paths. In this experiment, the robot utilizes the identified asphalt power model.

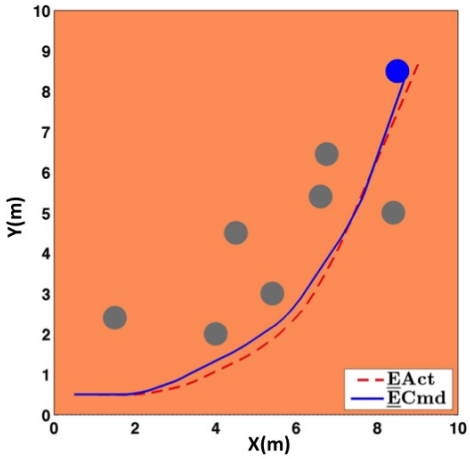


Figure 21: Experimental Scenario 1: robot on asphalt using the slip calibrated kinematic model. Notice that the robot closely follows the planned trajectory. The predicted energy was $456J$ and the actual energy was $417J$ (9.3% error). In this experiment, the robot utilizes the identified asphalt power model.

are obtained, it is possible to generate torque vs. turn radius predictions for different payloads without the need of additional experiments.

Figure 24, shows an experimental result where the robot utilizes the esti-

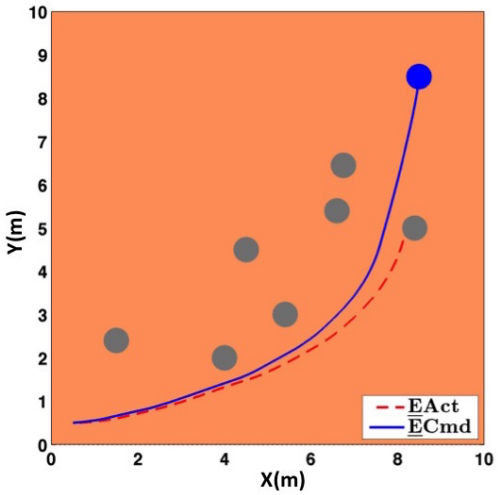


Figure 22: Experimental Scenario 2: robot on concrete but planning assuming slip model for asphalt. Notice the collision with the obstacle.

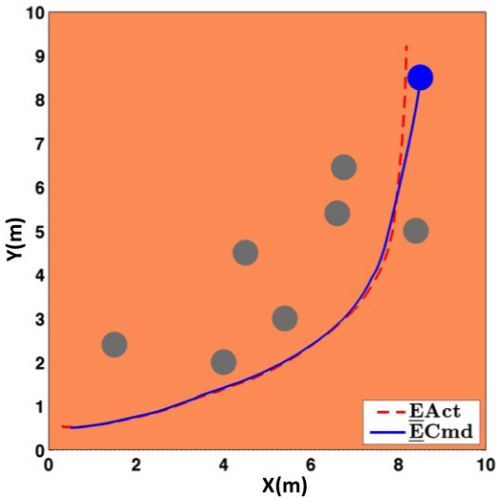


Figure 23: Experimental Scenario 2: robot on concrete and planning using the adapted slip model for concrete. Notice the good correspondence between the planned and executed trajectories. The predicted energy was $622J$ and the actual energy was $521J$.

mated surface parameters corresponding to $0kg$ payload but plans and executes an energy efficient trajectory for an $8kg$ payload. Thanks to the generalization of the terramechanic model, accurate torque and energy predictions were generated for the new payload, yielding an energy prediction error of

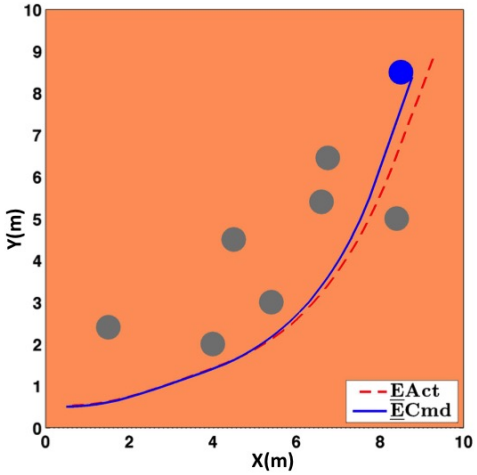


Figure 24: Experimental Scenario 3: Energy efficient motion planning with calibrated kinematic and dynamic models on an asphalt surface. In this scenario the neural network learned the dynamic model for a $0kg$ payload. The learned model was then mapped back to an equivalent terramechanic model, which is able to generalize to different payloads. In this scenario the robot is actually carrying an $8kg$ payload. The predicted energy was $625J$ and the actual energy was $545J$.

