A Nonlinear Model Predictive Control Algorithm for an Unmanned Ground Vehicle on Variable Terrain

by

Andrew Eick

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
December 10, 2016

Keywords: Model Predictive Control, Nonlinear Model Predictive Control, Unmanned Ground Vehicle

Copyright 2016 by Andrew Eick

Approved by

David Bevly, Professor of Mechanical Engineering
Peter Jones, Professor of Mechanical Engineering
John Hung, Professor of Electrical and Computer Engineering

Abstract

This thesis presents a Nonlinear Model Predictive Controller (NMPC) for an Unmanned Ground Vehicle (UGV) capable of controlling the vehicle over both smooth and rough terrain using measurements from GPS and a Light Detection And Ranging (LiDAR) unit equipped to the vehicle. Linear Model Predictive Control (MPC) and NMPC have become more widely used to control dynamic systems as computers have become more capable of handling the computational expense required by model predictive control. Though the use of NMPC rather than linear MPC creates an additional computational expense, NMPC allows for path planning in addition to control of the vehicle. This is particularly advantageous in scenarios in which the UGV is traversing terrain that contains obstacles of which the vehicle has no *a priori* knowledge.

Rough, off-road terrain contains multiple hazards for an UGV. In this thesis, hazards are classified into three groups: obstacles, rough traversable terrain, and rough untraversable terrain. These three types of hazards create a rollover risk for a UGV. The NMPC presented in this thesis is designed to mitigate this risk of rollover. Simulations of the NMPC in several different scenarios are presented, as well as results from experimental implementation of the NMPC on a test vehicle. Results from simulation and experimental implementation are provided that show the NMPC is able to navigate a UGV around obstacles to a target location without requiring the use of *a priori* knowledge of terrain and obstacles.

Acknowledgments

I'd like to thank several people for supporting me while working on my Master's Degree at Auburn. Firstly, I'd like to thank my parents for their encouragement, and my mom in particular for all the tasty food she brought to the lab.

Next, I'd like to thank my advisor, Dr. Bevly for giving me the opportunity to work in his lab. I've been able to learn so much in the past three years. I feel like I now "know enough to be dangerous". Dr. Bevly has always made sure that his students are well taken care of, and working for him these past few years has been a lot of fun.

Lastly, I'd like to thank all my labmates who have helped me with this research. In particular, I'd like to thank Dan Pierce for getting me this job. I'm glad I decided to be in your GPS group Dan.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction and Background

Unmanned ground vehicles (UGVs) allow transportation of items and/or persons in situations where a driver is not available or unable to maintain his/her attention on the task of driving the vehicle. A UGV is able to transport goods to people through areas that are too dangerous to allow human drivers to traverse, and can deliver people from one area to another without requiring the people being transported to be encumbered by the task of driving. In many scenarios that require the use of a UGV, the UGV is unable to rely on *a priori* knowledge of terrain and obstacle locations in order to navigate, and requires a perception-based sensor to locate hazards and plan a path around them. Nonlinear model predictive control is able to perform path planning while implementing both hard and soft constraints on the path taken by the vehicle. These constraints keep the vehicle from driving into unsafe scenarios that could result in rollover.

1.1 Overview of MPC

Model predictive control (MPC) has been used in industry since the 1980's. Originally MPC was only used for controlling processes in chemical plants and oil refineries, because computers at the time were unable to handle the large computational expense required by applying MPC to high dynamic systems. However, as computers have become more powerful in recent years, MPC has become a popular control method for multiple types of dynamical systems.

Model predictive control works by determining a series of control inputs subject to a model over a set amount of time. This length of time is called the control horizon. Once the series of inputs over the control horizon is determined, only the first input of that series is applied, and the process is repeated. Model predictive controllers, particularly nonlinear model predictive controllers, are known to suffer from a high computational expense.

1.2 Prior Research

The goal of this thesis is to provide a NMPC algorithm that is able to navigate a UGV over rough terrain using GPS and a perception-based sensor. Many authors have developed MPC and NMPC algorithms to control or aid in control of ground vehicles. Reviews of the research work found prior to assembling this thesis can be found in the following subsections.

1.2.1 Model Predictive Control for Semi-Autonomous Vehicles

Because MPC allows for easy implementation of constraints upon a system, many active vehicle safety features use model predictive control. One such safety feature is a direct yaw moment control system presented in [1]. In the work presented in [1], Lei presents an MPC coupled with a tire cornering stiffness estimator to better predict yaw rate available to the vehicle. Another vehicle safety feature that uses model predictive control is a lane-keeping controller shown in [2]. In [2] an NMPC is presented that uses the linear bicycle model in conjunction with a nonlinear tire model, much like the work presented in this thesis.

Nonlinear model predictive control can be advantageous, as it allows for the capture of nonlinear dynamics within a system. However, NMPC algorithms typically suffer from high computational expense, which can be prohibitive when trying to implement a controller in real time. In an effort to more accurately capture the dynamics of a UGV without the unwanted computational burden, an MPC strategy is presented in [3] and [4] that uses successive linearizations of a nonlinear tire model, which yields a linear time varying tire model. This strategy reduces run time and more accurately captures the dynamics of the vehicle when the tires of the vehicle are just outside the linear region while still maintaining a linear model. However, when the tires of the vehicle are operating well outside the linear region, this linearization breaks down.

1.2.2 Path Planning and Obstacle Avoidance

As previously mentioned, NMPC confers the added advantage of being able to path plan without *a priori* knowledge of terrain. In order to safely traverse terrain in this manner, an obstacle avoidance algorithm must be coupled with the NMPC. Several obstacle avoidance algorithms for ground vehicles exist in literature [5] - [9]. In these references, a constant longitudinal velocity of the vehicle is assumed, and a soft constraint is placed on obstacle avoidance. Assuming a constant longitudinal velocity is not ideal for off-road operation because the vehicle is likely to encounter scenarios in which the vehicle will need to change velocities in order to safely navigate. Also, a hard constraint on obstacle avoidance is preferable to a soft constraint for several reasons. With only a soft constraint on obstacle avoidance, the NMPC penalizes paths close to an obstacle, but does not guarentee to avoid the obstacle. Furthermore, the use of a hard constraint instead of a soft constraint, allows paths that are close to an obstacle but still a safe distance to not be unnecessarily penalized. In addition, with the hard constraint there is no obstacle avoidance parameter within the cost function to be tuned. The reason for the assumption of constant velocity and the use of a soft constraint on obstacle avoidance lies in the minimization method used by the NMPC. References [5], [7], and [8] use a gradient based method for cost function minimization, and [9] uses an interior point based method. These methods rely on having non-zero values for the partial derivative of the cost and constraint functions with respect to the control input. Because the equations of motion that govern the position of the vehicle relative to an obstacle do not directly contain any control input variables, a hard constraint cannot be applied in this manner. Because a hard constraint on obstacle avoidance and the ability to incorporate longitudinal velocity into the control input vector are desirable, the work presented in this thesis uses a different type of minimization technique called a pattern search. The pattern search will be described in detain in Chapter 3.

1.3 Contributions

This thesis presents an NMPC algorithm capable of safely navigating a UGV to a target location over rough terrain using GPS and a LiDAR unit. The NMPC presented performs path planning, taking into account knowledge of the terrain the UGV is traversing provided by a LiDAR, and does not rely on a flat or smooth terrain assumption as in [5] - [7] and [9]. Simulations of the performance of the NMPC are provided and these simulation results are validated with experimental data. Simulation and experimental results are analyzed and the benefits of NMPC versus linear MPC are discussed. In summary, this thesis provides the following contributions to the field:

- Development of a control algorithm for a UGV on rough terrain that is capable of both path planning and control
- NMPC algorithm features a hard constraint on obstacle avoidance
- NMPC algorithm presented has low enough computational burden for real time implementation
- Simulations of NMPC performance are validated with experimental data

Chapter 2

Vehicle Model

This chapter will outline various vehicle and tire models used for vehicle navigation and control, including the models used by the NMPC algorithm described by this thesis.

2.1 Planar Dynamic Model

The most prevalent model used to describe the planar dynamics of a vehicle is the bicycle model. There are two common variations of this model: the kinematic model and the lateral model [10]. These models are shown in Figures 2.1 and 2.2. In Figure 2.1, δ is the steer angle, β is the sideslip, r is the yaw rate, L is the wheel base length, and V_x and V_y are the longitudinal and lateral velocities respectively. In Figure 2.2, the wheel base length is split up into front wheel base length, a, and rear wheel base length, b. The simpler of the two bicycle models, the kinematic model, assumes zero lateral tire velocity i.e. the slip angles at each tire are zero. This assumption is invalid when the vehicle is traveling at high speeds or is turning [11]. The lateral model does not rely on this assumption of zero slip, and incorporates the front and rear tire slip angles shown by α_f and α_r, respectively. The calculation of the front and rear slip angles is shown in Eqs. (2.1 - 2.2) below.

$$\alpha_f = \delta - \tan^{-1}((V_y + ar)/V_x) \tag{2.1}$$

$$\alpha_r = \tan^{-1}((-V_y + ar)/V_x) \tag{2.2}$$

The front and rear tire slip angles are then mapped to a front and rear lateral tire force, F_{yf} and F_{yr}, respectively. This mapping is subject to a tire model, which is discussed in a later section. Because the UGV will likely experience tire slip throughout its operation, the lateral bicycle model was used instead of the simpler kinematic model.

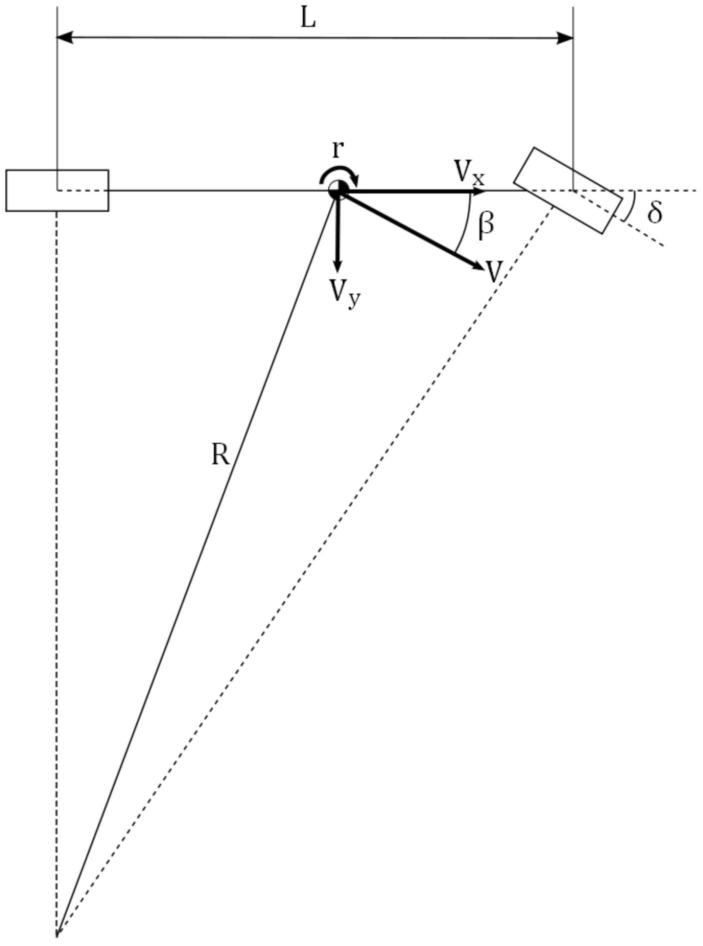

Figure 2.1: Kinematic Bicycle Model

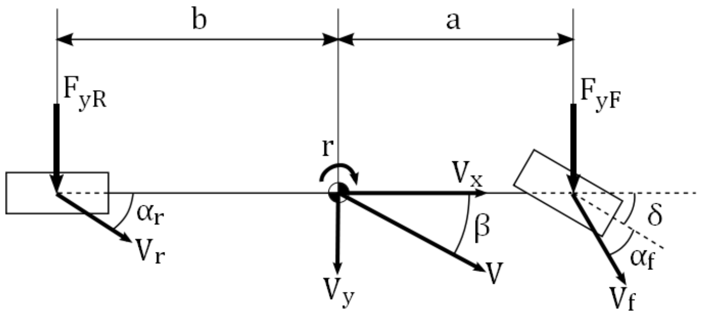

Figure 2.2: Lateral Bicycle Model

Once the lateral tire forces are determined, the heading, lateral velocity, and x and y positions in the local vehicle frame (ψ, V_y, and x and y, respectively) can be calculated. These calculations are described by Eqs. (2.3 - 2.6).

$$\ddot{\psi} = ((a * F_{yf}\cos(\delta)) - (b * F_{yr}))/I_{zz} \tag{2.3}$$

$$\dot{V}_y = (F_{yf}\cos(\delta) + F_{yr})/m - (V_x\dot{\psi}) \tag{2.4}$$

$$\dot{x} = V_x\cos(\psi) - V_y\sin(\psi) \tag{2.5}$$

$$\dot{y} = V_x\sin(\psi) + V_y\cos(\psi) \tag{2.6}$$

Where the yaw inertia of the vehicle is I_{zz} and m is the vehicle mass.

