Control Policy Design for Path Planning and Obstacle Avoidance Using Optimal Trajectory Generation of a Four-Wheel Based Extraterrestrial Rover

Jason Zhou{zzhou292@wisc.edu}, Yulong Yue{yyue32@wisc.edu}

May 8, 2023

1 Research Problem Statement

In this project, we proposed to design a rover path planning and following algorithm to provide an extraterrestrial rover navigation strategy. We propose to solve the problem by solving a closed-loop motion planning and control problem based on optimization with constraints. The control strategy will be benched in a simulated environment in Project Chrono. For the project, we will assume privileged information, such as locations, sizes, and shapes of the obstacles. Also, we will assume the environment is limited and fixed. The problem can be extended in the future to include sensing capabilities.

2 Background

2.1 Rover Control Strategy

Providing control solutions to extraterrestrial rover can always be challenging, due to the low tolerance to mistake and long signal transmission delay. Nowadays, most space exploration agencies, such as The National Aeronautics and Space Administration (NASA), are adopting two different ways for rover control:

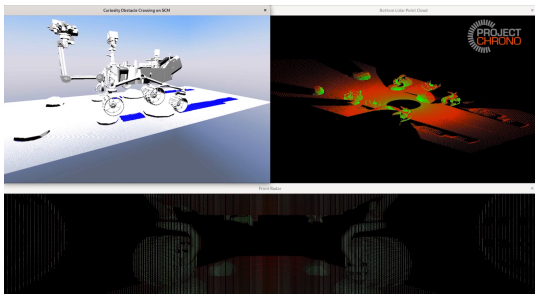
- Manual Explicit Motion Commands
- A propitiatory Autonomous Path Planning and Path Following Control Policies

The first manual control strategy is relatively simple. Based on the images transmitted from the rover, the ground base provides direct motion commands, such as "move forward for 5m", to the rover. The method relies heavily on human judgment, along with data and command transmission, which usually suffer from the signal delay between the ground base on earth and extra extraterrestrial planets. This control method will not be discussed in the project.

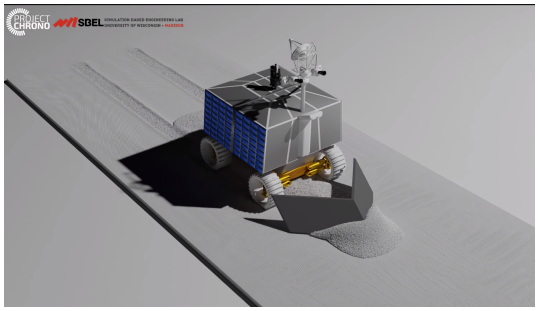
The second control method, although proprietary and few details have been disclosed by NASA, is an autonomous path planning and trajectory following control algorithm which matches the materials covered in the class. Based on the introduction provided by NASA, the ground base on earth, based on the images and data transmitted from the rover, sends a control signal containing the location of the next 'waypoint'. Then the rover, based on its own sensing capabilities using sensors such as camera and lidar, plans the route and tracking the planned trajectory. [1]

2.2 Simulation Through Project Chrono

Project Chrono is a multi-physics modeling and simulation infrastructure based on a platform-independent, open-source design. Project Chrono has been used widely for the simulation of autonomous vehicles and robotic systems. Project Chrono, in the past two years, received funding from NASA to provide simulation support to investigate wheel-terrain interaction of the 2024 NASA VIPER Moon Rover using different types of terra-mechanics simulation method, as shown in Figure 1(b) and Figure 1(a). Therefore, the software package already contains multiple ready-to-deploy extraterrestrial rover models and their control interfaces.



