2023 NDIA MICHIGAN CHAPTER
GROUND VEHICLE SYSTEMS ENGINEERING
AND TECHNOLOGY SYMPOSIUM
AUTONOMY, ARTIFICIAL INTELLIGENCE & ROBOTICS (AAIR) TECHNICAL SESSION
AUGUST 15-17, 2023 - NOVI, MICHIGAN

A RUNTIME MONITOR FOR PLATFORM PROTECTION AGAINST SKID-STEER UNTRIPPED ROLLOVERS

Elizabeth Dietrich[1], Sara Pohland[1], Daniel Genin[1], Aurora Schmidt[1], Gautam Vallabha[1], Anthony Composto[2], Marcus Randolph[2]

[1]Johns Hopkins University Applied Physics Laboratory, Laurel, MD
[2]Ground Vehicle Systems Center (GVSC) Ground Vehicle Robotics (GVR), Warren, MI

ABSTRACT

Unmanned ground vehicles (UGVs) that autonomously maneuver over off-road terrain are susceptible to a loss of stability through untripped rollovers. Without human supervision and intervention, untripped rollovers can damage the UGV and render it unusable. We create a runtime monitor that can provide protection against rollovers that is independent of the type of high-level autonomy strategy (path planning, navigation, etc.) used to command the platform. In particular, we present an implementation of a predictive system monitor for untripped rollover protection in a skid-steer robotic platform. The system monitor sits between the UGV's autonomy stack and the platform, and it ensures that the platform is not at risk of rollover by intercepting mobility commands sent by the autonomy stack, predicting platform stability, and adjusting the mobility commands to avoid potential rollovers. We demonstrate our implementation through experiments with skid-steer UGVs in Gazebo simulation and physical experiments.

Citation: E. Dietrich, S. Pohland, D. Genin, A. Schmidt, G. Vallabha, A. Composto, M. Randolph, "A Runtime Monitor for Platform Protection Against Skid-steer Untripped Rollovers," In *Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium* (GVSETS), NDIA, Novi, MI, Aug. 15-17, 2023.

1. INTRODUCTION

Unmanned ground vehicles (UGVs) that maneuver aggressively with a variety of high-level autonomy strategies (for path planning, navigation, etc.) are susceptible to a loss of stability even when not in contact with an obstacle. For example, when a UGV turns sharply at speed, its wheels or tracks can lift off the ground for kinematic reasons and lead to a rollover. These *untripped rollovers* are challenging to anticipate and avoid even for trained hu-

man operators, particularly in rugged or sloped terrain. The challenge is exacerbated for autonomous UGVs, since remote teleoperation increases the risk of rollover, which can damage the UGV and render it unable to complete its mission.

A solution to protecting vehicle platforms while assuring safe operations was presented in the Safety Reasoning System concept of [14], which proposes an independent safety monitoring component. We present a runtime monitor to protect against these untripped rollovers to allow the assured operation of the platform autonomy. Given planned mobility commands, our runtime monitor performs a safety computation (using a dynamics model of the UGV) to determine if the platform will be put at risk if it executes the commands. If the platform is at risk, the runtime monitor issues a braking command; otherwise, it continues with the original mobility commands.

In the remainder of this paper, we review existing work related to our proposed method in Section 2. We then develop a dynamics model for a skid-steer system (building on existing state-of-the-art) and design a runtime monitor using the dynamics model in Section 3. In Section 4, we validate the model for dynamics predictions with physical experiments and demonstrate the effectiveness of the runtime monitor in simulation. We focus on skid-steer systems as they are a common category of UGV for traversing complex terrain; however, the basis of our approach is generally applicable to other kinds of UGVs as well.

2. RELATIONSHIP TO EXISTING WORK

Current efforts to combat rollover of autonomous UGVs propose human-intervention [10] or platform alterations [11]. Our work uses a simplified design that decouples the basic rollover safety from high-level autonomy. The rollover safety is implemented by a software layer that sits between the autonomy and the platform, monitoring the mobility commands and overriding them (with braking commands) if needed. This allows the autonomy to be agnostic to low-level platform details. Additionally, it allows the use of complex autonomy approaches that cannot be fully verified for correct operations.