2.2 Roll Dynamic Model

There are two different types of dynamic roll models prevalent in literature: the full car model [10], and the curvilinear trajectory roll model [12] shown in Figures 2.3 and 2.4 respectively, where ϕ is the roll of the vehicle. Both roll models feature experimentally determined stiffness and damping coefficients, shown by K and C respectively. The inputs to the full car model are the terrain height on the tires of the vehicle, shown as z_1, z_2, z_3, and z_4 in Figure 2.3. The full car model offers the benefit of characterizing roll produced by the impact of the terrain on the vehicle, whereas the curvilinear trajectory roll model only characterizes roll produced by steering. However, unless the terrain height inputs to the full car model are known to within centimeter level accuracy, an accurate value for roll cannot be determined. Because it is unreasonable to assume that the terrain height under each tire can be known to the degree of accuracy that is required by the full car model, the curvilinear trajectory roll model was used in this thesis. The roll motion described in the curvilinear trajectory roll model is shown in Equation (2.7).

$$\ddot{\phi} = -(C_f/I_{xx})\dot{\phi} - (K_\phi/I_{xx})\phi + (mV_x h_{cg}/I_{xx})\dot{\beta} + (mV_x h_{cg}/I_{xx})\dot{\psi} \tag{2.7}$$

Where the roll inertia of the vehicle is I_{xx}, and h_{cg} is the center of gravity height.

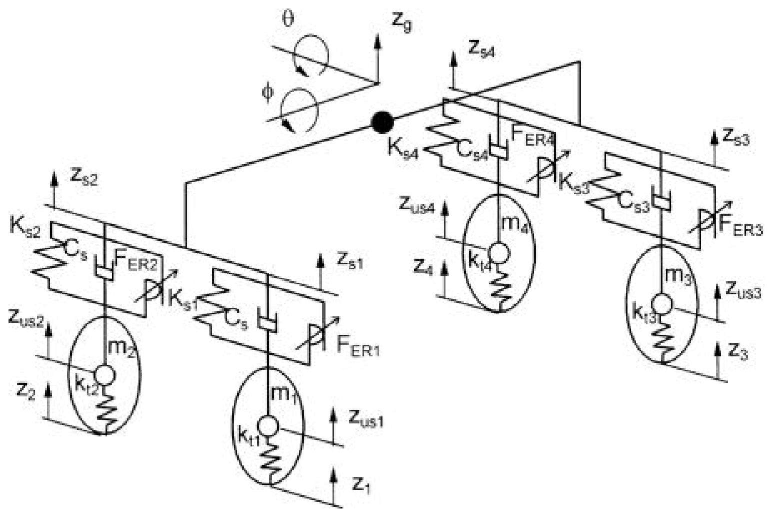

Figure 2.3: Full Car Model

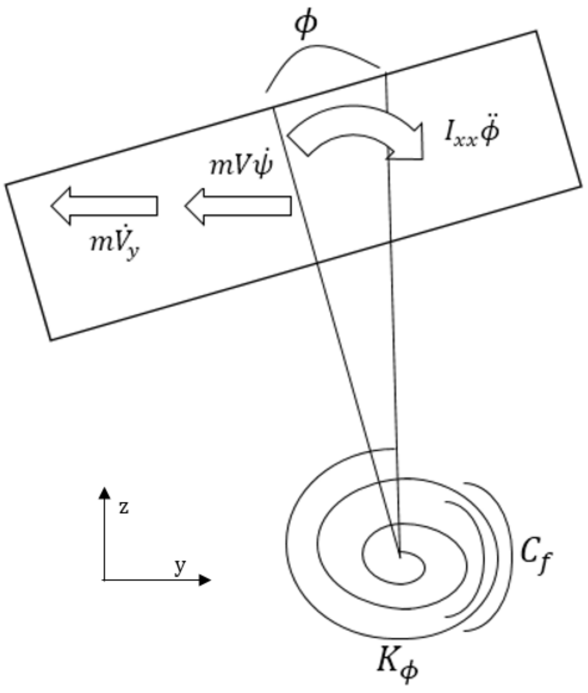

Figure 2.4: Curvilinear Trajectory Roll Model

2.3 Tire Model

As previously mentioned, the tire model maps slip angle, α, to a lateral tire force, F_y. The relationship between slip angle and lateral tire force is largely linear during normal vehicle operation, but the tire forces saturate under aggressive maneuvering. Because the UGV must operate over the full operational capability of the vehicle, and will at times exit the linear region of the tire curve, a nonlinear tire model is desired despite being more computationally burdensome than a linear model. The nonlinear model used is Pacejka's magic formula [13]. This model is shown in Figure 2.5, and is characterized by Eqs. (2.8 - 2.9).

$$F_{yf} = D * \sin(C * tan^{-1}(B * \alpha_f)) \tag{2.8}$$

$$F_{yr} = D * \sin(C * tan^{-1}(B * \alpha_r)) \tag{2.9}$$

Where B, C, and D are unitless shape parameters of the UGV's tires.

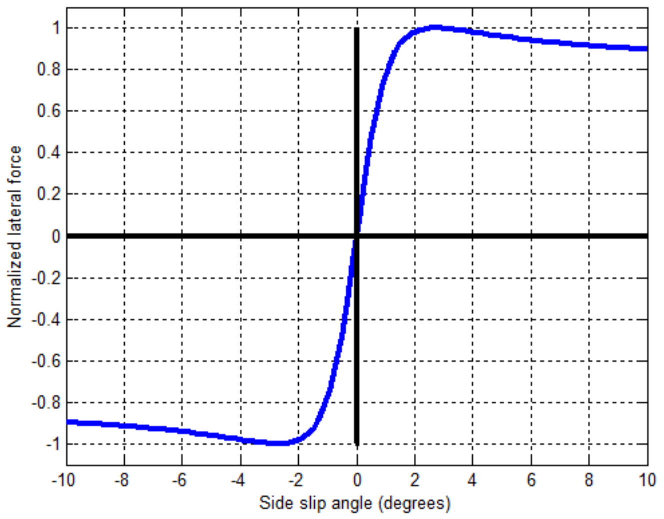

Figure 2.5: Pacejka Tire Model

Combining the lateral bicycle model, the Pacejka tire model, and the simpler roll model, a 7x1 vehicle state vector, $\mathbf{x}$, is derived and shown in Equation (2.10).

$$\mathbf{x} = [x, y, \phi, \dot{\phi}, \psi, \dot{\psi}, V_y]^T \tag{2.10}$$

The NMPC algorithm is developed using the above state vector, and the algorithm is described in detail in the next chapter.

Chapter 3

Nonlinear Model Predictive Control Algorithm

This chapter outlines the development of the Nonlinear Model Predictive Control (NMPC) algorithm using the nonlinear vehicle model presented in the previous chapter. The cost and constraint functions derived for the NMPC are detailed. These functions define optimality for the NMPC. The method employed to minimize the cost function subject to the constraint functions is then explored. In order to determine feasibility of real time implementation, run time of the NMPC is then evaluated.

3.1 Cost/Constraint Functions

In order to determine a suitable series of inputs over the control horizon, conditions for optimality must be defined by cost and constraint functions. Two cost functions were developed: one cost function that incorporates the predicted value of roll, and the other incorporates lateral acceleration instead. Both cost functions are used independently, and simulation results of each cost function are shown in the next chapter. The cost functions used by the NMPC are shown in Equations (3.1 - 3.2).

$$J = \int_0^t (\alpha\phi^2 + \beta\dot{\phi}^2 + \gamma d_t^2 + \eta r^2 + \epsilon g^2)dt \tag{3.1}$$

$$J = \int_0^t (\nu a_y^2 + \gamma d_t^2 + \eta r^2 + \epsilon g^2)dt \tag{3.2}$$

The parameters shown in this cost function are soft constraints for the NMPC. In the above equations, ϕ and $\dot{\phi}$ are roll and time rate of change of roll, respectively. Distance of the vehicle from the target is annotated as d_t, r and g are the roughness and grade of the terrain the vehicle would be traversing, respectively, and a_y is the lateral acceleration of the vehicle. In calculation of a_y, no longitudinal slip is assumed, and Equation (3.3) is used.

$$a_y = V_x\dot{\psi} + g\sin(\phi_{grade}) \tag{3.3}$$

Where the bank angle of the terrain the vehicle is traversing is ϕ_{grade}.

The weights on each of the parameters described above are given by α, β, γ, η, ν, and ϵ. Of these weights, γ must be by far the largest to ensure that the vehicle drives to the target location. In order to keep the vehicle from driving in unsafe regions, additional hard constraints are added, and are shown in Equations (3.4 - 3.10).

$$d_{ob} > r_{avoid} \tag{3.4}$$

$$|\delta| \leq 45^\circ \tag{3.5}$$

$$|\phi| \leq \phi_{thresh} \tag{3.6}$$

$$V_{xmin} \leq V_x \leq V_{xmax} \tag{3.7}$$

$$r < r_{thresh} \tag{3.8}$$

$$g < g_{thresh} \tag{3.9}$$

$$|a_y| < a_{y_{thresh}} \tag{3.10}$$

The distance of the vehicle to an obstacle is d_{ob}, and r_{avoid} is the radius around the obstacle in which the vehicle is not allowed to enter. This r_{avoid} parameter is set by the certainty of the measured location of an obstacle; if the NMPC is not certain of the exact location of an obstacle, the controller will give the obstacle a wider berth. Likewise if the NMPC is certain of the location of an obstacle, the controller might drive closer to the obstacle in an effort to follow a more optimal path. As previously mentioned, δ is the steer angle of the vehicle, and it is constrained to be within the physical limitations of the vehicle. Lateral and longitudinal velocities, denoted by V_y and V_x respectively, are constrained to be within predefined threshold values, along with terrain roughness and grade. These constraints are imposed to ensure the vehicle does not perform any unsafe maneuvers, nor traverse any unsafe terrain.