(a) Curiosiy Mars rover on SCM deformable terrain with lidar and radar sensing simulation



(b) VIPER Lunar rover conducting bulldozing experiment on granular terrain simulated by Smoothed-Particle Hydrodynamics (SPH) Method

Figure 1: Project Chrono contains complete rover models, control interfaces, and deformable terrain models for wheel-terrain interaction

In this project, we will develop the algorithm in Matlab, and use Project Chrono as our simulation environment. In the simulation, we will use the Soil-Contact Model (SCM) deformable terrain model to simulation soil deformation. The SCM model is a simple semi-empirical model which is built upon the original Bekker-Wong Equation for normal force computation, as shown in Equation 1, and Janosi-Hanamoto equation for tangential stress computation. A more detailed description can be found in [2]. In the Equation 1, K_c, K_ϕ, and n are semi-empirical parameters defining the soil properties, z is the sinkage, b is the average wheel contact width, and p is the normal pressure applied to the rover wheel (to provide normal force in the simulation). Note that the SCM is only included for high-fidelity terrain simulation purposes, but not included in the optimal path planning.

$$p = \left(\frac{K_c}{b} + K_\phi \right) z^n \,. \tag{1}$$

$$\begin{cases} t = t_{max}(1 - exp^{-\frac{j}{k}}) \\ t_{max} = c + ptan(\phi) \end{cases} . \tag{2}$$

Note that in Equation 1, p is the mean probe contact pressure, b is the width of the footprint, z is the sinkage of the footprint, and $K_{\phi]}$, K_c, and n are semi-empirical soil parameters based on curve-fitting. In Equation 2, j is the accumulated shear, c represents cohesion, ϕ represents internal friction angle based on Mohr Theory, and k is the Janosi parameter.

The Viper rover model designed in Project Project Chrono consists of an active suspension system, 4 independently controlled DC motors [3] and 4 independently controlled steering motors. Within the scope of the current project, no suspension manipulation and control is used and the steering is achieved by a 4-wheel steering mechanism.

3 Proposed Solution

The problem is dissected into two parts - trajectory generation and trajectory tracking in simulation. The first part, trajectory generation, is conducted using Imperial College London Optimal Control Software (ICLOCS 2), An Optimization Based Control package in Matlab/Simulink [4]. The optimization trajectory generation problem simplifies the problem by reducing it to 2-D and employing simplified rover dynamics, while representing obstacles as bounding circles. After trajectory has been generated, the planned path is parsed into a .csv file consisting of 200 way points.

In the second stage, known as trajectory tracking in simulation, the Project Chrono platform is utilized. The rover's complete dynamics, encompassing various components such as the DC motors, chassis, steering mechanism, and suspension system, are accurately modeled. The simulation is structured as a multi-body dynamic problem that incorporates rigid bodies and constraints between different bodies. The terrain is simulated through a semi-empirical deformable model called the Soil-Contact Model (SCM), which is explained in detail in Section 2.2. The output trajectory obtained

from the previous stage is then reconstructed into a Bézier curve, which the rover aims to pursue in the simulation. For the current problem, a PID controller is used for trajectory tracking.

3.1 Trajectory Generation

The problem is simplified into an optimal path planning and control optimization problem by assuming a square terrain with a center at (0.0, 0.0), and a fixed side length of 20 meters. Three different types of rocks with similar radius, while viewing birdeye view, are used. To further simplify the problem, a bounding circle, or what can be called a safety circle, is drawn around the rock. From a bird's eye view, looking from above, the problem environment is simplified into a 2D path planning and optimal control problem with a given square terrain, and with several circular-shaped obstacles, as shown in Figure 3.1.

We employ a dynamic model of a simplified rover as shown in Equations 3. This model is a 2-wheel steering car model migrated and modified from an example problem presented on ICLOCS website [5]. Due to the simplification nature of this dynamic model, the sim-2-real gap needs to be recognized.

$$\begin{cases} \dot{x} = v(t)cos(\theta(t)) \\ \dot{y} = v(t)sin(\theta(t)) \\ \dot{v}(t) = a(t) \\ \dot{a}(t) = u_1(t) \\ \dot{\theta}(t) = \frac{v(t)tan(\phi(t))}{l_a} \\ \dot{\phi} = u_2(t) \end{cases} . \tag{3}$$

In Equation 3, $x(t)$ and $y(t)$ are the rover position at time t, $v(t)$ is the rover velocity at time t, and $a(t)$ is the rover acceleration at time t. $\theta(t)$ is the rover yaw angle in the global frame, and $\phi(t)$ is the steering angle of the rover. The two inputs, $u1(t)$ and $u2(t)$, control the acceleration and the rate of steering angle change respectively.

