Supervisory Traction Control for a Slipping UGV

John A Broderick, Dawn M Tilbury & Ella M Atkins

Abstract—Unmanned Ground Vehicles (UGVs) face many mobility challenges as they operate in unstructured environments. In addition to insurmountable obstacles, UGVs can lose traction and slip, reducing maneuverability and potentially causing collisions with obstacles. This paper develops a dynamic model of a differentially-driven ground robot with lateral and longitudinal slip, using a ground friction model from the literature. A novel switching control system, with velocity input commands, is also presented. The controller switches from PID control to a trajectory planning mode when the robot begins to slip or when the normal command would exceed the ground friction forces. The trajectory is planned and executed based on commanded velocities. Stability and robustness of the trajectory planning controller are discussed. This control law is compared in simulation to a PID controller, using the UGV model, on different surfaces. The traction controller drives the UGV closer to the desired path derived from the reference trajectories.

I. INTRODUCTION

Unmanned ground vehicles (UGVs) operate in many different, unstructured environments. They are widely used inside buildings, on hard road surfaces, as well as on softer off-road surfaces. Some environments are easy for the UGVs to traverse, asphalt roads for example, while some terrains pose significant hazards, such as frozen lakes, sand dunes, and slippery hillsides. In addition, UGVs can suffer from non-catastrophic failures that limit mobility and operation [1]. Since many UGV operations take place where humans cannot safely and easily reach, simple failures can become catastrophic without ways to address the failures.

This paper presents a controller to address mobility failures. In particular, we develop a dynamic model of a UGV with longitudinal and lateral wheel slip and present a switching controller to deal with excessive wheel slip. We focus exclusively on two-wheeled, differentially-steered UGVs. While we are interested in command tracking, the motivation for this research is to prevent catastrophic situations for UGVs. From that perspective, we view excessive slip as a “fault” that needs to be mitigated or prevented before the UGV can return to normal operation. Additionally, we can augment a traction controller with other controllers designed to deal with other mobility faults, e.g., wheel or track failure or immobilization of the UGV in a terrain feature.

We first present related work on wheel slip and control in Section II. The slipping UGV model is derived in Section III. Section IV presents the supervisory traction control algorithm, including the switching conditions, and the trajectory planning method. A stability and robustness analysis is presented. Simulation results are discussed in Section V and conclusions are presented in Section VI.

II. BACKGROUND

Wheel slip is a well-known topic that has been studied in both automotive and wheeled robotic settings. Tracked robots can also experience slip. There are many different ways of formulating a model of tire/ground interaction [2]. Since we are focused on a robotic system with longitudinal and lateral slip, we used the formulas presented in [3] to develop our friction model for slipping.

Wheel slip is a known issue for position tracking for wheeled mobile robots. In [4], the authors study a three-wheeled robot with omni directional wheels. They present a dynamic model of the system and compare tracking performance of a PID controller with a model-based controller, demonstrating the need for incorporating slip into control laws. Williams et al. analyze the friction model for this class of wheeled robots [5]. Wang and Low present a control design perspective on skidding and slipping for different configurations of wheeled robots [6]. Their analysis focuses on the controllability of the different configurations, but only analyses a kinematic model. We focus on a dynamic model for increased performance.

Sarkar and Yun present a traction control algorithm for a robot with two drivable and steerable wheels [7]. While they present strong results for this class of robots, we are motivated by a differentially-driven robot for our research. An adaptive nonlinear controller for tracked robots with longitudinal slip is presented in [8]. This model focuses on a kinematic model for tracked vehicles, as opposed to a dynamic model. These are all similar works with different UGV configurations. A differentially-driven robot with lateral slip only is studied in [9]. Our research broadens their work by including longitudinal slip.