3.2 Direct Shooting Method

A direct shooting method [14] was used to calculate cost and minimize the cost function. The direct shooting method gets its name by being analogous to a marksman shooting at a target. An initial guess of the control parameters is made, and then shot forward in time using Runge-Kutta fourth order integration over the length of the control horizon. The total cost over the control horizon is computed, and changes are then made to the control inputs accordingly. The process is then repeated until the target is reached within a certain threshold. The steer angle input was broken up into three parameters as shown in Equation (3.11), where t_f is the length of time of the control horizon.

$$\delta(t) = u_0 + u_1(t/t_f) + u_2(t/t_f)^2 \tag{3.11}$$

Steer angle was parameterized in this manner to constrain any path calculated by the NMPC to be quadratic. A quadratic path was chosen as a balance between the ability of the NMPC to generate different types of paths and computational expense of the NMPC. With a higher order parameterization, the NMPC will be able to generate different types of paths, but with each additional parameter introduced, the computational expense is increased. A quadratic path is robust enough to avoid obstacles and deliver the vehicle to the target location, and only requires three parameters to be optimized. Furthermore, with this parameterization, only three parameters correspond to the entire series of inputs over the control horizon, rather than having a different steer angle calculated at each time step. Not having to determine a new steer angle at each time step will greatly improve run time.

This parameterization results in a vector of control input parameters $\mathbf{U} = [u_0, u_1, u_2, V_{x_{des}}]^T$, where $V_{x_{des}}$ is desired longitudinal velocity. Desired longitudinal velocity is incorporated into the control input vector in order to allow the velocity of the vehicle to change based on the requirements of the terrain being traversed. Because a closed form solution for the optimal control input does not exist, a pattern search algorithm [15] was used to change the input parameters after shooting forward.

3.3 Pattern Search

The pattern search algorithm minimizes the cost function by calculating the cost over the prediction horizon for an initial guess of input parameters, $\mathbf{U}$, then manipulating the parameters to successively lower the cost. In order to lower the cost in this manner, a generating matrix Γ is made and is shown in Equation (3.12) below.

$$\Gamma = \begin{bmatrix} 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \end{bmatrix} \tag{3.12}$$

Given that $\mathbf{U}$ has four components, the size of the generating matrix is 4x9. The number of columns, n, is defined by 2*(number of inputs)+1.

Each column of Γ is multiplied by a predefined value, δ_u. This value is then added to the initial guess of $\mathbf{U}$, as shown in Equation (3.13).

$$u_n = u_{guess} + \Gamma_n \delta_u \tag{3.13}$$

The cost of each input, u_n, is then calculated. If the cost is lowered by any u_n, that u_n is stored as u_{guess} for the next iteration. If the cost is not lowered by any of the u_n values, then the value of δ_u is divided by two for the next iteration. Iterations continue until δ_u is reduced to such a small value that further changes to **U** are negligible. Pseudocode of this process is shown in Table 3.1 below.

Table 3.1: Pseudocode of Pattern Search Algorithm

```
while $\delta_u > 0.01$
    if $J(u_n) < J(u_{guess})$
        $u_{guess} = u_n$
    else
        $\delta_u = 0.5\delta_u$
    end
end
```

Further information on this algorithm can be found in [15]. The pattern search algorithm finds a local minimum, rather than a global minimum, and is therefore sensitive to the initial guess for the optimal control input. However, because the main contributing factor to cost is known to be the distance from the UGV to the target location, an intelligent initial guess of the control input vector can be made.

3.4 Run Time

The two parameters that most affect the run time of the NMPC are the time step of the numerical integrator used in the direct shooting method, and the prediction horizon. A trade-off exists for both of these parameters between performance and computational expense. Based on testing in simulation, 0.01 s appeared to be a good compromise between run time and performance for the numerical integrator. The NMPC was then coded in C++ with this time step in order to determine run time for real time implementation. Figure 3.1 shows the run time for one optimization routine over varying prediction horizons. It is important to note that the run time of the NMPC algorithm is also sensitive to changes in location of the target and obstacle relative to the vehicle. The run times displayed in Figure 6 are determined for a vehicle located at (0m, 0m), an obstacle located at (10m, 0m), and a target located at (30m, 0m). All run times were evaluated on a computer equipped with an i7 processor at 2.8 GHz.

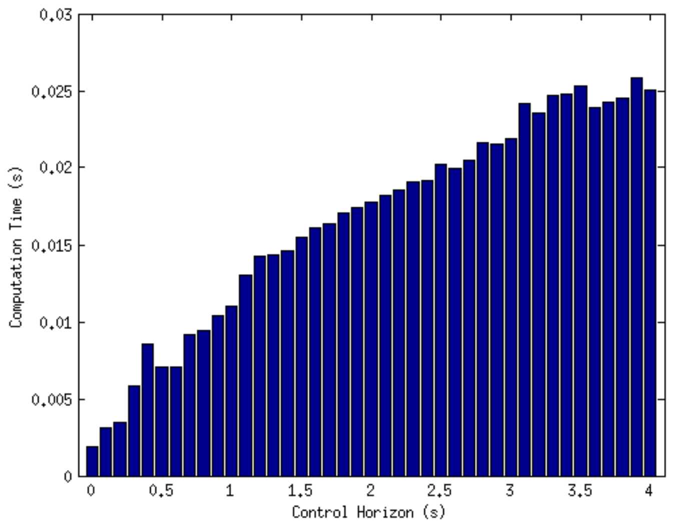

Figure 3.1: NMPC Run Times

From the above figure, it can be seen that control horizons from 0 - 4 seconds allow the NMPC to be able to run at 50 Hz. Therefore varying control horizons from 3 - 4 seconds were implemented. Simulations with of the NMPC are shown in the next chapter.

Chapter 4
NMPC Simulations

This chapter details the performance of the NMPC in two simulation environments. The first simulation environment was created in MATLAB using MATLAB's numerical integration tool, ode45 [16]. The second environment was developed using CarSim [17], a high fidelity vehicle simulation software. This chapter is broken up into three sections. First, the process used to simulate rough terrain is explored. Next, the simulation environment created in MATLAB is detailed and the results shown. In the third section of this chapter, the CarSim environment is detailed, and the results of those simulations are shown.

4.1 Terrain Generation

In order to simulate the performance of the NMPC algorithm over rough terrain, a point cloud of a terrain is generated using a Weierstrass-Mandelbrot function [18] [19]. This point cloud is broken up into grid cells, and a plane is fit to each grid using a least squares solution. Figure 4.1 below shows the generated terrain in MATLAB, while Figure 4.2 shows the same terrain with the planar least squares fit. The terrain shown in Figure 4.2 simulates the information given to the NMPC from the LiDAR measurements when determining the optimal path for the UGV. The terrain grade is calculated from the magnitude of the slope of each planar fit, and the terrain roughness is calculated from the residuals of the fit. The point cloud of generated terrain is provided to the CarSim simulation environment.

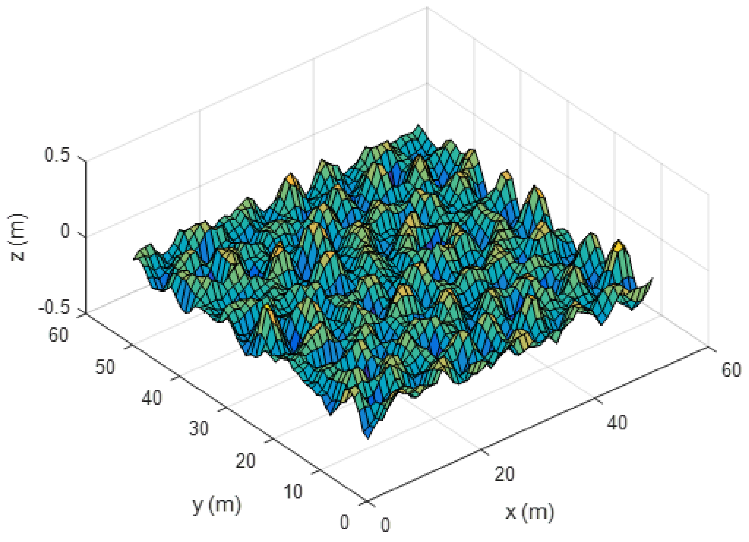

Figure 4.1: Generated Terrain

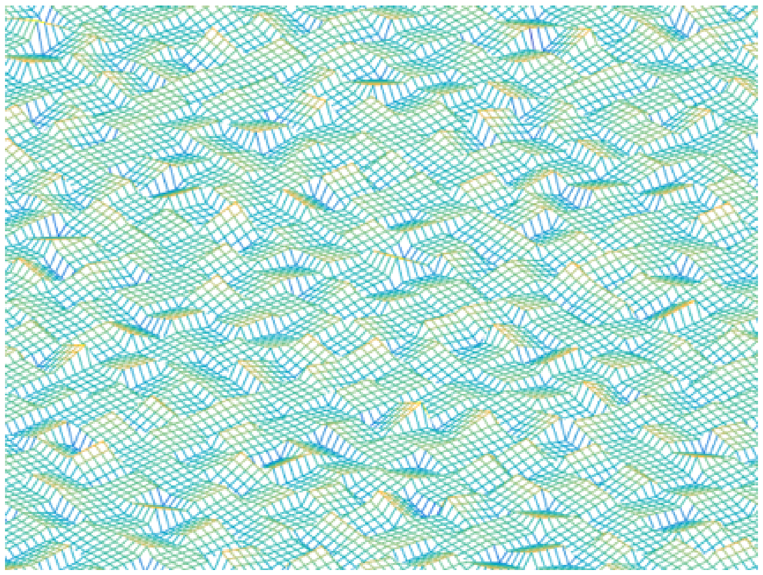

Figure 4.2: Terrain with Planar Fit

4.2 MATLAB Simulations

A simulation environment was created in MATLAB using an E-Class SUV model, shown in Figure 4.3. This vehicle model was chosen because much of the parameters of the vehicle were given in CarSim, which are provided in Table 4.1. Roll stiffness and roll damping (K_ϕ and C_f, respectively), were not given by CarSim. These parameters were hand tuned, and are only approximate values. In order to tune this parameters, known steer inputs were fed into a vehicle model in CarSim, and the roll of the vehicle was calculated. These same steer inputs were then fed into the vehicle model in the simulation environment created in MATLAB, and the roll stiffness and roll damping parameters were manipulated by hand in an effort to best match the results from the CarSim simulation.

Figure 4.3: E-Class SUV

Initially, simulations were conducted over flat terrain to examine the performance of the NMPC in the most benign conditions. The controller was run at 10 Hz, and the steer angle and longitudinal velocity computed by the NMPC were then fed into MATLAB's built in Runge-Kutta fourth order integrator, ode45. Transient changes of longitudinal velocity and steer angle of the vahicle was added by approximating changes in said angles and velocities as a first order response. In this way, when the desired steer angle or longitudinal velocity changed, the steer angle or velocity smoothly transitioned to the new target value rather than jumping to the new value like what would be seen in a step input. The system equations given to the NMPC and ode45 were the same. Table 4.2 shows the parameters used for

Table 4.1: Parameters of E-Class SUV

m	1590 kg
a	1.18 m
b	1.77 m
h_{cg}	0.72 m
I_{zz}	2687.1 kg m^2
I_{xx}	894.4 kg m^2
K_ϕ	100000 Nm/rad
C_f	10000 Nms/rad
B	9.55 rad^{-1}
C	1.3 rad^{-1}
D	6920 N

one of the simulations, and the results of the simulation are shown in Figures 4.4 - 4.6. Obstacle and target locations are given by (x, y) coordinates in a local reference frame, and the vehicle starts at (0 m, 0 m) for all simulations. All simulations on flat terrain feature the cost function shown in Equation (3.1). Variable terrain simulations also use the same cost function, unless otherwise stated below.