Detecting the possibility of an untripped rollover requires accurate prediction of skid-steer angular turn rates, which in turn requires an accurate motion model of the UGV. Existing methods to predict turn rates of skid-steer platforms provide inaccurate results. Current approaches to combat these inaccurate odometry estimates include implementing robust motion control algorithms [7] and improving kinematic modeling through data-driven techniques [8, 9]. Dogru and Marques [5] address these limitations by taking into account both the center of mass of the robot and its size. Our approach incorporates the improved kinematic model of Dogru and Marques [5] and the dynamics model presented by McCormick, et al. [6]. By combining these two models, we create an improved dynamics model with increased accuracy of turn rate predictions that we use as the basis for our predictive runtime monitor.

We validate our dynamics model using a physical Clearpath Jackal and assess our runtime monitor with a simulated Clearpath Husky. While both these platforms are skid-steer, our approach can later be applied to other types of UGVs. The design of our runtime monitor allows us to modularize platform-specific details in two ways. First, our skid-steer dynamics model is parameterized so that it can be easily adapted to a variety of skid-steer platforms. Second, it is possible to replace the skid-steer dynamics model with the dynamics model for any other steering mechanism, so it is straightforward to extend our approach to any desired Army platform.

3. APPROACH

In this section, we describe our approach to developing the skid-steer model, which is used to infer the state of the vehicle based on internal measurements. We then present the formulation of the risk prediction that estimates the potential for untripped rollover based on predicted vehicle state. Using the reachability method for risk prediction, we

develop the method for runtime monitoring that inspects and regulates the vehicle motion commands to ensure risky states are prevented.

3.1. Skid-Steer Platform Model

Skid-steered platforms have wheels that are fixed longitudinally parallel to the structure. All of the wheels are connected to the motors and contribute to the traction of the platform. Skid-steered platforms rotate by varying the speeds of the wheels on the left and right sides [5]. One side's wheels move faster than the other, causing the vehicle to skid. We analyze both the kinematics and dynamics of skid-steer platforms to achieve high precision predictions of vehicle motion.

Kinematic modeling of skid-steered platforms can be divided into two parts: the straight motion and the rotations. The modeling of rotations, which occur due to skid of the platform is complicated and depends on each side's wheel speed, geometry of the platform, center of mass, and the surface [5]. We utilize the improved kinematics model presented by Dogru and Marques [5]. The approach they utilize derives the turn rate of a skid-steered robot, assuming it is rotating at a point close to its center of mass. We improve our rotational velocity predictions through the equation they present:

$$\omega = \frac{(u_L - u_R)}{\beta} \tag{1}$$

where u_L and u_R are the translational speeds of a point on the surface of the wheels with respect to the axis of the wheel. These values can be calculated using $u_L = r\Omega_L$ and $u_R = r\Omega_R$, where r is the radius of the wheels and Ω_L and Ω_R are the rotational velocities of the left and right wheels respectively. The parameter β allows us to define the wheel base of our platform. In our experiments, this parameter was estimated by rotating the robot in place at different speeds and assuming $u_L = -u_R$. For further implementation details, we direct readers to [5].

To further model the complex motion of skid-steer platforms, we utilized the highly accurate linear graph (LG) dynamics model proposed by McCormick, et al. [6]. This model was specifically designed for the Clearpath Husky vehicle. The LG model consists of various subsystems that encompass multiple physical domains and functions of the robotic system, including the electrical subsystem, the drivetrain subsystem, the axles and wheels for both of the independent left and right side powertrains, and the translational and rotational subsystems [6]. This allows for representation of the linear and rotational movements of the entire robot through the state-space model:

$$\frac{dx}{dt} = A \begin{bmatrix} \omega_{J_{LW}} \\ \omega_{J_{RW}} \\ v_{M_H} \\ \omega_{J_H} \\ i_{L_{L1}} \\ i_{L_{L2}} \end{bmatrix} + B \begin{bmatrix} V_{s_1} \\ V_{s_2} \end{bmatrix} \tag{2}$$

In Equation 2, $\omega_{J_{LW}}$ and $\omega_{J_{RW}}$ are the rotational velocities of the left and right wheels respectively, v_{M_H} and ω_{J_H} are the linear and rotational velocities of the skid-steer platform, and $i_{L_{L1}}$ and $i_{L_{L2}}$ represent the current through the left and right motors. Additionally, in Equation 2, V_{s_1} and V_{s_2} estimate the voltage inputs to the state-space model as

$$V_{s_1} = 24 \cdot (c \cdot v_t + 0.541 \cdot c \cdot v_r) \tag{3}$$

$$V_{s_2} = 24 \cdot (c \cdot v_t - 0.541 \cdot c \cdot v_r) \tag{4}$$

In the above equations, v_t is the commanded translational velocity and v_r is the commanded turn rate. To conserve space, we will not provide the implementation details of matrices A and B, as well as the full derivation of the dynamics model, which are provided by McCormick, et al. [6].