After the simplification, we assume that each obstacle has the properties shown in Equation 4, in which x_{obs_k} and y_{obs_k} represent the location of the k'th obstacle's bounding circle, and r_{obs_k} represent the radius of the bounding circle of the k'th obstacle. Note that since in the algorithm the geometry of the rover chassis is not taken into consideration, the decision of the bounding circle's radius needs to be further expanded by at least rover's half width. If we assume the entire time domain of the problem is t_f, and there are M obstacles presented in the environment. Assuming the starting location of the rover is [x_{start}, y_{start}], the given target location is [x_{end}, y_{end}], the initial rover yaw angle is yaw_{t_0}, and the initial rover steering angle 0. We formulate the initial condition, terminal condition, and path constraint for the optimal control problem:

$$\begin{cases} x(0) = x_{start}, y(0) = y_{start} \\ x(t_f) = x_{end}, y(t_f) = y_{end} \\ \theta(0) = yaw_{t_0} \\ \phi(0) = 0 \\ r(k) < (x(t) - x_{obs_k})^2 + (y(t) - y_{obs_k})^2 \\ 0 \leq t \leq t_f, 0 \leq k \leq M \end{cases} . \tag{4}$$

In addition to the above path constraints, the vehicle dynamic and state constraints are formulated as below:

$$\begin{cases} -10 \leq x(t) \leq 10, -10 \leq y(t) \leq 10 \\ 0 \leq v(t) \leq 3.0 \\ 0 \leq a(t) \leq 0.8 \\ -\frac{\pi}{6} \leq \phi \leq \frac{\pi}{6} \end{cases} . \tag{5}$$

Where the position states, $x(t)$ and $y(t)$, are bounded between -10 and 10, the square terrain which we pre-defined. The magnitude of the velocity of the rover is bounded between 0.0 m/s and 3.0 m/s, and the magnitude of the acceleration of the rover is bounded between 0.0 $\frac{m}{s^2}$ and 0.8 $\frac{m}{s^2}$. Note that the constraints for the velocity magnitude and acceleration magnitude implies that the path generated from the optimization problem considers forward movement only, and no reverse motion is considered. The max steering, ϕ, is bounded between -30 deg to 30 deg, or $-\frac{\pi}{6}$ and $\frac{\pi}{6}$.

An extra constraint is added to ensure on-board equipment's stability:

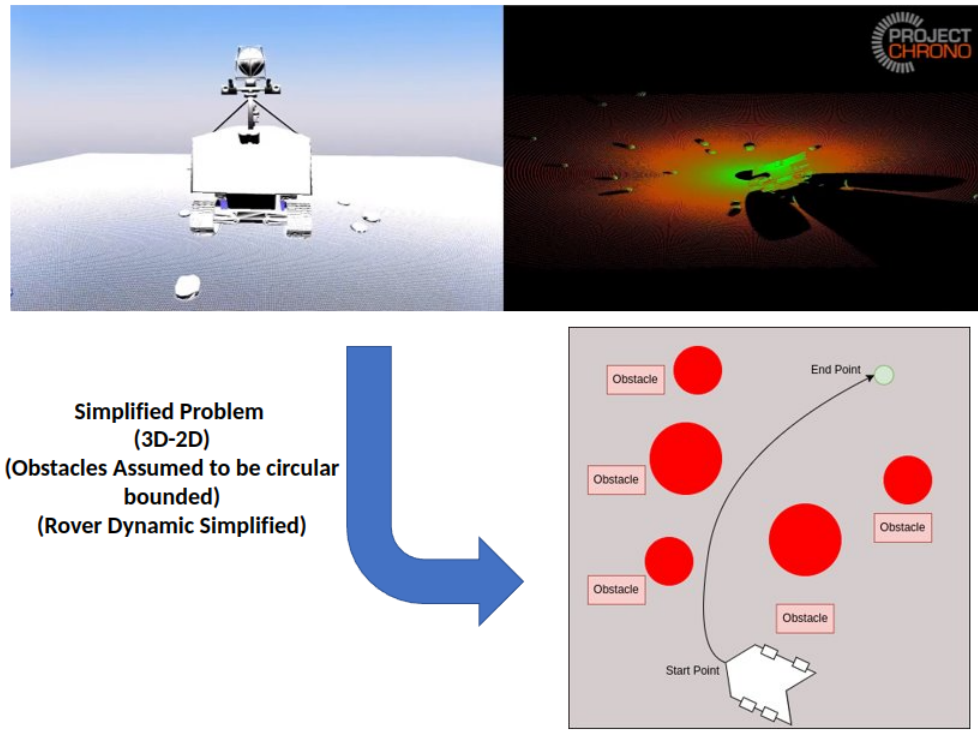


Figure 2: The problem will be simplified into a 2-dimensional path planning and optimal control problem. The optimization will be solved in the 2-D space and applied and benched in the 3-D simulation.

$$-0.4 \leq \frac{u_2(t)}{l_{axes} cos^2(\theta(t))} \leq 0.4 \tag{6}$$

In the above equation, the l_{axes} represents the distance between two axes of the rover. For the Viper lunar rover which is modeled in this project, the $l_{axes} = 1.4m$. In addition to the above constrains defined, a maximum final time is defined, as below:

$$0 \leq t_f \leq 100 \tag{7}$$

The objective is for rover to finish the path in minimum time, as a results, the optimal control problem is formulated as:

$$\min_{x,u,t_f} t_f \tag{8}$$

3.2 Trajectory Tracking in Simulation

After the optimization problem has been solved, a control input series u_{opt} and a control path series defined by x_{opt} and $yopt$ are expected to be the output. However, the control input series u_{opt} will need to be post-processed in order to feed into the Chrono simulation. This is due to the fact that the simplified rover dynamic equations defined in Equation 3 is a significantly simplified control, and the environment and the obstacle modeling are simplified as well. However, the actual rover control and rover dynamic response, using a high-fidelity rover dynamic modeling and simulation, depend on multiple factors, such as wheel-terrain interaction, DC motor modeling, steering and suspension mechanisms and etc.

Therefore, only the optimal path obtained from the first part, trajectory generation, is used, and the second part of the problem is formulated as trajectory tracking. The path obtained from the trajectory generation consists of 200 way points, which are reconstructed into a Bézier curve for easier evaluation and tracking. A simple PID controller is used for tracking, as shown below:

	Initial Rover Position	Initial Rover Yaw Angle	Final Rover Position
Scenario 1	[-5.0, -6.0]	60 deg	[0.0, 8.0]
Scenario 2	[-6.0, -7.0]	110 deg	[8.0, 4.8]
Scenario 3	[-6.0, -7.0]	110 deg	[0.0, 8.0]
Scenario 4	[-5.0, 4.0]	30 deg	[9.0, -8.0]

Table 1: Rover's initial position, final position, and initial yaw angle for each scenario.

	Obs 1	Obs 2	Obs 3	Obs 4	Obs 5
Scenario 1	[2.0, 0.0]	[-4.0, -1.0]	[0.0, 1.2]	[3.0, 5.0]	[-3.0, 2.0]
Scenario 2	[3.0, -2.0]	[-3.0, 1.0]	[0.0, -0.2]	[8.0, -2.5]	[-3.0, -2.0]
Scenario 3	[5.0, -1.4]	[0.2, 4.8]	[0.0, 0.0]	[6.4, 4.5]	[-3.8, -2.0]
Scenario 4	[3.0, -3.4]	[-5.2, -7.8]	[0.0, 0.8]	[7.4, -1.3]	[-6.8, 2.0]
	Obs 6	**Obs 7**	**Obs 8**	**Obs 9**	**Obs 10**
Scenario 1	[7.0, 1.0]	[4.0, -7.0]	[-8.0, 5.0]	[7.6, -4.0]	[-8.1, -2.3]
Scenario 2	[5.0, 3.0]	[3.2, 4.0]	[-8.0, 7.0]	[5.8, 6.7]	[-7.2, -2.3]
Scenario 3	[3.5, 3.0]	[-2.8, 4.0]	[-8.0, 8.0]	[2.8, -3.7]	[-7.2, 2.3]
Scenario 4	[1.9, -7.1]	[6.8, 4.0]	[6.9, 7.0]	[-3.8, 0.0]	[-7.2, 6.3]