As shown in Table 4.2, a control horizon of 4 s was used. This control horizon was found to be a good compromise between ability to recognize and avoid an obstacle, and planning a more optimal path to the target location. Simulation results showed that the NMPC is more able to generate an optimal path with a lower control horizon, though less able to recognize obstacles in the path of the vehicle. This effect is owing to the parameterization of the steer input. Within a smaller control horizon, there are fewer possible paths for the NMPC to evaluate; therefore the NMPC is more likely to select an optimal one. However, with the smaller control horizon, the NMPC is also more likely to generate a path into a position to where it will be unable to escape collision with an obstacle when the obstacle is finally seen by the NMPC. In order to mitigate this risk, minimum and maximum values for control horizon can be set based on the speed of the vehicle and the number of obstacles in the vicinity of the vehicle.

Table 4.2: Parameters for MATLAB Flat Terrain Simulation 1

α	1
β	1
γ	10000
r_{avoid}	3 m
V_{xmax}	3 m/s
V_{xmin}	0 m/s
ϕ_{thresh}	15°
t_c	4 s
Target Location	(50 m, 0 m)
Obstacle Location	(25 m, 0 m)

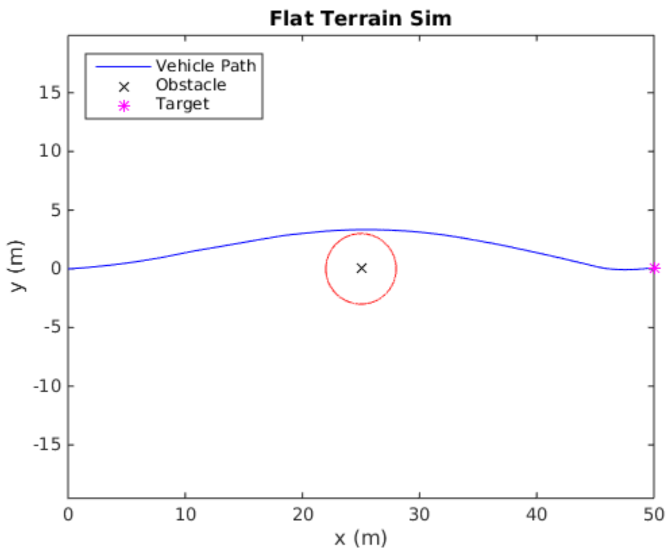

Figure 4.4: Vehicle Path for MATLAB Simulation 1, Flat Terrain

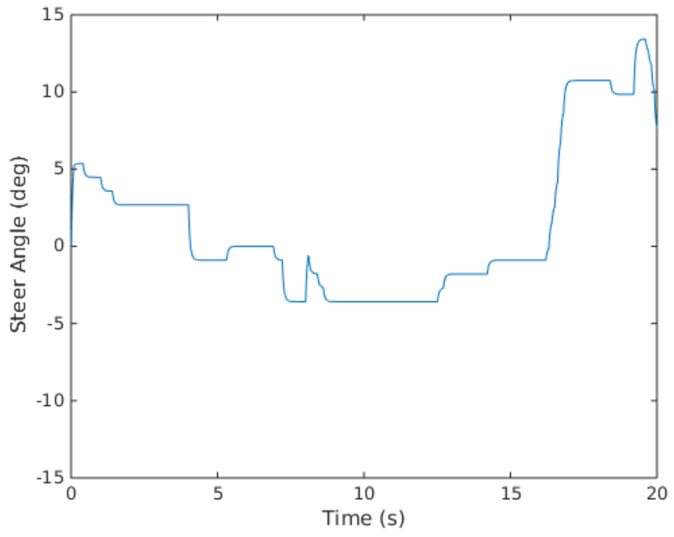

Figure 4.5: Steer Angle for MATLAB Simulation 1, Flat Terrain

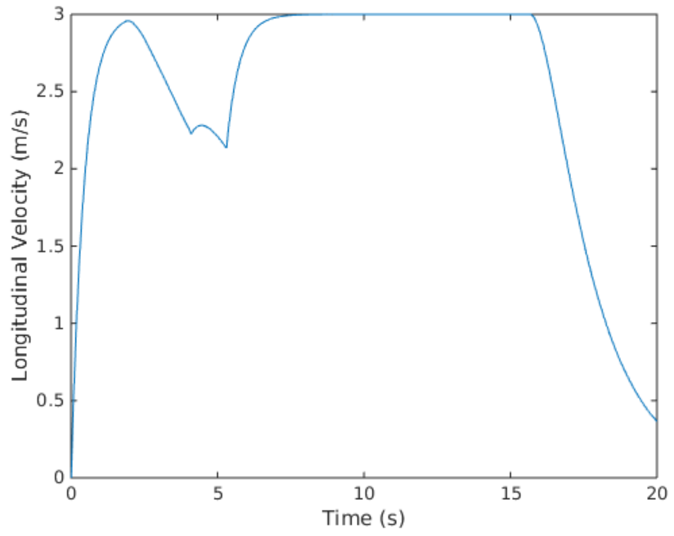

Figure 4.6: Longitudinal Velocity for MATLAB Simulation 1, Flat Terrain

The results of another flat terrain simulation (Simulation 2) are shown in Figures 4.7 - 4.9. For this simulation, the obstacle is located at (15 m, 15 m) and the target location is (30 m, 30 m). The control horizon is 3.5 s, r_{avoid} is 4 m, and the maximum longitudinal velocity is constrained to 4 m/s. For both flat terrain simulations, the NMPC generates a path that is a near global minimum of the cost function.

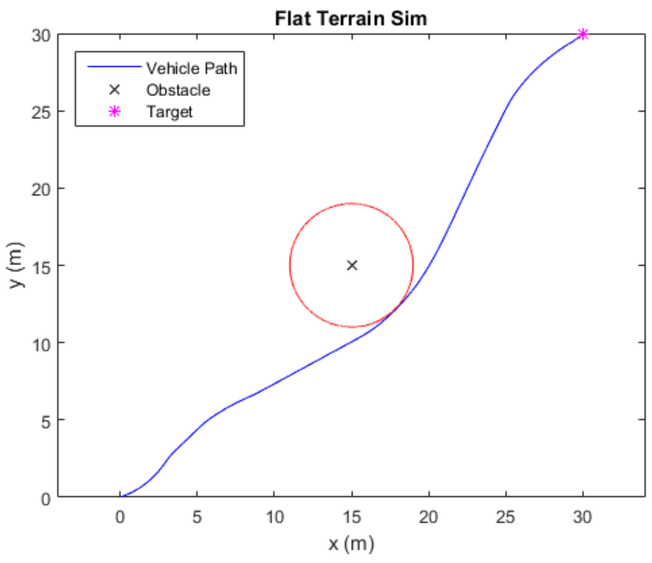

Figure 4.7: Vehicle Path for MATLAB Simulation 2, Flat Terrain

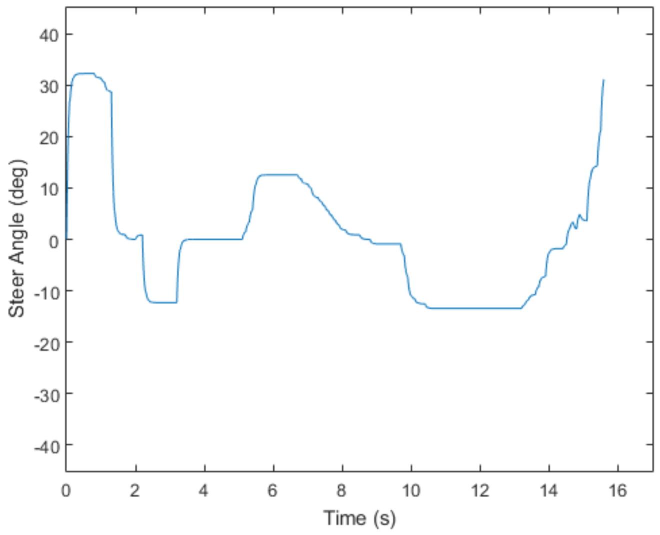

Figure 4.8: Steer Angle for MATLAB Simulation 2, Flat Terrain

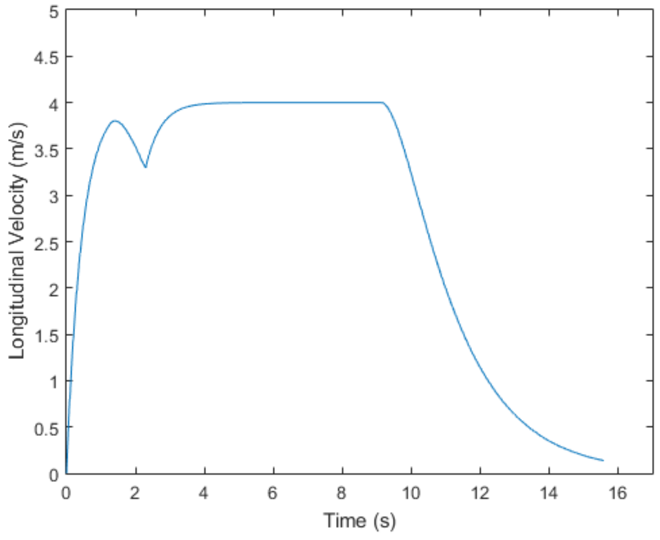

Figure 4.9: Longitudinal Velocity for MATLAB Simulation 2, Flat Terrain

Next, simulations of the vehicle over varying terrain roughness were conducted, as well as simulations featuring multiple obstacles. These simulations based on the MATLAB vehicle model do not take the effect of the terrain on the vehicle into account, but knowledge of the terrain roughness and grade does affect the path planned by the NMPC. An initial variable terrain simulation was conducted using two types of terrain: mildly rough traversable terrain, and moderately rough traversable terrain. For all simulations, green squares denote mildly rough terrain, and orange squares denote moderately rough terrain. Figures 4.10 - 4.13 show a simulation (Simulation 3) over mildly rough terrain with multiple obstacles. Parameters for the simulation are shown in Table 4.3.