Nandy et al. present a dynamic model for a differentially-driven mobile robot with slip [10]. We expand their model by using a different method of eliminating constraints [11] and a different friction model [3].
A. Dynamics

To derive the equations of motion, we use the Euler-Lagrange method. We model the UGV as three different rigid bodies: the platform and two wheels. Additionally we assume that the UGV is operating on flat ground. We can form the Lagrangian from the kinetic energy:

\[
L = \frac{1}{2}m_p \left( \dot{x}_c^2 + \dot{y}_c^2 + \Delta^2 \dot{\phi}^2 - 2\Delta \dot{x}_c \sin \phi - \dot{y}_c \cos \phi \right) + \frac{1}{2} (I_p + 2I_m) \ddot{\phi}^2 + \frac{1}{2}I_w \left( \dot{\theta}_1^2 + \dot{\theta}_2^2 \right) + \frac{1}{2}m_w \left( r_\theta \dot{\delta}_l - \dot{\delta}_l \right)^2 + 2\zeta^2. \tag{1}
\]

There are 6 different forces acting on the system: lateral and longitudinal friction at each wheel and motor torques acting on each wheel. Since the lateral slip is in the same direction for both wheels, the lateral friction forces add to one lateral force. We define the generalized forces as

\[
Q = [0, 0, 0, \tau_r, \tau_l, -H_r, -H_l, K_r + K_l]^T. \tag{2}
\]

Using the Euler-Lagrange method, we can write the system dynamics in the form \( M(q) \ddot{q} + C(q, \dot{q}) \ddot{q} + G(q, \dot{q}) = B(q)u \). We have included the friction terms in \( G(q, \dot{q}) \), even though the friction terms depend on the velocities, due to the non linear dependence on velocity. In addition, we must take into account the state constraints at the wheel contact points. Under the assumption of no slipping, the constraints relate the wheel velocities \( \theta_{r,l} \) to the system velocities \( (\dot{x}_c, \dot{y}_c, \dot{\phi}) \).

By introducing slip, we now formulate the constraints as:

\[
\begin{align*}
\dot{x}_c \cos \phi + \dot{y}_c \sin \phi + b \dot{\phi} &= r \dot{\theta}_r - \dot{\delta}_r \tag{3} \\
\dot{x}_c \cos \phi + \dot{y}_c \sin \phi - b \dot{\phi} &= r \dot{\theta}_l - \dot{\delta}_l \tag{4} \\
\dot{y}_c \cos \phi - \dot{x}_c \sin \phi - \phi \dot{d} &= \dot{\zeta}. \tag{5}
\end{align*}
\]

The first two constraints can be combined and integrated to form the holonomic constraint:

\[
2b \dot{\phi} - r(\dot{\theta}_r - \dot{\theta}_l) + \delta_r - \delta_l = C_1 \tag{6}
\]

We can combine the constraints into one equation \( A(q) \ddot{q} = a(q) \). Using the method of [11], we create a vector of generalized velocities \( \nu \) and create the state vector \( x = [q, \nu]^T \). The dynamics of this system are:

\[
\begin{align*}
\dot{x} &= \begin{bmatrix}
S\nu + \eta \\
-(S^T M S)^{-1}(S^T M \gamma + S^T C \ddot{q} + S^T G)
\end{bmatrix} \\
&\quad + \begin{bmatrix}
0 \\
(S^T M S)^{-1} S^T B
\end{bmatrix} u
\end{align*} \tag{7}
\]

where \( \eta(q) \) is a particular solution for the constraint equation \( A(q) \ddot{q} = a(q) \), \( S(q) \) is a matrix whose columns span the null space of \( A(q) \), and \( \gamma = S(q) \nu + \dot{\nu}(q) \).
Fig. 2: Block diagram of the supervisory control system

B. Traction Model

The values of $H_{r,l}$ and $K_{r,l}$ are calculated using the method outlined in [3]. This method assumes a pneumatic tire. The frictional force is a function of both the lateral ($s_L$) and longitudinal ($s_S$) wheel slip values [3]. To define slip, we first introduce radial velocity $v_R$ and the inertial velocity $v_W$ for each wheel. Mathematically, these are defined as $v_W = \sqrt{\dot{\rho}^2 + (r \dot{\theta}_{r,l} - \delta_{r,l})^2}$ and $v_R = r \dot{\theta}_{r,l}$. We also define the side slip angle as $\alpha = \arctan\left(\frac{\dot{\rho}}{r \dot{\theta}_{r,l} - \delta_{r,l}}\right)$. Using these two velocities, we can write the slip for the wheels as:

$$s_L = \begin{cases} \frac{v_R \cos \alpha - v_W}{v_R \cos \alpha}, & v_R \cos \alpha \leq v_W \\ \frac{v_R \cos \alpha - v_W}{v_R \cos \alpha}, & v_R \cos \alpha > v_W \end{cases}$$

$$s_S = \begin{cases} \frac{v_R \sin \alpha}{v_W}, & v_R \cos \alpha \leq v_W \\ \tan \alpha, & v_R \cos \alpha > v_W \end{cases}$$