We use the predictions of the rotational velocities of the left and right wheels from this model in Equation 1 to improve the predictions of skid-steer

angular velocity. The results of experimental testing of this approach are presented in Section 4.

3.2. Runtime Monitor

The objective of the rollover runtime monitor is to protect UGVs against untripped rollovers, increasing the overall safety and stability of the vehicle. Skid-steer platforms are difficult to maneuver on sloped surfaces, especially on rough terrains. These rollovers can be attributed to skid-steer platforms performing aggressive maneuvers involving fast linear or rotational velocity commands. By utilizing a rollover runtime monitor, we can ensure that autonomy commands do not cause the platform to rollover and can mitigate unsafe commands by reducing linear speed, even on steep slopes.

The rollover runtime monitor utilizes the model described in Section 2 and a Rollover Index (RI) based on the Load Transfer Ratio (LTR). The LTR is a measure of the load on each wheel of the platform, which can be used to determine when the wheels lift off the road [1] (see Figure 1).

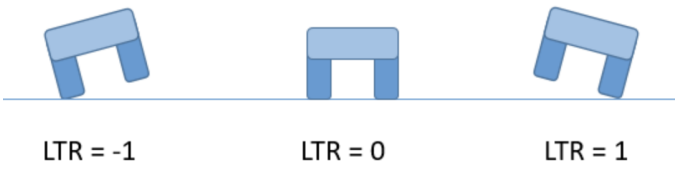

Figure 1: LTR represents the load distribution between wheels. 0 corresponds to load being equally distributed and ± 1 corresponds to load being completely on one set of wheels.

Since it is difficult to directly measure vertical tire loads, the LTR is usually estimated from other vehicle states, including vehicle lateral acceleration, yaw rate, roll angle, etc. [2]. The LTR is one of the most commonly used RIs, and has multiple definitions. We chose the following LTR formulation, presented by Imine, et al. [1]:

$$LTR = \frac{2(h + h_R)}{T}\frac{a_y}{g} + \frac{h}{T}\phi \tag{5}$$

where h is the height of the center of gravity in relation to the roll axis, h_R is the roll axis height in relation to the ground, $(h + h_R)$ is the height of the vehicle's center of gravity, T is its track width, g represents gravitational acceleration, ϕ is the platform roll angle, and a_y is the lateral acceleration of the vehicle's geometric center which can be calculated by:

$$a_y = v\psi + gsin(\phi + \xi) \tag{6}$$

where we additionally have v, the vehicle speed, ψ, the yaw rate, and ξ, the lateral road inclination.

Since the skid-steer platforms we are considering do not have substantial suspension systems, it can be assumed that ϕ will be zero or close to zero. Additionally, we redefine $(h + h_R)$, the height of the vehicle's center of gravity, as h_{cg}, as well as v, the vehicle speed, and ψ, the yaw rate, to be v_H and w_H respectively. By substituting for a_y in Equation 5, the LTR utilized in our proposed runtime monitor is calculated under these assumptions as follows:

$$LTR = \frac{2h_{cg}}{T}\left(\frac{v_H w_H + gsin(\xi)}{g}\right) \tag{7}$$

When the left (right) wheels lift off of the road, and a rollover occurs, the LTR = 1 (-1). The runtime monitor relies on predictions from our skid-steer model and the LTR to determine what commands will be safe for the platform to ensure $|LTR| < 1$.

The design of the rollover runtime monitor is based on the safety analysis originally developed for stress testing the Airborne Collision Avoidance System X (ACAS X), the Federal Aviation Administration's (FAA) next-generation collision avoidance advisory system for aircraft [3], [4], [13], and was additionally applied for pedestrian avoidance in [15]. In [4], we developed a key property of system state termed *safeability*. A state is safeable if at the next decision iteration the system can be made safe through the use of an available maneuver (e.g., safe stop), regardless of what command is executed on the

current time step. The safeability property is an extension of a traditional *safe* property, which is constructed for a specific model of own-vehicle dynamics with known bounds on sensor and environmental uncertainty. Under the system assumptions, the safeability property allows us to determine precisely when the current commands must be overridden to maintain safety.