Table 4.3: Parameters for MATLAB Simulation 3, Multiple Obstacles

α	1
β	1
γ	10000
η	10
ϵ	10
r_{avoid}	3.0 m
V_{xmax}	5 m/s
V_{xmin}	0 m/s
ϕ_{thresh}	15°
t_c	3.0 s
Target Location	(43 m, 0 m)
Obstacle Locations	(8 m, 10 m) (13 m, 0 m) (26 m, -3 m) (32 m, 5 m)

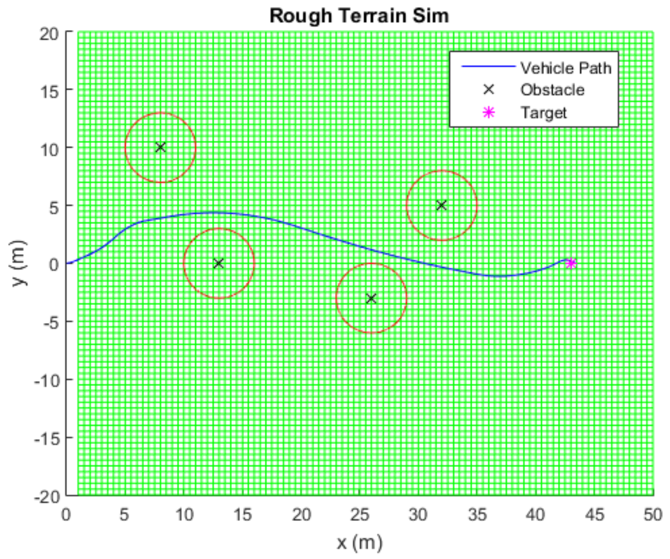

Figure 4.10: Vehicle Path for MATLAB Simulation 3, Multiple Obstacles

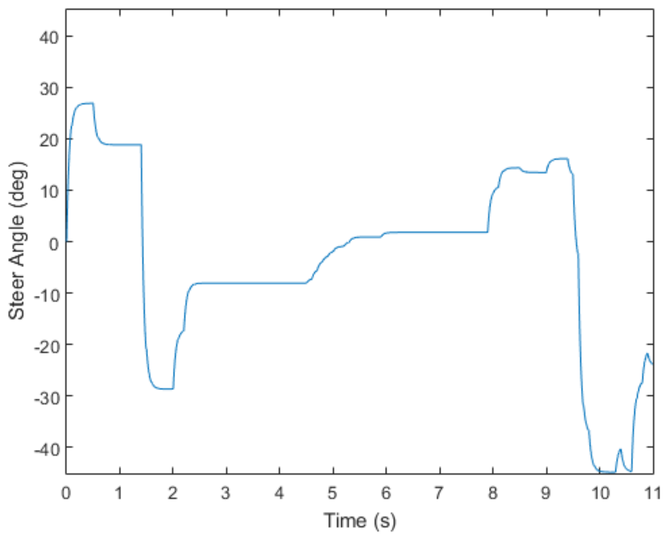

Figure 4.11: Steer Angle for MATLAB Simulation 3, Multiple Obstacles

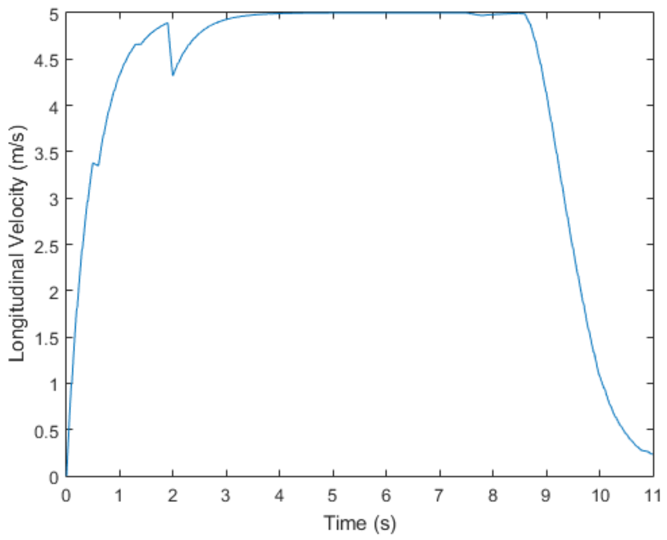

Figure 4.12: Longitudinal Velocity for MATLAB Simulation 3, Multiple Obstacles

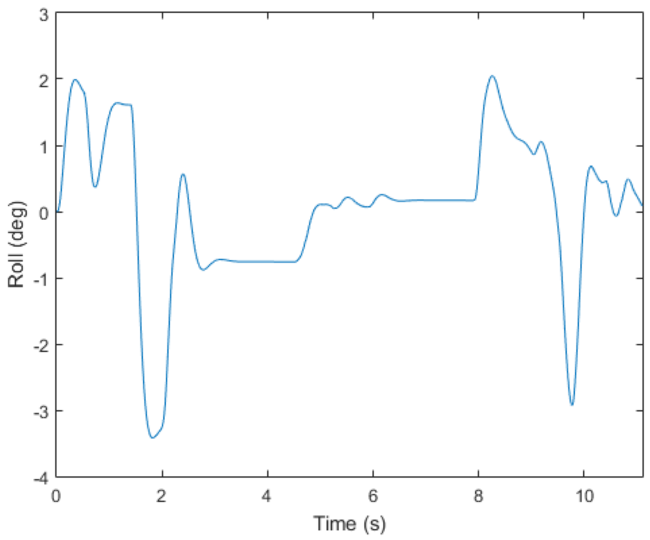

Figure 4.13: Vehicle Roll for MATLAB Simulation 3, Multiple Obstacles

Figures 4.14 - 4.17 show a simulation (Simulation 4) of the vehicle over both mildly rough and moderately rough terrain, and the parameters used in the simulation are shown in Table 4.4. In addition to the added cost of the NMPC generating a path through the moderately rough terrain, the maximum longitudinal velocity of the UGV was further constrained to be 2 m/s. This additional constraint imposes a velocity that is not practical for transportation, but the constraint is imposed to illustrate the ability of the NMPC to change target longitudinal velocity based on terrain roughness. While the path the NMPC generates does successfully avoid the obstacle and bring the UGV to the target, it is a sub-optimal trajectory. A more optimal path would be for the UGV to turn to the right when avoiding the obstacle, and avoid the moderately rough terrain altogether. Also, the path generated by the NMPC is unnecessarily aggresive when avoiding the obstacle. This is likely a result of poor characterization of the roll stiffness and roll damping of the vehicle. The suboptimality in the path generated by the NMPC is caused by the pattern search finding a local minimum rather than a global minimum.

Table 4.4: Parameters for MATLAB Simulation 4, Variable Terrain

α	1
β	1
γ	10000
η	10
ϵ	10
r_{avoid}	3.5 m
V_{xmax}	5 m/s
V_{xmin}	0 m/s
ϕ_{thresh}	15°
t_c	3.5 s
Target Location	(45 m, 0 m)
Obstacle Location	(18 m, 0 m)

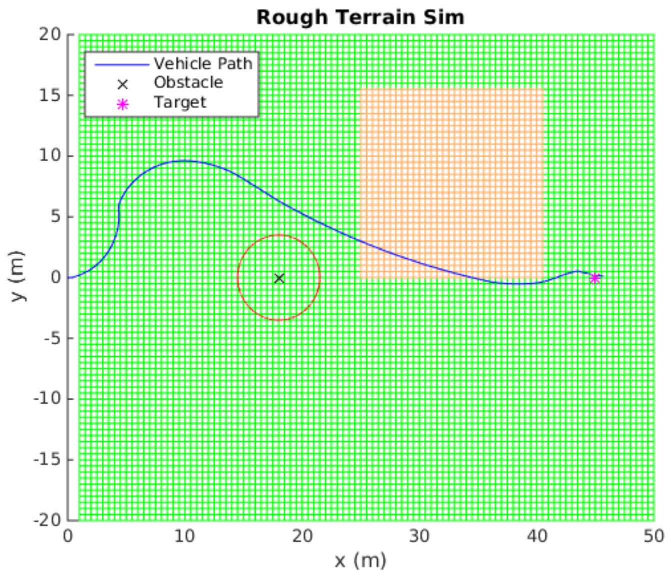

Figure 4.14: Vehicle Path for MATLAB Simulation 4, Variable Terrain

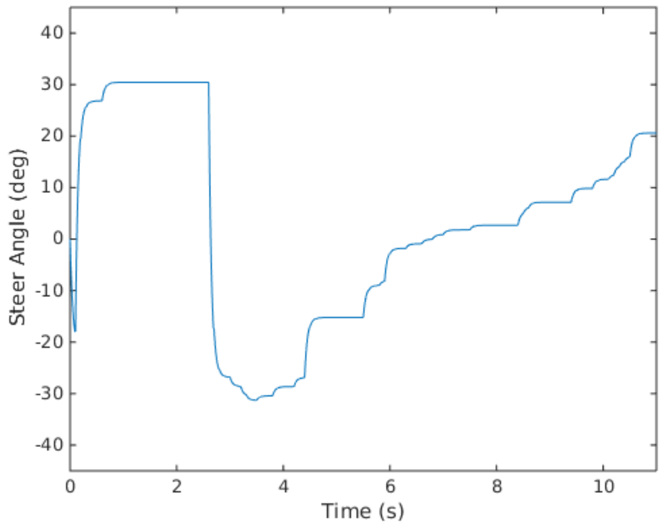

Figure 4.15: Steer Angle for MATLAB Simulation 4, Variable Terrain

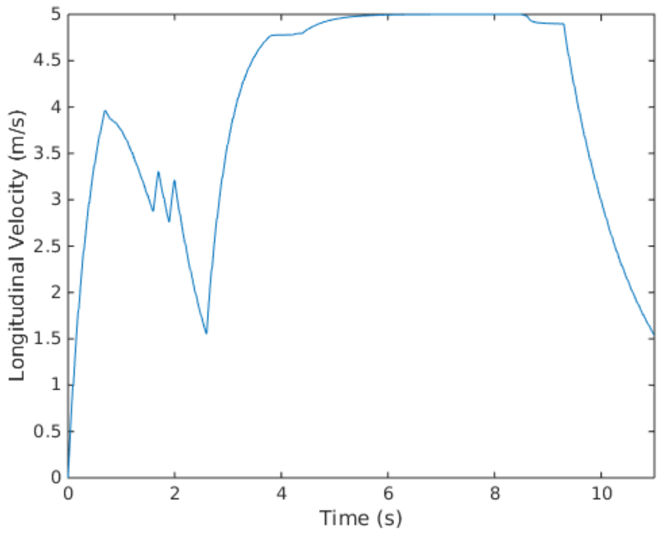

Figure 4.16: Longitudinal Velocity for MATLAB Simulation 4, Variable Terrain

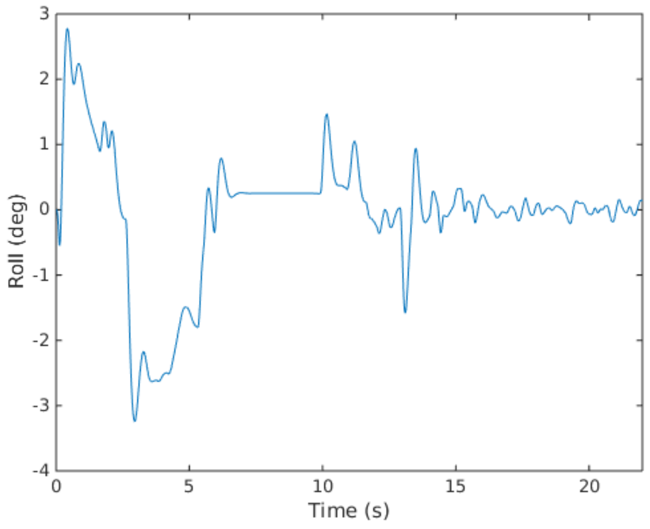

Figure 4.17: Vehicle Roll for MATLAB Simulation 4, Variable Terrain

In an effort to generate a path closer to the global minimum of the cost function, the lateral acceleration cost function was used (Equation (3.2)). Table 4.5 shows the parameters used in simulation with the lateral accleration cost function. Velocity within the moderately rough region of terrain (again, shown in orange) was constrained in the same manner as before. Simulation results (Simulation 5) are shown in Figures 4.18 - 4.20. The NMPC performs better with the lateral acceleration cost function, but still generates a suboptimal path. The path generated again takes the vehicle into the rough terrain. However, the vehicle does not unnecessarily aggresively steer away from the obstacle, and a much smoother trajectory is generated around the obstacle. This is likely due to the simpler vehicle model, which does not characterize roll directly. As a result, there is not a large amount of tuning and system identification required. As previously mentioned, in order to directly characterize vehicle roll accurately using the curvilinear trajectory roll model, the roll stiffness and roll damping parameters of the vehicle must accurately be known, and these parameters can be difficult to measure. Using a lateral acceleration model instead of a roll model allows for greater ease of implementation.