The coefficient of friction is found using Burckhardt’s formula, using values for different surfaces (See [3]). The lateral and longitudinal forces are converted to the body frame as $H_{r,l}$ and $K_{r,l}$. Since the friction also depends on the normal force $N_{r,l}$, at the point of contact, we derive

$$N_l = \frac{mg (1 + 2 \frac{h}{b} \mu_{K_r})}{2 + \frac{h}{b} (\mu_{K_r} - \mu_{K_l})}$$

$$N_r = \frac{mg (1 - 2 \frac{h}{b} \mu_{K_r})}{2 + \frac{h}{b} (\mu_{K_r} - \mu_{K_l})},$$

where $\mu_{K_{r,l}}$ is the coefficient of friction in the lateral direction for each wheel. The complete UGV model is coded up as a series of MATLAB functions to provide a simulation platform for a slipping UGV.

IV. SUPERVISORY TRACTION CONTROL

In this section, we discuss controlling a slipping UGV using the model from Section III. The reference command inputs are the forward and angular velocities of the system. These commands can come from a human operator in a teleoperation maneuver or from an autonomous path planner.

When operating in slippery conditions, the response to a command can introduce excessive slip. Since longitudinal slip reduces the lateral friction (see [3] for a discussion of this topic), this slip can reduce maneuverability for the UGV. Also, the slipping UGV can execute a longer path, possibly colliding with obstacles the operator is trying to avoid.

A simple solution is to estimate the friction coefficient $\mu$ and saturate motor torque commands to $u_{r,l} \leq \mu N_{r,l}$. While this simple solution would limit the slip, it would not deal with errors caused by taking longer paths. To address the path issue, the supervisory traction controller identifies situations where the UGV is or will be slipping. The trajectory mode then plans a trajectory based on the commanded velocities and a desired position derived from the commands. We include the position in our controller, even though the commands are velocities, because fundamentally, all ground robot control is focused on positioning the robot; the position control loop is either handled in by a human (in teleoperation mode) or by an autonomous planner. This supervisory traction controller will improve the velocity following by adding a sense of position, reducing disturbances to higher controllers.

We divide control into two modes: PID and trajectory planning. When no slip is detected, PID control is used to achieve the commanded velocity. When slipping occurs, a trajectory is planned by optimizing over a cost function penalizing deviations in final position, final velocity, and slip along the path. Since the optimal trajectory is formed using the same dynamical model as was used in PID analysis, we can use the generated input as an open loop control. In a physical vehicle, we can either use the input in an open loop control or can use a trajectory following control law to follow the desired trajectory.

In this paper, we will assume that the friction coefficient is known and all state variables are available for feedback. While we are taking UGV position into account while planning the trajectory, we only address the stability of the system with regard to velocity tracking and leave position tracking to higher level controllers. This controller will limit position errors caused by slipping, but will not track a path.

A. Switching Criteria

We define the following switching criteria to enter the trajectory planning mode:

- Lateral slip velocity $\zeta$ exceeds some threshold
- Longitudinal slip velocity $\delta$ exceeds some threshold
- Control input would exceed the ground reaction force

Mathematically, these conditions are

$$|\zeta| > \zeta_{max}$$

$$|\delta_{r,l}| > \delta_{max}$$

$$|u_{r,l}| > \mu_{max} N_{r,l}.$$  (12)

When operating in PID mode, if any of these criteria are met, the controller switches to the trajectory planning mode. A full trajectory is planned and executed, then the switching criteria are reevaluated. If any of the criteria for slip are still met, the controller plans another trajectory. The controller returns to PID mode when the switching criteria are not met. Algorithm 1 shows the calculation of the control input.