Table 2: Obstacle setup for each scenario.

$$u(t) = K_p e(t) + Ki \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{9}$$

The error term $e(t)$, in Equation 9, is defined as below:

$$\begin{cases} s(t) = A_{local}^{global}(t)(m_{dist} * [1.0, 0.0, 0.0]^T) \\ t(t) = eval(s(t)) \\ \hat{y}(t) = A_{local}^{global}(t)([0.0, 1.0, 0.0]^T) \\ e(t) = (s(t) - t(t)) * (\hat{y}(t)) \\ e(t) = clamp(e(t), -1.0, 1.0) \end{cases} . \tag{10}$$

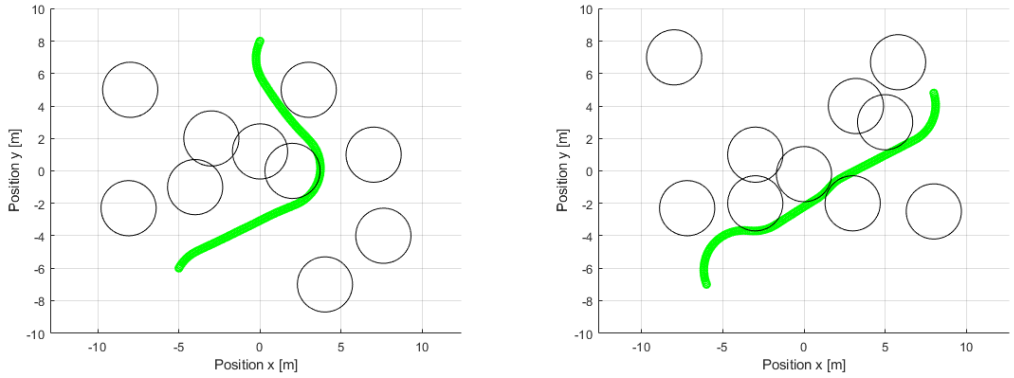
Equation 10 involves several variables. Firstly, m_{dist} denotes the lookahead distance, while $s(t)$ represents the sentinel point. Secondly, $t(t)$ refers to the target point on the Bézier curve that is in proximity to the sentinel point $s(t)$. Lastly, $\hat{y}(t)$ is a unit vector oriented towards the rover's lateral direction. Notably, the error term $e(t)$ is limited between -1 and 1, signifying the maximum left and right steering errors, respectively. Note that since in the current problem setup, the rover is moving relatively slow using a linear DC motor model with maximum rotational speed π rad/s and a stall torque of 300 N/m. By conducting serval experiments, it is believed that a positive K value is good enough for trajectory tracking to work relatively well.

4 Results

4 different obstacle arrangement has been tested and examined to benchmark the path planning and trajectory tracking's robustness. The positions of the obstacles, and the scenario parameters for each testing case are shown in Table 1 and Table 2. Note that 3 rock geometries are used and they have similar geometry, within the scope of the current project. Therefore, for simplification, a 1.5-m radius is used for all obstacle geometries.

The path planning results from ICLOCS are shown in Figure 3. The screenshots, and full simulation animation rendered by chrono::irrlicht submodule, can be found in Figure 4. The comparison between target trajectory and the actual trajectory by using a PID tracking control is shown in Figure 5. Figure 6 shows the rover's speed profile and steering angle input in the simulation for each scenario.

The simulation results show that no collision happens and the trajectory planning and tracking work well in all 4 scenarios. Parameter tunning might be performed to further improve the PID tracking control's performance.