Table 4.5: Parameters for MATLAB Simulation 5, Variable Terrain

α	1
γ	5000
η	20
ϵ	20
r_{avoid}	3.5 m
V_{xmax}	5 m/s
V_{xmin}	0 m/s
$a_{y_{thresh}}$	6 m/s^2
t_c	4.0 s
Target Location	(45 m, 0 m)
Obstacle Location	(18 m, 0 m)

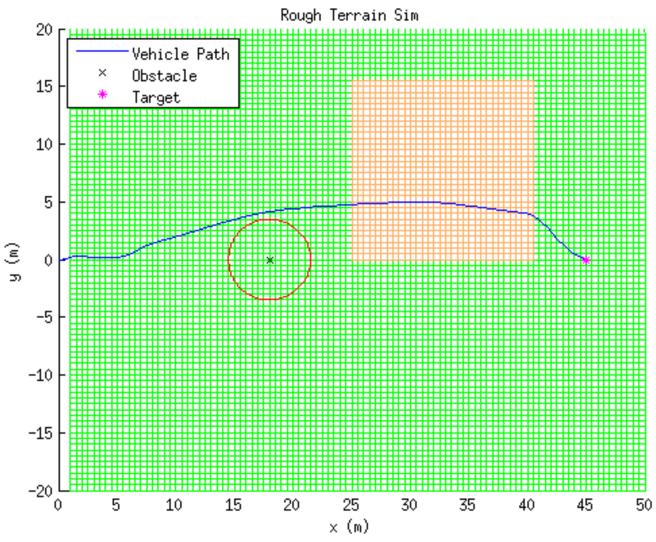

Figure 4.18: Vehicle Path for MATLAB Simulation 5, Variable Terrain

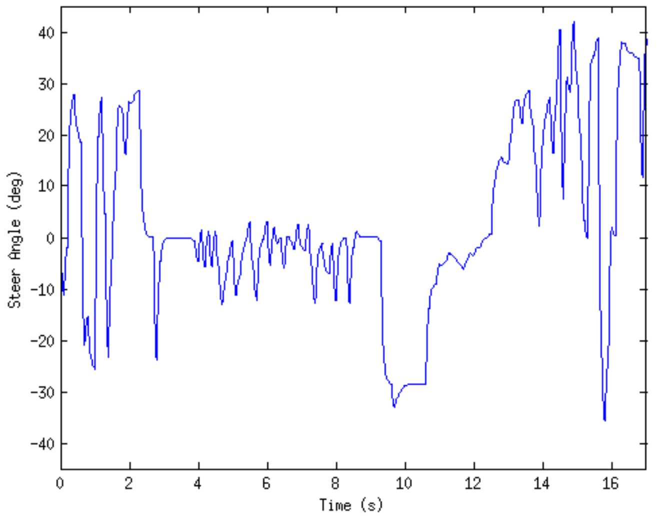

Figure 4.19: Steer Angle for MATLAB Simulation 5, Variable Terrain

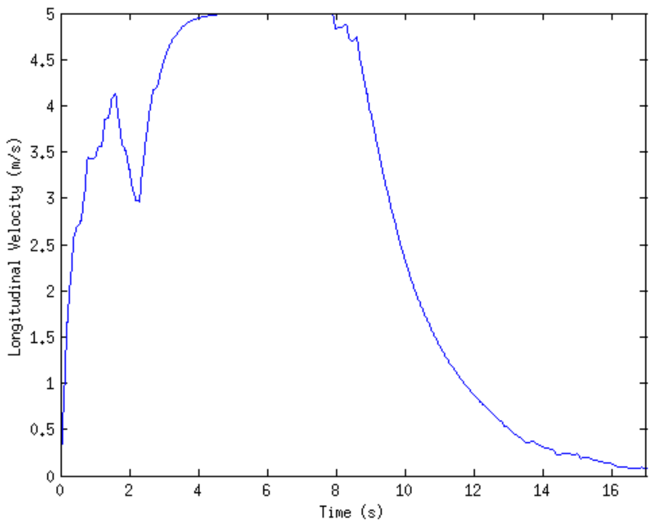

Figure 4.20: Longitudinal Velocity for MATLAB Simulation 5, Variable Terrain

4.3 CarSim Simulations

CarSim is a high fidelity vehicle modeling software with a Simulink interface that is capable of simulating vehicle motion on different types of terrain. Unlike the simulations developed in MATLAB, simulation environments created with CarSim can account for the effect of rough terrain on the motion of the vehicle. Figure 4.21 shows the Simulink model used for testing with CarSim.

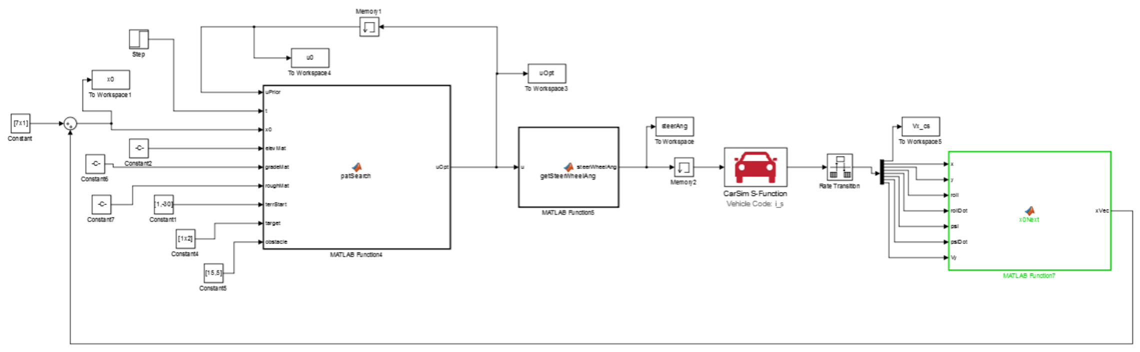

Figure 4.21: CarSim/Simulink Model

The only input provided by the NMPC to the vehicle in the CarSim environment was steer angle. The desired longitudinal velocity computed by the NMPC was constrained to be

2 m/s, and this velocity was set as a constant target for the vehicle in CarSim. The target location was set to (30 m, 10 m), and an obstacle was placed at (15 m, 5 m). r_{avoid} was set to 3 m. Unfortunately, transient changes in steer angle inputs could not be simulated without causing CarSim to crash, so the steer angle of the vehicle in the CarSim simulation was assumed to be able to instantaneously change when given new commands. The terrain shown in Figure 4.1 was used for simulation, and the NMPC was run at 2.5 Hz, which was the highest frequency that CarSim would allow. The cost function featuring the curvilinear trajectory roll model was used for simulation in CarSim (Equation 3.1). Figures 4.22 - 4.24 show the results of the CarSim simulation.

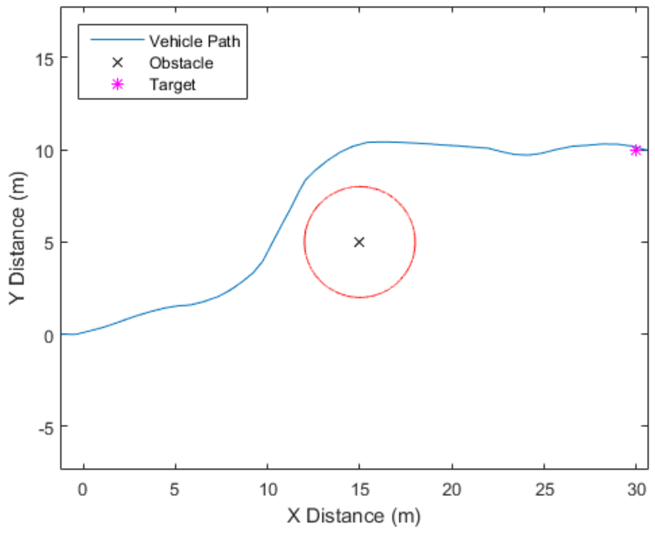

Figure 4.22: CarSim Simulation: Vehicle Path

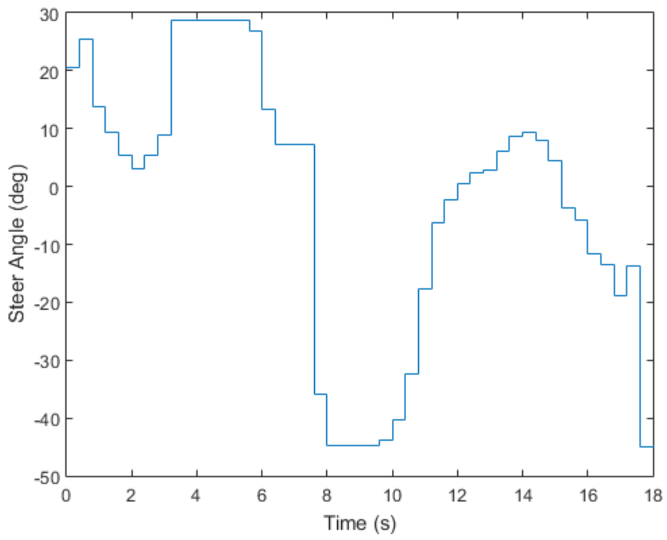

Figure 4.23: CarSim Simulation: Steer Angle

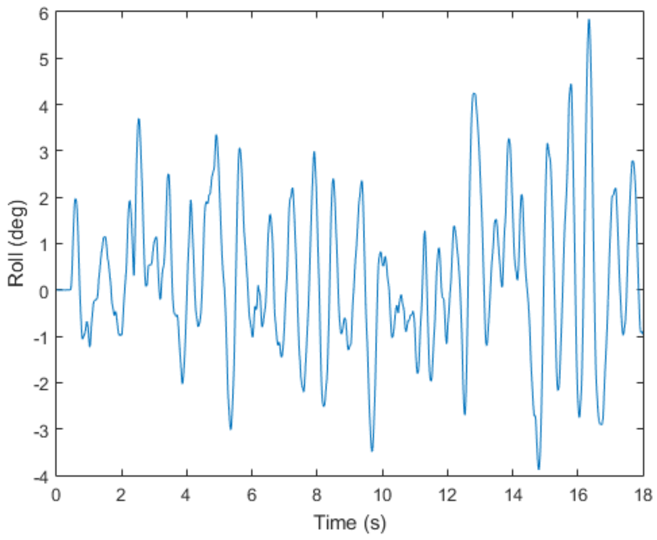

Figure 4.24: CarSim Simulation: Roll

As previously mentioned, the NMPC sometimes generates a suboptimal path to the target location as a result of the pattern search finding a local minimum. This generation of a suboptimal path highlights one of the drawbacks of NMPC. Owing to the suboptimality of paths generated by the NMPC, use of this NMPC is only recommended in situations where the ability to plan a path and control the vehicle simultaneously is required. In scenarios in which a desired path is known *a priori* and only vehicle control is required, use of a linear MPC in conjunction with a linearized vehicle model is recommended. However, in the absence of *a priori* path knowledge, the NMPC is able to provide a safe path to a target location.

Chapter 5
Experimental Implementation of NMPC

In order to validate the simulation results, the NMPC was implemented on Auburn's test vehicle, the Prowler. This chapter details the experimental implementation of the NMPC. The first section describes the experimental setup of the Prowler. The following section discusses sources of error that are unique to this particular experimental implementation. Finally, the last section shows experimental results.