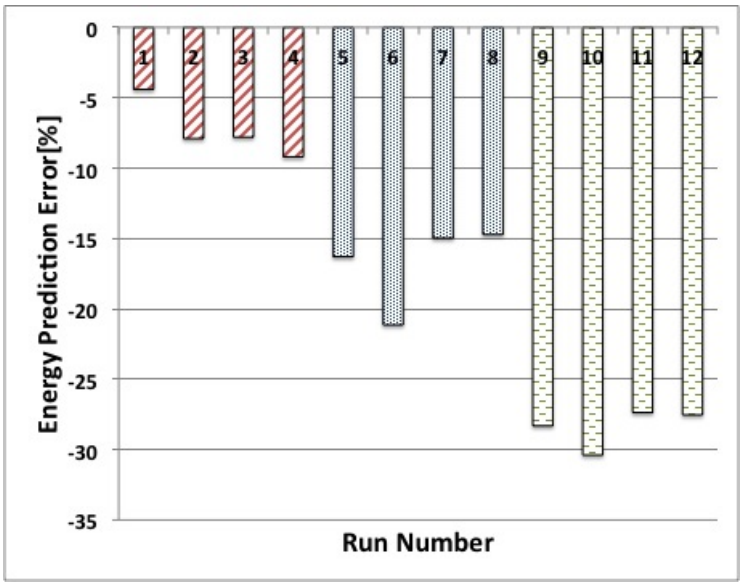


Figure 25: Energy prediction errors on asphalt (runs 1-4), concrete (runs 5-8), and grass (runs 9-12).

12.8%. It is relevant to note that with the learning module turned on the average energy prediction error was -6.3% on asphalt and -16.8% on concrete. However, if the adaptation is turned off, the average energy prediction error was 23% on asphalt and 42.2% on concrete.

Finally, Fig. 25 shows typical energy prediction errors for various planning scenarios on asphalt, concrete, and grass surfaces. The negative signs refer to over-prediction of the model, which can serve as a safety factor and is always preferred over under-prediction. The over-prediction of energy consumption is partly due to the fact that the predicted torques tend to be greater than the experimental torques, leading to an over-prediction of power. Integration of the small error in power over time leads to increased errors in the prediction of energy. Although not observed in these experiments, It is worth mentioning that there could be some trajectories or trajectory regions that lead to underestimation of energy consumption. As envisioned from Fig. 18, the energy prediction errors are larger for the grass surface due to the large torque variability. The computational performance of the algorithms were of $240ms$ for SBMPO, 9 $ms/iteration$ for IPEM and 0.05 $ms/iteration$ for MRAN. The performance time for SBMPO corresponds to the average time required to plan an energy efficient trajectory on diverse scenarios in a $10m \times 10m$ area and assuming a robot linear speed of $0.6m/s$. Notice that this time is low enough to perform replanning whenever the environment or the vehicle models change significantly.

5. Conclusions and Future Work

This paper developed a new methodology to perform online learning of terrain dependent robot models that are highly relevant for motion planning applications. The proposed methodology combines features of both terramechanics and data driven approaches. The terramechanic formulation is used here as a means to generate sound initial conditions for a neural network, which is in turn capable of performing fast online learning of critical terrain dependent terms of the robot dynamic model.

The selected neural network provides generality to the proposed learning methodology as it is capable of capturing wheel-terrain interactions from different terrains without having to rely on very specific model formulations. The paper also shows experimentally that when terramechanic-based models are available, it is possible to find equivalences between them and the neural network representations. This mapping between the different models is shown

to be effective to achieve generalization to changes in vehicle payload, without the need for extensive experimentation.

This paper also performed integration of calibrated dynamic and slip models with a motion planner. In particular, it experimentally showed the necessity of combining these two models in energy efficient motion planning tasks. Dynamic models that predict torque requirements as a function of vehicle commands are important as they lead to estimates of energy consumption and provide minimum turn radius constraints. Similarly, slip models are also necessary to accurately predict robot motion within the planning cycle. When slip models are ignored, the executed robot trajectories differ abruptly from the planned ones, and in many situations it leads to collisions with obstacles.

Future work will include the inclusion of near to far learning as a more intelligent way to initialize robot models. In other words, models will be initialized through learning of correlations of perceptual features of the terrain ahead of the robot and terrains previously navigated by the robot. In addition, environments where the robot navigates in mixed terrain surfaces will be considered. In very heterogeneous terrains, it would be required to rely on a perception system capable of automatically separate the data of the different terrains being traversed. The separated data would then be used to adapt vehicle models to each section of the different terrains.

Finally, the research here developed will be transitioned to unique mobility platforms such as the XRL hexapedal robot [25]. By doing this transition, it will be possible to explore more challenging terrains.

Acknowledgments

This work was supported by the collaborative participation in the Robotics Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD 19-01-2-0012. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes not withstanding any copyright notation thereon.

References

[1] W. Yu, O. Chuy, E. Collins, P. Hollis, Analysis and experimental verification for dynamic modeling of a skid-steered wheeled vehicle, IEEE Transactions on Robotics 26 (2) (2010) 1–19.

[2] C. Ordonez, N. Gupta, W. Yu, O. Chuy, E. Collins, Modeling of skid-steered wheeled robotic vehicles on sloped terrains, in: Proceedings of the ASME Dynamic Systems and Control Conference, Fort Lauderdale, FL, 2012.