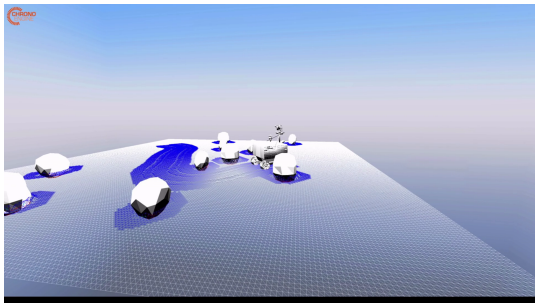
(a) Scenario 1

(b) Scenario 2

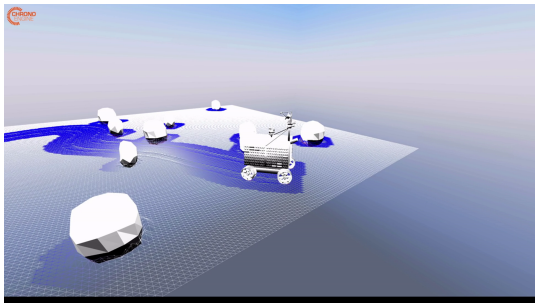
(c) Scenario 3

(d) Scenario 4

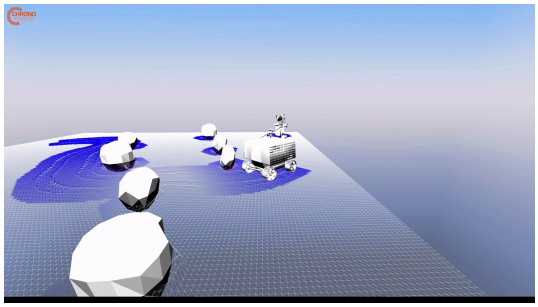
Figure 3: Trajectories generated by ICLOCS. In the each figure, the green curve is the optimal trajectory and each circle represents the bounding circle of each obstacle.



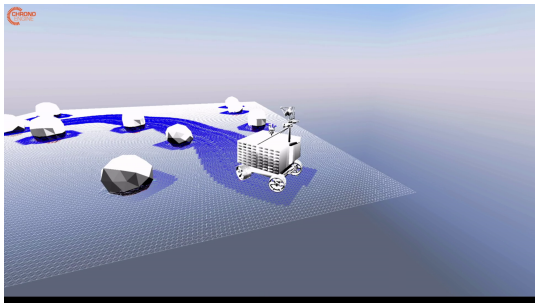
(a) Scenario 1 [6]



(b) Scenario 2 [7]



(c) Scenario 3 [8]



(d) Scenario 4 [9]

Figure 4: Screenshot of the simulation for each scenario. Full simulation animation can be found in the corresponding citation entry.

5 Limitation and Future Work

The current control algorithm assumes privileged information regarding terrain size, obstacle locations, and rover location, which is limiting due to time constraints. However, future research aims to replicate real-life engineering scenarios by acquiring information through sensors such as camera, lidar, radar, and position sensors. Project Chrono has the capability to simulate sensor data, as demonstrated in Figure 1(a), but there is a need to reconstruct the 3D environment and obtain terrain and obstacle data from the real environment. Therefore, there are plans to conduct research to enable this in the near future.

It is essential to validate the control algorithms' robustness and reliability by benchmarking them against more complex scenarios. This requires testing them against a wider range of obstacle sizes and locations. The rationale behind this is that complex scenarios offer more challenging environments for the control algorithms, which helps to identify areas of improvement. Moreover, these tests can provide valuable insights into how the algorithms perform under different conditions and highlight any potential weaknesses or failure points. Therefore, it is important to perform a thorough investigation of the algorithms' performance under various conditions to ensure their reliability and effectiveness in real-world situations.