Conceptually, the state space of the system can be divided into mutually disjoint sets corresponding to varying levels of safety. During normal operation the system resides within the safeable set, where it can be guaranteed from rollover for any primary autonomy command. Due to changes in terrain, path curvature or speed, it may move from the safeable set into the critical set, in which only a subset of available maneuvers can guarantee future safety. When the system is in a critical state, we restrict the control to the set of safe maneuvers, so that an excursion into an unsafe state is be prevented. The unsafe set is the set of states where system safety can no longer be guaranteed by the runtime monitor, but failure, i.e., rollover, may or may not actually occur, depending on the tightness of the bounds used.

Crucially, the partitioning into subsets is such that it is not possible for the system to pass from a safeable state into an unsafe state without visiting the critical set. This allows the runtime monitor a chance to prevent an excursion into the unsafe set. This follows directly from the definition of safeability, which guarantees that any successor state of a safeable state cannot be unsafe. In practice, this means that to determine if the state is safeable it is necessary to perform reachability analysis, and that the safeability of a state depends on the available remediation maneuvers.

The proposed rollover runtime monitor performs the described reachability analysis by repeatedly calculating LTR on each decision iteration using the improved skid-steer model and the provided mobility command.

We design the rollover runtime monitor to check the state of the system at regular intervals. The update rate may be chosen to be faster than the following analysis, but for the purposes of this demonstration, we chose an update rate of once per second. The runtime monitor checks the current state of the platform and the current mobility command at this interval. The safeable set marks the velocities and turn rates the platform can travel, in which a braking command that is issued at the next decision interval would be able to prevent any potential rollover. As such, when the vehicle is in the safeable set, no intervention is necessary. The critical set corresponds to vehicle velocities and turn rates at which only a restricted set of commanded future velocities and turn rates would be able to guarantee the future absence of rollover. To calculate the boundary of the critical set, we first compute the expression for the critical LTR.

According to Clearpath Husky documentation, the maximum linear and rotational deceleration of a Husky does not exceed 1.0 m/s^2 and 2.0 rad/s^2 in absolute value, respectively. Therefore, in one second, linear and rotational velocities can change by at most 1.0 m/s and 2.0 rad/s in absolute value, respectively. When we detect an unsafe state and motion, we will modify the linear and rotational velocities to 1.0 m/s and 2.0 rad/s slower, respectively, than in the unsafe ($|LTR| \geq 1$) state.

We denote the predicted linear velocity by v_p and the predicted rotational velocity by w_p, where each value (v_p and w_p) uses the worst-case upperbound derived in section 4.3. We calculate the critical LTR by adding these values to the current predicted linear and rotational velocity, v_p and w_p.

$$LTR_{crit} = \frac{2h_{cg}}{T}\left(\frac{(v_p - 1)(w_p - 2) + g\sin(\xi)}{g}\right) \tag{8}$$

Given the critical LTR from Equation 8, we can calculate our critical command set defined by the lin-

ear velocity, v_c, and rotational velocity, w_c:

$$v_c = \frac{1}{w_p}\left(\frac{(LTR_{crit} \cdot T \cdot g)}{2h_{cg}} - g\sin(\xi)\right) \quad (9)$$

$$w_c = \frac{1}{v_p}\left(\frac{(LTR_{crit} \cdot T \cdot g)}{2h_{cg}} - g\sin(\xi)\right) \quad (10)$$

In practice, our monitor receives the planned autonomy commands. It then performs the previously defined safety computations to determine whether the planned commands are safe to execute until the next decision cycle update. If the safety computation returns a result in the critical or unsafe set, the runtime monitor issues the target speed and turn rate of $(0, 0)$. This zero command effectively slows down the vehicle during the following control interval. If the vehicle, reenters the safeable set of velocity states, then at the next decision interval the command from the primary autonomy is again passed through to the low-level platform controller; in other words, the intervention to brake is non-locking. In future development, we plan to test a range of speed commands that are not targeting a full stop, which we could accomplish by deriving the set of critical states based on the planned slow-down targets. In our experiments for this paper, and for simplicity, the safety command is a non-locking brake command.