[3] A. Mandow, J. Martinez, J. Morales, J. Blanco, A. Garcia-Cerezo, J. Gonzales, Experimental kinematics for wheeled skid-steer mobile robots, in: IEEE International Conference on Intelligent Robots and Systems, San Diego, CA, 2007, pp. 1222–1227.

[4] L. Martinez, A. Mandow, J. Morales, S. Pedraza, A. Garcia-Cerezo, Approximating kinematics for tracked mobile robots, International Journal of Robotics Research 24 (2005) 867–878.

[5] Z. Song, Y. Zweiri, L. Seneviratne, Nonlinear observer for slip estimation of skid-steering vehicles, in: IEEE International Conference on Robotics and Automation, Orlando, FL, 2006, pp. 1499–1504.

[6] L. Caracciolo, A. Luca, S. Iannitti, Trajectory tracking control of a four-wheel differentially driven mobile robot, in: Proceedings of the IEEE International Conference on Robotics and Automation, Detroit, MI, 1999, pp. 2632–2638.

[7] K. Kozlowski, D. Pazderski, Modelingandcontrolofa4-wheelskidsteering mobile robot, International Journal of Applied Math and Computer Science 14 (2004) 477–496.

[8] W. Ehlert, B. Hug, I. Schmid, Field measurements and analytical models as a basis of test stand simulation of the turning resistance of tracked vehicles, Journal of Terramechanics 29 (1) (1992) 57–69.

[9] J. Wong, C. Chiang, A general theory for skid steering of tracked vehicles on firm ground, Proceedings of the Institution of Mechanical Engineers, Part D, Journal of Automotive Engineerings (2001) 343–355.

[10] S. Hutangkabodee, Y. Zweiri, L. Seneviratne, K. Altho, Multi-solution problem for track-terrain interaction dynamics and lumped soil parameter identification, in: P. Corke, S. Sukkariah (Eds.), Field and Service Robotics, Vol. 25 of Springer Tracts in Advanced Robotics, Springer Berlin Heidelberg, 2006, pp. 517–528.

[11] K. Iagnemma, S. Kang, H. Shibly, S. Dubowsky, Online terrain parameter estimation for wheeled mobile robots with application to planetary rovers, IEEE Transactions on Robotics 20 (5) (2004) 921–927.

[12] L. Ray, D. Brande, J. Lever., Estimation of net traction for differential-steered wheeled robots, Journal of Terramechanics 46 (3) (2009) 75–87.

[13] N. Seegmiller, F. Rogers-Marcovitz, G. Miller, A. Kelly, Vehicle model identification by integrated prediction error minimization, The International Journal of Robotics Research 32 (8) (2013) 912–931.

[14] J. Yi, D. Song, J. Zhang, Z. Goodwin, Adaptive trajectory tracking control of skid-steered mobile robots, in: Proceedings of the IEEE International Conference on Robotics and Automation, Roma, Italy, 2007.

[15] J. Yi, H. Wang, J. Zhang, D. Song, S. Jayasuriya, J. Liu, Kinematic modeling and analysis of skid-steered mobile robots with applications to low-cost inertial measurement unit based motion estimation, IEEE Transactions on Robotics 25 (5) (2009) 1087–1097.

[16] R. Siegwart, I. Nourbakhsh, Introduction to Autonomous Mobile Robots, MIT Press, 2004.

[17] N. Gupta, Dynamic modeling and motion planning for robotic skid-steered vehicles, Ph.D. thesis, The Florida State University, Tallahassee, FL (June 2014).

[18] B. Reese, E. Collins, Sampling based control of a combustion process using a neural network model, in: Proceedings of the IEEE Systems, Man, and Cybernetics, San Diego, CA, 2014.

[19] Y. Lu, N. Sundararajan, P. Saratchandran, A sequential learning scheme for function approximation using minimal radial basis function neural networks., Neural Computation 9 (2) (1997) 461–478.

[20] J. Platt, A resource-allocating network for function interpolation, Neural Computation 3 (2) (1991) 213–225.

[21] N. Sundararajan, P. Saratchandran, L. Wei, Radial Basis Function Neural Networks with Sequential Learning, World Scientific Publishing Co, 1999.

[22] D. Delling, P. Sanders, D. Schultes, D. Wagner, Engineering route planning algorithms. Algorithmics of large and complex networks, Springer, 2009.

[23] M. Likhachev, G. Gordon, S. Thrun, Ara*: Anytime a* with provable bounds on sub-optimality, in: Advances in Neural Information Processing Systems: NIPS Conference, MIT Press, 2004.

[24] D. D. Dunlap, C. V. Caldwell, J. E. G. Collins, O. Chuy, Motion planning for mobile robots via sampling-based model predictive optimization, in: Mobile Robots, Vol. 1, InTech, 2011.

[25] G. C. Haynes, J. Pusey, R. Knopf, A. M. Johnson, D. E. Koditschek, Laboratory on legs: an architecture for adjustable morphology with legged robots, in: SPIE Unmanned Systems Technology XIV, Baltimore, Maryland, USA, 2012.