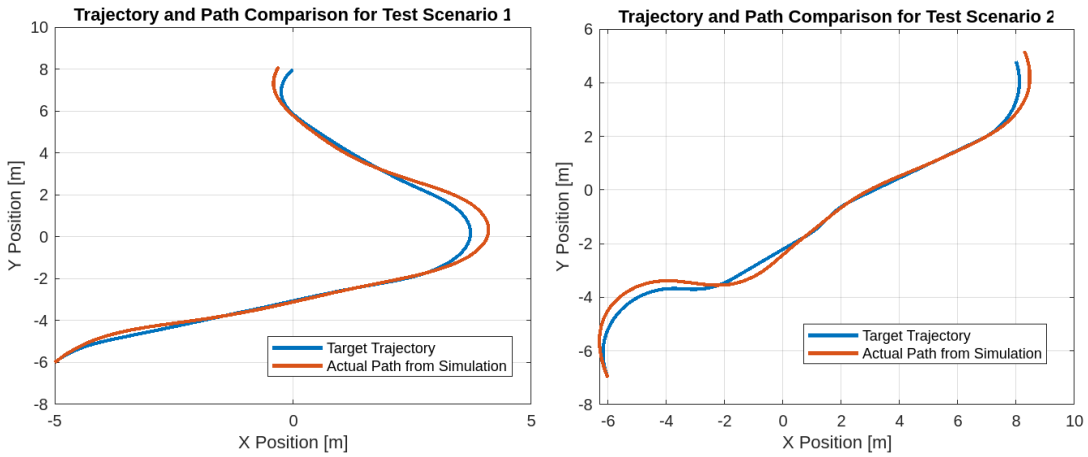
References

[1] K. Rainey, "Station astronauts remotely control planetary rover from space," 2013.

[2] R. Serban, J. Taves, and J. Zhou, "Real-time Simulation of Ground Vehicles On Deformable Terrain," *Journal of Computational and Nonlinear Dynamics*, pp. 1–10, 02 2023.

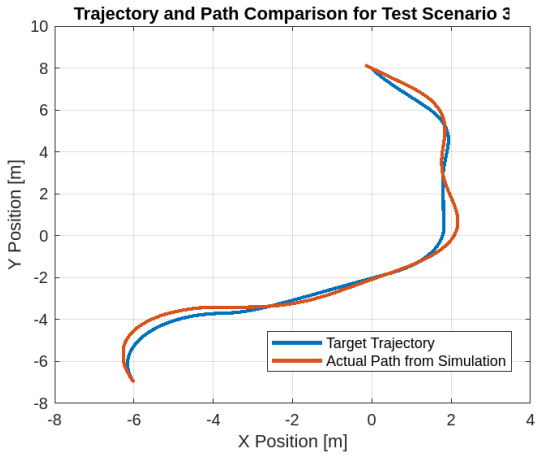
[3] Z. Zhou, W. Hu, R. Serban, and D. Negrut, "Modeling linear dc motor in chrono rover models," tech. rep., Simulation-Based Engineering Laboratory, University of Wisconsin-Madison, 2021.

[4] Y. Nie, O. Faqir, and E. Kerrigan, "Iclocs2: Try this optimal control problem solver before you try the rest," pp. 336–336, 09 2018.