Currently, while the controller is in trajectory planning mode, it ignores changes in the commanded velocities. When a new trajectory is planned, the controller uses the current velocity commands. Future work is required to determine when to replan the trajectory due to changes in commands.
Algorithm 1 Structure of supervisory traction controller

\begin{align*}
  k &= 0 \\
  \textbf{while } k \geq 0 \textbf{ do} \\
  &\quad \text{Read states } x \\
  &\quad \text{Define } e = [v_d, \omega_d] - [v, \omega] \\
  &\quad \textbf{if } \text{None of the conditions in (12) are met then} \\
  &\quad \quad \text{Calculate PID command} \\
  &\quad \quad v = K_P e + K_D \dot{e} + K_I \int_0^t e \, dt \\
  &\quad \quad u = [v_1 + v_2; v_1 - v_2] \\
  &\quad \quad (\text{Convert from velocity to motor commands}) \\
  &\quad \quad \text{Apply input } u \\
  &\quad \quad k = k + 1 \\
  &\quad \textbf{else} \text{ Trajectory Planning Mode} \\
  &\quad \quad \text{Choose } t_f \text{ and } (x_f, y_f) \\
  &\quad \quad \text{Plan trajectory based on cost function (13)} \\
  &\quad \quad \text{Determine inputs } u_k \text{ to } u_{k+t_f} \\
  &\quad \quad \textbf{for } i = k \rightarrow k + t_f \textbf{ do} \\
  &\quad \quad \quad \text{Apply input } u_i \\
  &\quad \quad \textbf{end for} \\
  &\quad \quad k = k + t_f \\
  &\quad \textbf{end if} \\
  \textbf{end while} \\
\end{align*}

B. Trajectory Planning

The planned trajectory must be feasible as well as limiting slip, converging to the velocity commands and limiting distance errors to the desired position. We use the following cost function to penalize slip and errors in final position \((x_f, y_f)\), heading \(\phi_f\) and velocity \((v_f, \omega_f)\) at the end of the trajectory from the corresponding desired values:

\begin{equation}
  J = K_\phi (\phi_f - \phi_d)^2 + K_d [(x_f - x_d)^2 + (y_f - y_d)^2] + K_v (v_f - v_d)^2 + K_\omega (\omega_f - \omega_d)^2 + \int_0^{t_f} v^T C v \, dt \tag{13}
\end{equation}

We can formulate \(C\) as \(S^T \hat{C} S\), where \(\hat{C}\) is a diagonal matrix that penalizes the slip states. Using this cost function, we can find an optimal trajectory that the UGV can traverse. We solve this optimization problem using the fmincon function from MATLAB, a numerical optimization technique.

In order to use this cost function, we need to determine the desired final position \((x_d, y_d)\), heading \(\phi_d\) and \(t_f\). The final position is based on the commanded velocities by creating a circular arc originating from the starting location with the radius of the circle equal to \(r = v_d/\omega_d\). The length of the arc is proportional to the error in velocity; larger velocity errors will have larger arcs. A length \(L\) is calculated as follows:

\begin{equation}
  L = \max(e_v^0 P_v, e_\omega^0 P_\omega) \\
  L = \min(L_0, L_{\text{max}}, \Psi_{\text{max}} r) \tag{14}
\end{equation}

where \(e_v^0 = v_0 - v_d\) and \(e_\omega^0 = \omega_0 - \omega_d\), \(v_0\) and \(\omega_0\) are the initial velocities, \(P_v\) and \(P_\omega\) are constants, and \(\Psi_{\text{max}}\) is the maximum angle of the arc. To determine the final time \(t_f\), we calculate the nominal time \(t_n = L/v_d\) to traverse the arc; \(t_f\) is calculated as

\begin{equation}
  t_f = (L^2/P_t + 1) \, t_n \tag{15}
\end{equation}

where \(P_t\) is a surface-specific gain. This equation for \(t_f\) was determined by trial and error; linear or constant multipliers required more iterations of trajectory planning.

C. Stability Analysis

For the trajectory planning mode to be useful, the UGV needs to track the desired velocities, even when the controller stays in the trajectory planning mode for several iterations. To guarantee that the velocity errors are bounded, we will show that, with sufficiently large \(K_v\) and \(K_\omega\), the weighted norm of the velocity errors \(||e|| = K_v e_v^2 + K_\omega e_\omega^2 \leq ||e_s||\), where \(||e_s||\) is the norm at the beginning of the trajectory, \(e_v = v_f - v_d\), and \(e_\omega = \omega_f - \omega_d\). We first look at a simpler cost function by setting \(K_d = K_\phi = 0\).