5.1 Experimental Setup

The Prowler is an ATV test vehicle equipped with a GPS receiver, inertial measurement unit (IMU), on-board computer, and motors that control steering and throttle position. The Prowler is shown in Figure 5.1, and the motors used to control steering and throttle are shown in Figure 5.2. The vehicle parameters of the Prowler used in experimental implementation are shown in Table 5.1. All testing done with the Prowler was conducted on the skid pad at the National Center for Asphalt Technology (NCAT). The skid pad was chosen for testing because it is a large flat surface, and is a benign environment for testing performance of the NMPC. Because testing was done on a flat surface and the Prowler was not equipped with a sensor that could measure roll or roll rate (ϕ and $\dot{\phi}$ respectively), roll and roll rate were both assumed to be zero. Because roll stiffness and roll damping of the Prowler were unknown, an additional constraint was placed on changes in the steering wheel angle between successive time steps to be under a predefined threshold value of 12.5°. This was done because in the absence of roll stiffness and roll damping parameters, the NMPC is unable to characterize roll produced by steer input. This added constraint ensured the Prowler would not rollover from too aggressive a steer input.

Vehicle states measured by GPS were position, course, and longitudinal velocity. Yaw rate was measured by the IMU. Throughout experimental testing, course was assumed to be equivalent to heading (i.e. zero sideslip). Errors introduced by this assumption and other errors introduced in experimental implementation will be discussed more in the next section.

For experimental testing of the NMPC, target and obstacle locations were surveyed using GPS prior to running the NMPC, and both obstacle and target locations were then given to the NMPC. This mitigated much of the error in relative distance measurements between the vehicle and obstacle and vehicle and target. Once the target and obstacle locations were determined, the Prowler was initialized a distance away from the obstacle and in the direction in line with both the target and obstacle, and the NMPC was then started and allowed to control the vehicle. Because the initial course of the Prowler had to be determined, and reliable course measurements could only be obtained from GPS when the Prowler was moving, the first four control inputs to the steering wheel determined by the NMPC were thrown out. Initializing the course of the Prowler in this way allowed for measurements from GPS to be converted into a local frame.

Table 5.1: Parameters of the Prowler

m	544 kg
a	0.7239 m
b	0.7239 m
h_{cg}	0.4 m
I_{zz}	3500 kg m^2
B	9.55 rad^{-1}
C	1.3 rad^{-1}
D	6920 N

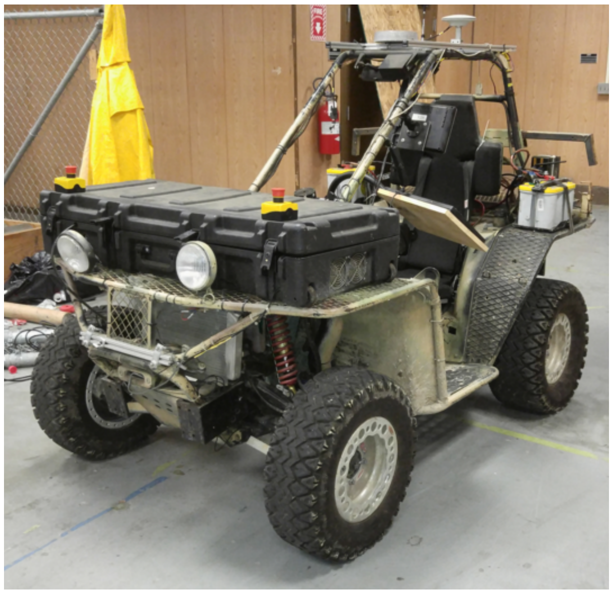

Figure 5.1: The Prowler

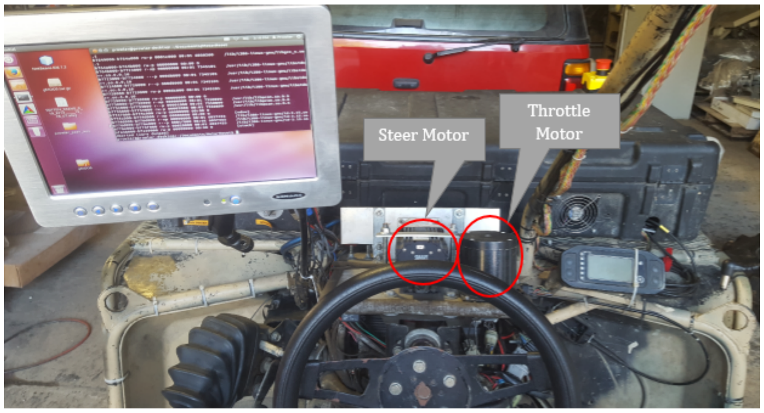

Figure 5.2: Prowler Control Motors

5.2 Sources of Error

Bringing the NMPC out of the simulation environment, and into an experimental environment introduces several sources of error that impacted the performance of the NMPC. Throughout the simulations shown in Chapter 4, perfect knowledge of the vehicle state vector was assumed to be known. This is obviously a poor assumption, as the measurements provided to the NMPC will contain noise. The noise in the measurements from the sensors (with the exception of course measurements), was small enough to be neglected.

A large source of error within the experimental implementation of the NMPC stemmed from position and course measurements provided by GPS. Because stand-alone GPS was used, the position solution provided to the NMPC had an error of 1 - 3 meters [20]. However, because the NMPC uses relative distance to target and obstacle locations, this error was largely mitigated. The main sources of error introduced by GPS measurements were from noisy course measurements. GPS course is measured by differencing carrier phase measurements, which produced a large amount of noise when the Prowler was moving at low speeds. The amount of noise in course measurements is shown by Figure 5.3. The effect of the noisy measurement of course is amplified as the target location is placed farther away from the initial location of the Prowler. This is because the target and obstacle locations are transformed into a local frame once the course of the Prowler is initialized. If the initial course of the Prowler is off by several degrees, the path the NMPC plans will have error in the predicted location of the Prowler. The farther the Prowler travels, the larger the error will be. After travelling 100 meters on the skid pad, the Prowler would often be 1 - 2 meters away from the target location owing to poor course initialization. In addition to poor course initialization, noisy course measurements will cause the steer input to fluctuate around a set point when driving straight. This was managed in experimental implementation by running the NMPC at a lower frequency. Several potential solutions to problems created by noisy course measurements exist. Course measurements should be filtered in some way before they are sent to the NMPC. However, implementation of a better navigation solution is outside the scope of this thesis.

Another source of error within experimental implementation is that many of the parameters, namely the tire model (i.e. the tire slip relationship), of the Prowler were not known to a high degree of certainty. Of the vehicle parameters shown in Table 5.1, only the mass of the Prowler along with the front and rear wheel base lengths were measured. Error in the model parameters of the vehicle causes the NMPC to plan a path that is different than the path generated once the inputs from the NMPC are applied to the vehicle. During experimental testing, it was evident that the linear region of the Prowler tires was much smaller than that of the vehicle model used by the NMPC. In order to minimize this error in the tire model, the Prowler was constrained such to not perform aggressive steering and stay within the linear region of the tire. In addition, the error in path generation caused by error in model parameters is directly proportional to the control horizon; therefore during experimental testing the control horizon was kept low. With these additional constraints, the error introduced from lack of knowledge of vehicle model parameters was kept to a minimum. Error in tire shape and mass moment of inertia parameters could be reduced by performing system identification on the test vehicle, and tuning the parameters in question to match datasets taken with the Prowler.

Finally, a large amount of error was introduced by poor control of velocity. For all experimental data sets, the velocity was constrained to be a set value, and a driver or a controller would attempt to control the vehicle to that value. However, at low speeds it is a non-trivial task to maintain the constant velocity expected by the NMPC, and the Prowler was often travelling at a velocity other than what the NMPC commanded. This has a similar effect to the error introduced by incorrect vehicle parameters; the NMPC is generating a path that is different than what is actually being realized.

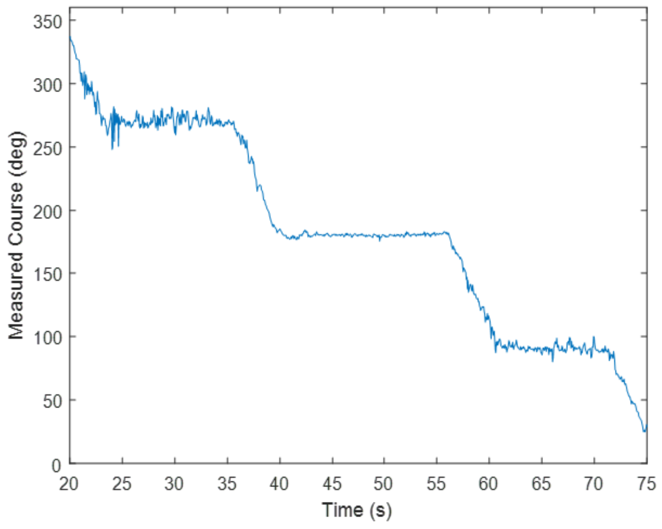

Figure 5.3: Noise in Course Measurement

5.3 Experimental Results

The NMPC was employed to initially control only the steer angle of the vehicle and the velocity was controlled manually. Once satisfactory results were obtained with steer control only, velocity control was implemented. The results of steer control only (semi-autonomous control) and steering and velocity control (fully autonomous control) are detailed in the sections below.

5.3.1 Semi-Autonomous Control

For the data shown in this section, the NMPC was constrained to a target velocity of 3 m/s. The driver of the Prowler manipulated the throttle in an effort to maintain this velocity. A constant control horizon of 3.5 s was used. Experimental results agree with simulation results in that the NMPC is able to generate a path around an obstacle and to a target location, but the path generated was suboptimal.

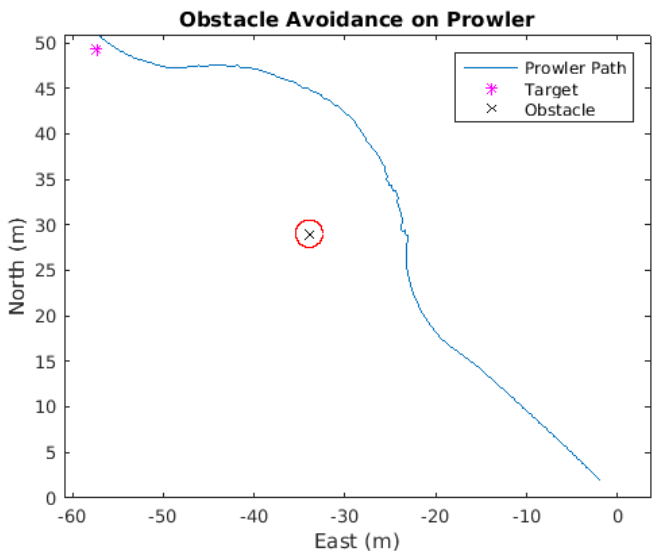

Figure 5.4: Semi-autonomous Obstacle Avoidance on Prowler

5.3.2 Fully Autonomous Control

For the sake of ease of implementation, the target velocity for fully autonomous data sets was constrained to be a set value of 4.5 m/s. Velocity was controlled to this target value with a lower level PID controller that commanded throttle position. The value of 4.5 m/s was chosen because it was large enough to be maintained by the PID controller with only throttle control, yet small enough to be able to safely test the NMPC. In order to obtain better results from the NMPC with velocity control implemented, the control horizon was scheduled based on distance of the vehicle to an obstacle. The control horizon was initially set to be 2.2 s, but was reduced to 1 s once the vehicle passed the obstacle. This scheduling of the control horizon allows the vehicle to more precisely hit the target location after avoiding the obstacle, but a suboptimal path is still generated. Figures 5.5-5.6 show experimental results of the NMPC controlling both steering and velocity.