4. RESULTS

We present the results of testing the improved kinematic model on a skid-steer platform to validate the turn rate estimates against motion capture recordings (see Sections 4.1 and 4.2). This data is used to estimate the errors associated with the turn rate estimates (Section 4.3), which are inputs to the runtime monitor for preventing untripped vehicle rollover. In Section 4.4, we test the runtime monitoring approach in simulation tests using Gazebo.

4.1. Experimental Setup

In order to validate the skid-steer model developed in this paper, we utilized the skid-steer robot Clearpath Jackal. Jackal is an approximately 1/2 scale model of the Clearpath Husky.

Jackal was driven by commanding linear and rotational velocities from 0 to 1 m/s and 0 to 0.7 rad/s respectively. Jackal contains an internal velocity limiter, which constrains the maximum linear and rotational velocity of Jackal to 1.75 m/s and 0.7 rad/s respectively. These tests were done on flat concrete and the Jackal's rotational velocity was recorded using the motion capture system, Optitrack [12].

4.2. Validation of Skid-steer Platform Model

Our improved dynamics model was validated by sampling 6 runs with a variety of linear and rotational velocities. Jackal was commanded to rotate at the velocities displayed in Table 1.

Table 1: Commanded Jackal Velocities for Validation of Skid-Steer Platform Model

Sample	Linear Velocity	Rotational Velocity
Run 1	0.5 m/s	0.1 to 0.7 rad/s
Run 2	0.5 to 1.0 m/s	0.1 to 0.7 rad/s
Run 3	0.5 to 1.0 m/s	0.5 rad/s
Run 4	0.5 m/s	0.5 rad/s
Run 5	1.0 m/s	0.7 rad/s
Run 6	0.5 m/s	0.1 rad/s

Table 2: Estimated vs. Measured Average Rotational Velocity for Jackal

Sample	Measured	Estimated
Run 1	0.4093 rad/s	0.4111 rad/s
Run 2	0.4175 rad/s	0.4209 rad/s
Run 3	0.5016 rad/s	0.5021 rad/s
Run 4	0.4905 rad/s	0.4865 rad/s
Run 5	0.6867 rad/s	0.6989 rad/s
Run 6	0.1126 rad/s	0.1211 rad/s

The dynamics model estimates the rotational velocity of Jackal given linear and rotational velocity

mobility commands. The measured and estimated rotational velocities for Jackal were compared at each time-step in the sample runs as seen in Figure 2. We found that the estimates of our proposed model closely match the Jackal measurements. Additionally, the average measured and estimated rotational velocities for each run are compared in Table 2.

4.3. Error Analysis

In order to create an effective runtime monitor for rollover protection, we analyzed our empirically collected measurements to estimate uncertainty bounds on the turn rate of the vehicle. The data sets of rotational velocities, measured and estimated, were smoothed using a Savitzky-Golay filter[1] and used to analyze the error of our proposed model. Table 3 shows the mean square error (MSE) and max point-wise difference for each sample run.

The error in measured versus estimated predicted angular velocities exhibits non-linear dependence on speed. We used a second degree polynomial approximation to model the relationship between error and speed. This approach allows us to keep the simplicity of the original model and at the same time account for possible deviations from the model behavior.

Table 3: Error Analysis for Jackal Experiment

Sample	MSE	Max Point-Wise Difference
Run 1	0.00005	0.0138
Run 2	0.00005	0.0266
Run 3	0.00002	0.0118
Run 4	0.00003	0.0158
Run 5	0.00020	0.0274
Run 6	0.00008	0.0169

For the error model, we utilized the sample runs on Jackal that recorded constant commanded velocities, Runs 3, 4, and 5. Given a commanded turn rate, R_c, and a commanded linear velocity, L_c the model error will be

$$\begin{matrix} \pm(0.1017R_c^2 - 0.06394R_c + 0.02231) \\ \pm(0.007942L_c^2 + 0.01016L_c + 0.009284) \end{matrix} \quad (11)$$

By using the max point-wise difference in our error analysis, our rotational velocity predictions will encompass the worst-case scenario. This is critical for the runtime monitor to be able to guarantee safety of the platform during rotations.

4.4. Simulation Tests of Runtime Monitor

In order to assess the performance of the proposed runtime monitor for platform protection against untripped rollovers, tests were performed in the Gazebo simulation environment on a Clearpath Husky. Gazebo is a 3D robotic systems simulator that is integrated with the ODE physics engine. Additionally, integration of Gazebo with ROS allows us to utilize ROS topics and record command signals and sensor data of the robot [6]. We built a world in Gazebo consisting of a variety of slopes ranging from 5 to 50 degrees (Fig. 3).