First, choose an input \(u = u_0\) to the system (7) such that \(||e_0|| = ||e_s||\). Such a trajectory can be constructed through a linearization of the the input-output system of \(u\) to \([e_s, \dot{e}_s]\). The relative degree of this system is two and a full-state feedback can be designed by nonlinear analysis to match the goal condition. The input will not saturate because the only external forcing on the UGV is friction, which can only dissipate energy from the system. If the energy dissipated by friction would cause \(u\) to saturate, then the UGV would not be slipping and the controller would remain in PID mode.

We can calculate \(J_0 = [||e_0|| + \int_0^{t_f} C v_0 \, dt]. \) Since velocity errors are the same at \(t_0\) and \(t_f\), we can calculate \(||e_s||\).

Next, suppose there is a different \(u\) such that \(\int v^T C v \, dt < \int v_0^T C v_0 \, dt\). If we increase \(K_v\) and \(K_\omega\) sufficiently, any possible increase in \(||e||\) will be more than offset the decrease in \(\int v^T C v \, dt\), making \(J > J_0\). Thus, this trajectory cannot be optimal. Conversely, if there is another \(u\) such that \(\int v^T C v \, dt > \int v_0^T C v_0 \, dt\) and \(e_s\) or \(e_\omega\) decreases over the interval, by increasing \(K_v\) or \(K_\omega\), this trajectory will have a smaller \(J\).

The optimal input will be either \(u_0\) or some \(u\) such that \(||e|| < ||e_s||\). Creating a series of these trajectories one after another will be stable in the sense that \(||e||\) will not increase and, if there exists a control that decreases \(||e||\), with sufficiently large \(K_v\) and \(K_\omega\), \(||e||\) will tend to 0.

For the original cost function (13), a similar analysis holds. Instead of only \(\int v^T C v \, dt\) being able to decrease, the terms \(K_\phi (\theta_f - \theta_d)^2\) and \(K_d [(x_f - x_d)^2 + (y_f - y_d)^2]\) can also decrease. We can make a similar argument that sufficiently large \(K_v\) and \(K_\omega\) will result in \(||e|| \leq ||e_0||\). The limiting values of \(K_v\) and \(K_\omega\) will depend on the current friction values as well as \(t_f\). We found that decreasing the values of \(t_f\) caused the velocity to converge to the commands faster.

D. Robustness Analysis

The trajectory planning and the switching criteria depend on knowing the friction coefficient at the current position. In a simulation environment, we know the parameters exactly. In a physical setting, the friction coefficient needs to be estimated (see [15] for an example). Underestimating the friction coefficient will cause the trajectory planning mode to be entered when the UGV could actually execute the commanded torques without slipping. From (12), if

\[ where \ P_t \ is a surface-specific gain. This equation for \(t_f\) was determined by trial and error; linear or constant multipliers required more iterations of trajectory planning.
Traction Control
PID− motor limits
PID− friction limits
(a) Reconfigurable has less deviation from a desired circle

Fig. 3: Starting with both wheels slipping at .045 m/s, desired velocity \( [v_d, \omega_d] = [2 \text{ m/s}, \pi/3 \text{ rad/s}] \), on snow

<table>
<thead>
<tr>
<th>Parameter</th>
<th>mass</th>
<th>wheel radius</th>
<th>h</th>
<th>h</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>( K_P )</td>
<td>( K_I )</td>
<td>( K_D )</td>
<td>( \mu_{\text{max}} )</td>
<td>Value</td>
</tr>
<tr>
<td>Value</td>
<td>([8, 5]^T)</td>
<td>0</td>
<td>0</td>
<td>150 Nm</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>( K_V )</td>
<td>( K_\omega )</td>
<td>( K_d )</td>
<td>( K_\phi )</td>
<td>Value</td>
</tr>
<tr>
<td>Value</td>
<td>8/30(^1)</td>
<td>8/20(^1)</td>
<td>100/12(^1)</td>
<td>10/60(^1)</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>( \tilde{C} )</td>
<td>Value</td>
<td>diag((0, 0, 0, 0, 0, 40, 65))</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( \mu_{\text{ext}} < \mu_{\text{max}} \), a smaller torque will satisfy the switching condition. While the resulting trajectory would underestimate the mobility of the UGV and limit the performance, the trajectory would be feasible and the UGV could operate.