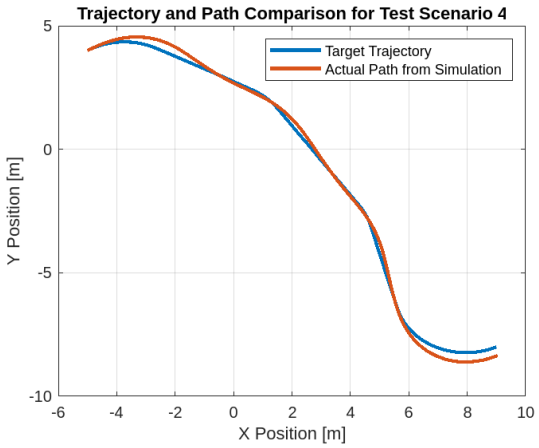


(a) Scenario 1

(b) Scenario 2

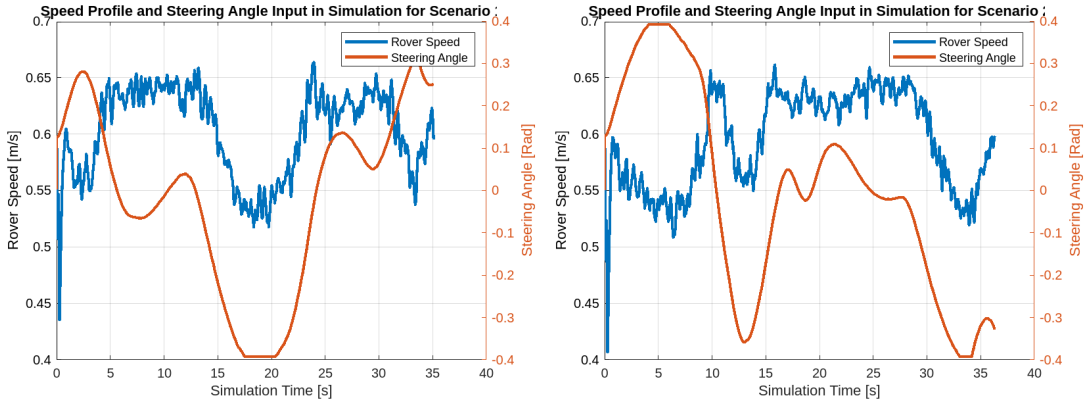


(c) Scenario 3



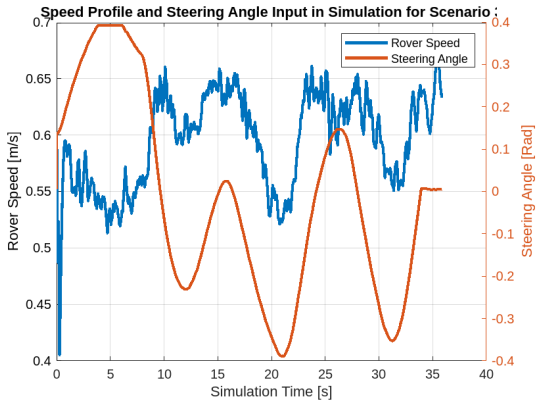
(d) Scenario 4

Figure 5: Comparison between target trajectory and the actual simulation path for each scenario. Note that the trajectory tracking in enforced by using a PID controller.

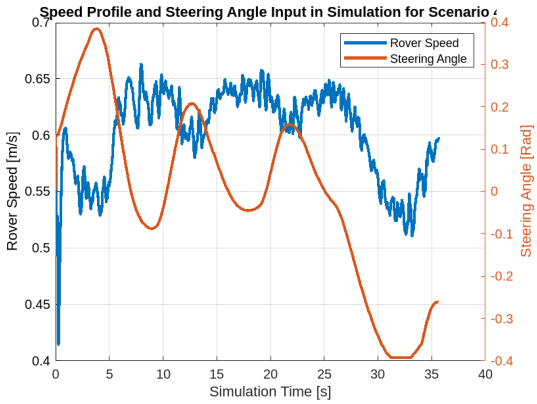


(a) Scenario 1

(b) Scenario 2



(c) Scenario 3



(d) Scenario 4

Figure 6: Rover's speed profile and steering angle input in the simulation for each scenario.

[5] ICLOC2, "Example: parallel parking." http://www.ee.ic.ac.uk/ICLOCS/ExampleParallelParking.html, 2023.

[6] Z. Zhou, "Viper Rover optimal path planning and obstacle avoidance using iclocs - scenario 1." Simulation-Based Engineering Laboratory, University of Wisconsin-Madison, https://uwmadison.box.com/s/x5ygp8wax9ljo5iwoknrqyok4mpf8b5a, 2023.

[7] Z. Zhou, "Viper Rover optimal path planning and obstacle avoidance using iclocs - scenario 2." Simulation-Based Engineering Laboratory, University of Wisconsin-Madison, https://uwmadison.box.com/s/pyl3hq6nhgo4vosjjv57co34zbe4jvc3, 2023.

[8] Z. Zhou, "Viper Rover optimal path planning and obstacle avoidance using iclocs - scenario 3." Simulation-Based Engineering Laboratory, University of Wisconsin-Madison, https://uwmadison.box.com/s/llk4madfc9sowhn9oldv1zqfnjqpw9fz, 2023.

[9] Z. Zhou, "Viper Rover optimal path planning and obstacle avoidance using iclocs - scenario 4." Simulation-Based Engineering Laboratory, University of Wisconsin-Madison, https://uwmadison.box.com/s/r1l5kp7lqrrn9gi4ig588zdvndr2jx0b, 2023.