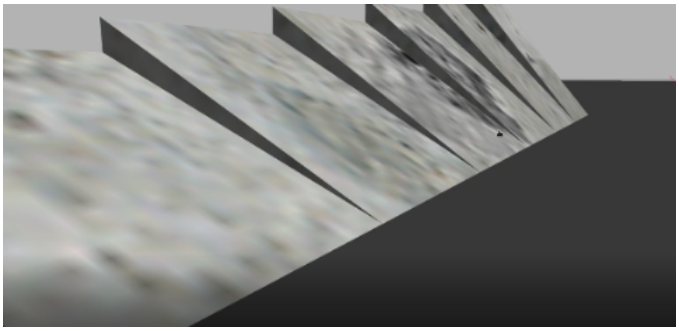

Figure 3: Gazebo simulated slopes used in Husky experiments. We identified which slope resulted in frequent rollovers (40°) and used that slope to test the rollover runtime monitor.

The 40-degree slope was selected due to the frequent occurrence of rollovers at this angle. The Husky was commanded a range of linear and rotational speeds between 0.5 to 1.0 m/s and 0.5 to 2.0

[1] As implemented by scipy Python package.

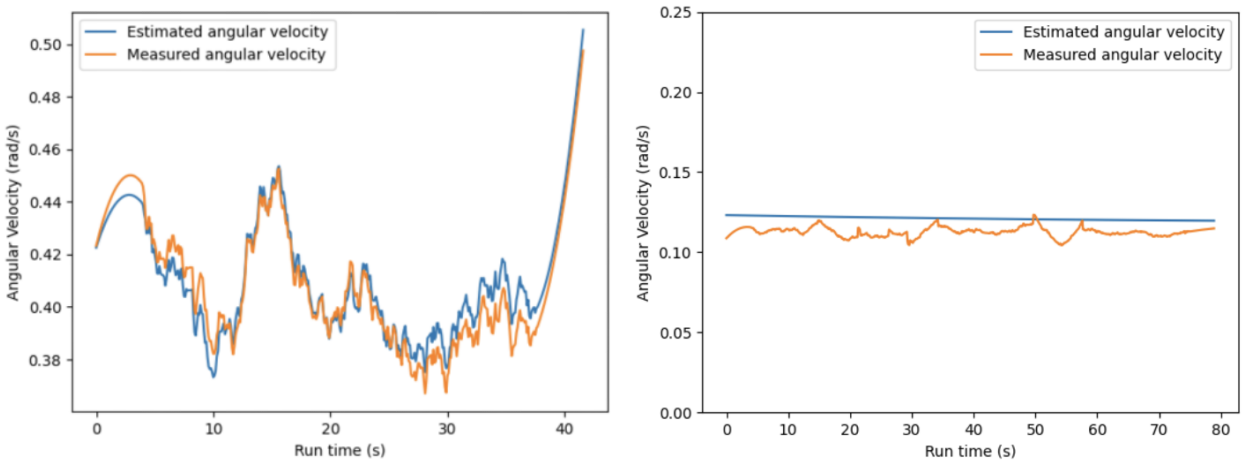

Figure 2: Estimated and measured velocity of Jackal for sample run 1 with 0.5 m/s linear command and a varying rotational command between 0.1 and 0.7 rad/s (left) and sample run 6 with 0.5 m/s linear command and 0.1 rad/s rotational command (right).