Overestimating the friction coefficient would delay triggering the switching conditions until the slip exceeds the threshold. Additionally, the planned trajectory would exceed the UGV’s capabilities and would introduce more slip, which is what the controller is attempting to prevent. Therefore, when estimating the friction coefficient in a physical system, it is imperative to not overestimate the friction coefficient.

V. SIMULATION RESULTS

For the simulation, we used the parameters of a wheeled similar in size to an iRobot Packbot (See Table I for values). We simulated the UGV on asphalt and snow surfaces, using friction parameters from [3]. We compare the supervisory controller with PID control, using two different input saturations. In the first case, the input is saturated based on the motor torque limits (motor limits). In the second case, the inputs are saturated based on the friction force (friction limits). The PID gains for this example were manually tuned for the asphalt surface.

A. Circular Trajectory

In the first scenario, the UGV is given a constant desired velocity of \( [v_d, \omega_d] = [4 \text{ m/s}, \pi/2 \text{ rad/s}] \) on snow. Initially, both wheels are slipping at 0.045 m/s. Figure 3 compares three different controllers for this scenario.

Figure 3(a) shows the \( x-y \) trajectory for the three different controllers. The PID-motor limits trajectory has large slip values and cannot execute the circle. The PID-friction limits trajectory completes a circle with moderate displacement. Using the supervisory traction controller, a circular trajectory is completed near the origin. Figure 3(b) shows the velocity profiles for the different controllers. The PID-motor limits case has large overshoot in both velocities and does not reach steady state. The PID-friction limits case accelerates forward faster than the traction controller, but the traction controller accelerates faster in angular velocity. In both the PID-friction limits case and the supervisory traction controller, the slip is reduced to negligible amounts.

The PID-friction limits could be tuned to give a similar response to the traction controller. However, to make a PID system robust to different surfaces, distinct tunings would be needed for every surface.

B. Response to Change in Command

The second scenario involves changes in the reference commands as the UGV travels. The UGV is commanded to travel straight forward for a certain distance, then a constant turn command is given in addition to a forward velocity. After the UGV turns 180\(^\circ\), the UGV is commanded to go straight forward again. Specifically, the input is as follows:

\[
\begin{bmatrix}
    v_d \\
    \omega_d
\end{bmatrix} = \begin{cases} 
    \begin{bmatrix} 4, 0 \end{bmatrix}^T & x_c < 5 \text{ or } \phi > \pi \\
    \begin{bmatrix} 1, 1 \end{bmatrix}^T & \text{else}
\end{cases}
\]

(16)

The trajectory was executed on two different conditions: once on asphalt under PID control, and twice on snow, once under PID-friction limits, and once under supervisory traction control. Figure 4 shows the \( x-y \) plot for the different scenarios. The x marks denote one second intervals along the trajectory. The asphalt scenario is able to quickly execute the desired turn and, after the turn command is removed, responds by ending the turn more quickly than either case on snow. Comparing the two runs on snow, the supervisory controller completes the desired turn in a smaller circle and stops turning more quickly than the PID case.

The velocity profile, shown in Figure 4(b), shows that the traction controller executes the turn at a lower speed for longer than the PID controller. This is due to trying to match the position as well as the desired velocities. Decreasing the speed allows for a tighter turn to be completed, a desirable result of this controller.

4360
This paper developed a dynamic model for a slipping UGV, using the Lagrangian approach. Using this model, we presented a supervisory traction controller that switches from a PID law to a trajectory planning mode when excessive wheel slip is or will be encountered. In this trajectory planning mode, we trade off commanded velocity and the desired position based on velocity commands. We demonstrated this control through simulations on asphalt and snow.

In future work, we plan on implementing this controller on a physical system. To model slopes and other terrain features, a gravity vector will be added to the model. In addition, we can add controllers for other faults, such as complete immobilization of the UGV in sand or other terrain, as well as designing a path planner that utilizes the predicted location information provided by the supervisory traction controller.

VI. CONCLUSION

REFERENCES