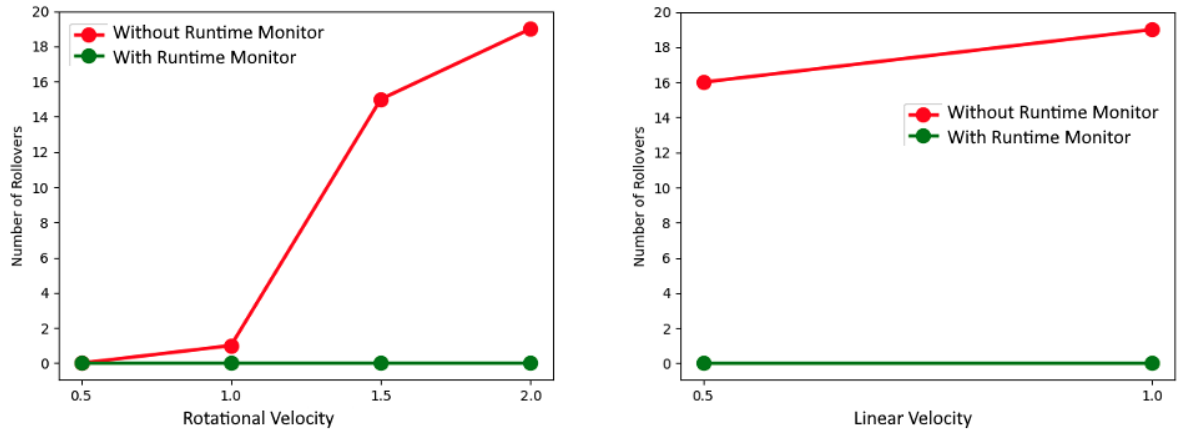

Figure 4: Safety performance comparing the number of rollovers the Husky underwent with and without the rollover runtime monitor, per commanded rotational velocity (left) and linear velocity (right). A total of 35 rollovers were observed.

rad/s respectively. The sampled velocities are displayed in Table 4. For each command tuple, the Husky underwent ten trials with and without the rollover runtime monitor, for a total of 20 runs.

The traversed trajectories of the Husky encompassed between two and four rotations depending on the mobility commands; an example trajectory is plotted on Figure 5. The Husky was initialized at the same position during each trial and performed multiple rotations. We found that rollover most often occurred when the Husky platform was pointed orthogonal to the slope of the ramp.

Table 4: Commanded Husky Movements on 40° Gazebo Simulated Slope

Sample	Linear Velocity	Rotational Velocity
Run 1	0.5 m/s	0.5 rad/s
Run 2	0.5 m/s	1.0 rad/s
Run 3	0.5 m/s	1.5 rad/s
Run 4	0.5 m/s	2.0 rad/s
Run 5	1.0 m/s	0.5 rad/s
Run 6	1.0 m/s	1.0 rad/s
Run 7	1.0 m/s	1.5 rad/s
Run 8	1.0 m/s	2.0 rad/s

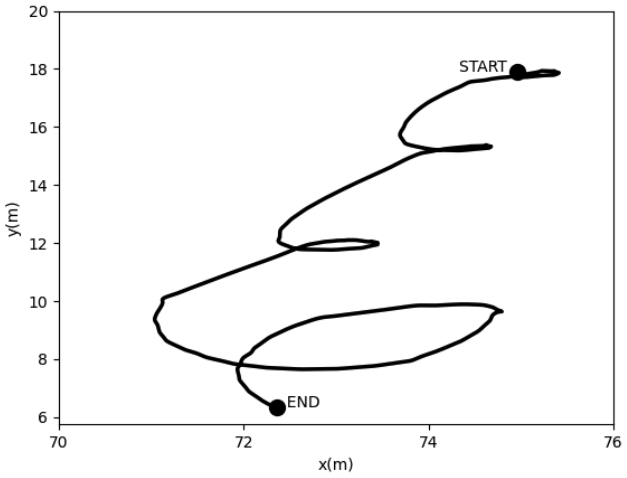

Figure 5: Sample Husky trajectory on the 40° slope (overhead view of slope). The Husky undergoes 2-4 rotations over its trajectory.

We found that the rollover runtime monitor was 100% effective at preventing untripped rollovers of the skid-steer Husky platform in the trials we conducted. Figure 4 shows the number of rollovers per rotational and linear velocities that the Husky encountered during the 80 test scenarios. We see an increase in the number of rollovers as the rotational and linear velocities increase.

5. CONCLUSIONS AND FUTURE WORK

This paper presents a runtime monitoring approach to protect skid-steer platforms against untripped rollovers. This system was implemented and tested using a simulated Clearpath Husky traversing a steep slope. The rollover runtime monitor successfully resolved the scenarios in which aggressive mobility commands, which were allowed without the rollover runtime monitor, resulted in platform rollover. This demonstrates the value of employing independent runtime monitoring to enable the adoption of complex autonomy systems.

In future work, we plan to include a set of intermediate safety actions that limit the velocity of the vehicle to safe ranges. Rather than correcting the vehicle with a braking command, the reachability approach that we employed can also derive critical thresholds for commands that reduce the velocity and turn rates of the autonomy commands. In the future we hope to formally verify the correctness of our runtime monitor. We are actively testing an implementation of our runtime monitor in outdoor field tests and plan to further validate the kinematics model on sloped surfaces, as well as cases where additional weight is placed on the vehicle. Additionally, we plan to extend this work to detecting changes in vehicle dynamics and to apply the method to a variety of platforms that exhibit different dynamics, such as Ackermann steering.

6. ACKNOWLEDGMENTS

This work was funded by U.S. Army DEVCOM Ground Vehicle Systems Center (GVSC). The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

7. REFERENCES

[1] Imine, Hocine and V. Dolcemascolo. “Rollover risk prediction of heavy vehicle in interac-

tion with infrastructure." International Journal of Heavy Vehicle Systems 14, 2007, pp. 294-307.

[2] Petar Jevtic, Yue Shi, and Yan Chen, "Loss Modeling of Rollover for Autonomous Vehicles," SOA Research Report, November, 2019.

[3] R. W. Gardner, D. Genin, R. McDowell, C. Rouff, A. Saksena, and A. Schmidt, "Probabilistic model checking of the next-generation airborne collision avoidance system," in 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC), 2016, pp. 1–10.

[4] J. Jeannin, K. Ghorbal, Y. Kouskoulas, A. Schmidt, R. Gardner, S. Mitsch, and A. Platzer, "A formally verified hybrid system for safe advisories in the next-generation airborne collision avoidance system," International Journal on Software Tools for Technology Transfer (STTT), vol. 9035, pp. 21–36, 2017.

[5] Sedat Dogru and Lino Marques. 2021. An improved kinematic model for skid-steered wheeled platforms. Auton. Robots 45, 2 (Feb 2021), 229–243.

[6] E. McCormick, H. Lang, and C. W. de Silva, "Dynamic Modeling and Simulation of a Four-Wheel Skid-Steer Mobile Robot Using Linear Graphs," Electronics, vol. 11, no. 15, p. 2453, Aug. 2022.

[7] Maalouf, E., Saad, M., Saliah, H. (2006). "A higher level path tracking controller for a four-wheel differentially steered mobile robot." Robotics and Autonomous Systems, 54(1), 23–33.

[8] Mandow, A., Martínez, J. L., Morales, J., Blanco, J. L., García-Cerezo, A., Gonzalez, J. (2007) Experimental kinematics for wheeled skid-steer mobile robots. In 2007 IEEE/RSJ international confer- ence on intelligent robots and systems (pp. 1222–1227).

[9] Pentzer, J., Brennan, S., Reichard, K. (2014). Model-based prediction of skid-steer robot kinematics using online estimation of track instantaneous centers of rotation. Journal of Field Robotics, 31(3), 455–476.

[10] K. Steadman, C. Stubbs, A. Baskaran, C. G. Rose, D. Bevly, "Teleoperated Ground Vehicle Rollover Prevention via Haptic Feedback of the Zero-Moment Point Index," In Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium (GVSETS), NDIA, Novi, MI, Aug. 16-18, 2022.

[11] H. Luo, Z. Chen, A. Naveen, B. Li, "Dynamic Modeling and Prediction of Rollover Stability for All-Terrain Vehicles", In Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium (GVSETS), NDIA, Novi, MI, Aug. 11-13, 2020.

[12] NaturalPoint, "Motion Capture Systems - OptiTrack Webpage." [Online]. Available: optitrack.com. [Accessed: Nov-2022].

[13] Y. Kouskoulas, A. Schmidt, J. Jeannin, D. Genin and J. Lopez, "Provably Safe Controller Synthesis Using Safety Proofs as Building Blocks," 2019 7th International Conference in Software Engineering Research and Innovation (CONISOFT), Mexico City, Mexico, pp. 26-35, 2019.

[14] P. Frederick, M. Del Rose, and K. Cheok, "Autonomous Vehicle Safety Reasoning Utilizing Anticipatory Theory," In Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium (GVSETS), NDIA, Novi, MI, 2019.

[15] D. Genin, E. Dietrich, Y. Kouskoulas, A. Schmidt, M. Kobilarov, K. Katyal, S. Sefati, S. Mishra, and I. Papusha, "A Safety Fallback Controller for Improved Collision Avoidance," The 2nd International Conference on Assured Autonomy (ICAA), Laurel, MD, 2023.