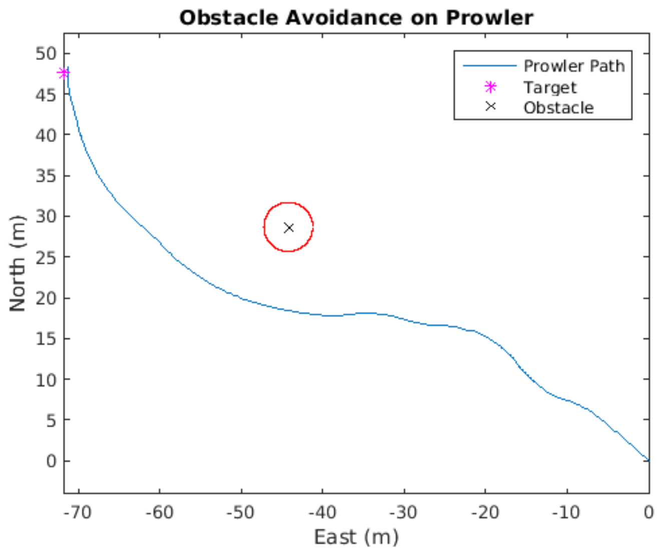

Figure 5.5: Fully Autonomous Obstacle Avoidance on Prowler: Simulation 1

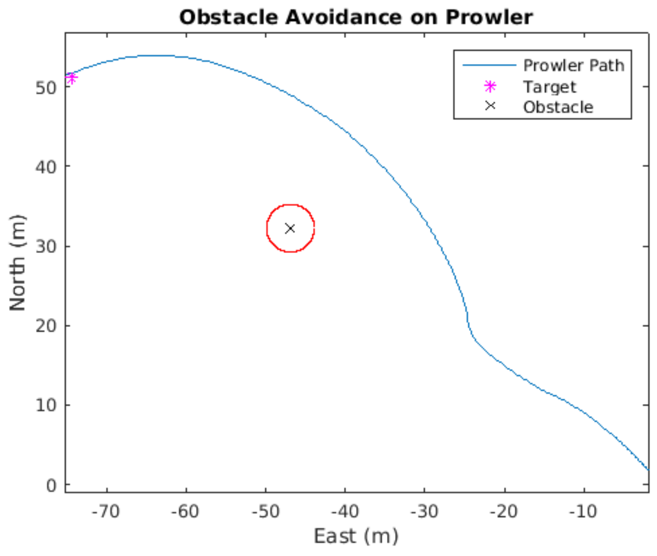

Figure 5.6: Fully Autonomous Obstacle Avoidance on Prowler: Simulation 2

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, a nonlinear model predictive control algorithm (NMPC) was presented for use with an unmanned ground vehicle. First, several vehicle models are presented, including the models used by the NMPC. The NMPC uses a lateral bicycle model with a Pacejka tire model in order to capture the planar dynamics of the vehicle, and a curvilinear trajectory model to capture the roll dynamics of the vehicle. The use of more complicated vehicle dynamic models may increase the accuracy of characterization of the motion of a vehicle, but at the cost of increased computational expense. In addition to added computational expense, more complicated models also contain additional parameters that are not easily measurable and can differ significantly from vehicle to vehicle. An example of this is the tire curve parameters used by the Pacejka tire model. These tire model parameters also change based on the vertical load on the tire and the tire pressure. If many of these parameters are not known to a high degree of certainty, the benefits of using these more complicated vehicle models is mitigated.

The NMPC algorithm presented in this thesis differs from NMPC algorithms in literature as the algorithm presented herein features a hard contraint on obstacle avoidance. The hard constraint on obstacle avoidance was desirable, as it ensures that the NMPC will not generate a path that would bring the vehicle an unsafe distance to an obstacle. However, introducing this hard constraint required using a different cost minimization method than what was used in literature. Instead of the cost minimization algorithms that are prevalent in literature, a pattern search algorithm was used. The pattern search algorithm and other cost minimization algorithms used for control of vehicles are suceptible to finding local minima rather than the global optimal solution. The nonlinearities of vehicle dynamics make determining a global minimum of cost very challenging. Control inputs do not appear directly in the equations of motion of the vehicle. Because of this, a closed form solution of the minimum of the cost function presented cannot be obtained.

Another challenge of NMPC is the run time required. Because model predictive control operates by calculating a series of inputs into the future at each time step, run time can be quite burdensome. The run time of the NMPC presented in this thesis was sufficiently small enough for the NMPC to operate at 10 Hz. The quickness of the run time owes itself largely to the parameterization of the control input vector. By parameterizing the steer input over the control horizon into three different components as shown in Equation (3.11), only three parameters are required to be calculated for an entire series of control inputs.

Simulations of the performance of the NMPC in two different simulation environments were presented: a simulation environment created in MATLAB, and a CarSim simulation environment. These simulations reveal the often suboptimal performance of the NMPC, but also validate the ability of the NMPC to navigate a vehicle to a target location around obstacles. The results from simulation are validated with experimental implentation of the NMPC on Auburn's test vehicle, the Prowler. Several errors are introduced when bringing the NMPC out of the simulation environments and into experimental implentation, and were detailed in Chapter 5. Testing was conducted at the National Center for Asphalt Technology on a skid pad, which was a smooth flat surface that offered a benign environment for testing the NMPC. Experimental tests confirmed what was shown by the simulations: the NMPC will navigate the vehicle around an obstacle and to a target location, but the path is often suboptimal.

Owing to the suboptimality of the path generated by the NMPC, a different path planning and control algorithm is recommended in situations where *a priori* knowledge of terrain roughness and obstacle locations is known, and the ability to generate a path and control inputs simultaneously is not needed. In the case where *a priori* information is available, a simpler control algorithm can adequately control the vehicle with less computational expense. However, in the absence of such knowledge, the work presented in this thesis lays a foundation for UGV control over variable terrain.

6.2 Future Work

There are several methods in which the experimental and simulation results could be improved upon. One such way would be to perform a more intensive search for the minimum of the cost function. The run time of the NMPC is well below the one second requirement of experimental implementation, and a slightly larger computational expense would not hinder performance of the NMPC. A potential way of performing a more intensive search for the minimum of the cost function would be to employ a "swarming method"[1] in conjunction with the pattern search. A "swarming method" would involve making several initial guesses for control input rather than just one, and finding the corresponding local minimum for each guess with the pattern search. The local minimum with the lowest cost would then be chosen to be input to the vehicle. Using this type of method would ensure that the NMPC did not arbitrarily prefer one direction of turning over another, for example because one turn direction yielded a local minimum before the other turn direction was tried.

As previously mentioned, there is a trade-off between optimality of the path generated and ability of the NMPC to recognize and avoid obstacles when setting the value for the control horizon. Throughout all of the simulations and some of the experimental data, the control horizon was a constant value. Better performance can be achieved if the control horizon changes throughout operation of the NMPC dependent on vehicle speed and relative location of detected obstacles.

[1]This idea is courtesy of a hot tub conversation in Utah with Dan Pierce.

As with any model based control method, knowledge of model parameters is invaluable. The performance of the NMPC in the experimental environment suffered from lack of complete knowledge of test vehicle parameters. Measuring many of these parameters, such as the tire curve model parameters, roll stiffness, and roll damping, is not feasible, but these parameters can be estimated using system identification. Once more accurate values for the vehicle parameters in question are obtained, the path generated by the NMPC will more accurately match the path driven by the vehicle.

Finally, as previously mentioned, a better vehicle navigation solution would enhance the performance of the NMPC. In experimental implementation, raw sensor measurements were fed directly to the NMPC. These measurements contained a lot of noise, and negatively impacted the performance of the NMPC. Rather than use raw measurements, a GPS/INS solution could be employed to filter measurements given to the NMPC. A GPS/INS solution would allow for a better determination of initial course of the vehicle as well as allow for more reliable updates of the heading of the vehicle, which would improve performance of the NMPC.

Bibliography

[1] Z. Lei, L. Yu, P. Ning, L. Xiaoxue, and S. Jian, "Vehicle Direct Yaw Moment Control Based on Tire Cornering Stiffness Estimation," in *ASME 2014 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers, 2014, pp. V003T49A004–V003T49A004.

[2] C. Liu, C. Lee, A. Hansen, J. K. Hedrick, and J. Ding, "A computationally efficient predictive controller for lane keeping of semi-autonomous vehicles," in *ASME 2014 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers, 2014, pp. V001T10A004–V001T10A004.

[3] C. E. Beal and J. C. Gerdes, "Model predictive control for vehicle stabilization at the limits of handling," *Control Systems Technology, IEEE Transactions on*, vol. 21, no. 4, pp. 1258–1269, 2013.

[4] S. M. Erlien, J. Funke, and J. C. Gerdes, "Incorporating non-linear tire dynamics into a convex approach to shared steering control," in *American Control Conference (ACC), 2014*. IEEE, 2014, pp. 3468–3473.

[5] A. Gray, M. Ali, Y. Gao, J. Hedrick, and F. Borrelli, "Semi-autonomous vehicle control for road departure and obstacle avoidance," *IFAC Control of Transportation Systems*, pp. 1–6, 2012.

[6] J. Liu, P. Jayakumar, J. L. Overholt, J. L. Stein, and T. Ersal, "The role of model fidelity in model predictive control based hazard avoidance in unmanned ground vehicles using lidar sensors," in *ASME 2013 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers, 2013, pp. V003T46A005–V003T46A005.

[7] Y. Yoon, J. Shin, H. J. Kim, Y. Park, and S. Sastry, "Model-predictive active steering and obstacle avoidance for autonomous ground vehicles," *Control Engineering Practice*, vol. 17, no. 7, pp. 741–750, 2009.

[8] J. Park, D. Kim, Y. Yoon, H. Kim, and K. Yi, "Obstacle avoidance of autonomous vehicles based on model predictive control," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 223, no. 12, pp. 1499–1516, 2009.

[9] J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal, "A multi-stage optimization formulation for mpc-based obstacle avoidance in autonomous vehicles using a lidar sensor," in *ASME 2014 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers, 2014, pp. V002T30A006–V002T30A006.

[10] R. N. Jazar, *Vehicle Dynamics Theory and Application.* Springer, 2008.

[11] J. Ryan, "A fully integrated sensor fusion method combining a single antenna gps unit with electronic stability control sensors," Master's thesis, Auburn University, 2011.

[12] R. Andrzejewski and J. Awrejcewicz, *Nonlinear Dynamics of a Wheeled Vehicle*, D. Y. Gao and R. W. Ogden, Eds. Springer, 2005.

[13] E. Bakker, H. B. Pacejka, and L. Lidner, "A new tire model with an application in vehicle dynamics studies," SAE Technical Paper, Tech. Rep., 1989.

[14] D. D. Morrison, J. D. Riley, and J. F. Zancanaro, "Multiple shooting method for two-point boundary value problems," *Commun. ACM*, vol. 5, no. 12, pp. 613–614, Dec. 1962. [Online]. Available: http://doi.acm.org/10.1145/355580.369128

[15] V. Torczon, "On the convergence of pattern search algorithms," *SIAM Journal on Optimization*, vol. 7, no. 1, pp. 1–25, 1997. [Online]. Available: http://dx.doi.org/10.1137/S1052623493250780

[16] MathWorks, *MATLAB The Language of Technical Computing*, 2015.

[17] C. U. Manual, "Mechanical simulation corporation," *Ann Arbor, MI*, vol. 48013, 2002.

[18] M. Ausloos and D. Berman, "A Multivariate Weierstrass-Mandelbrot Function," *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 400, no. 1819, pp. 331–350, 1985.

[19] J. J. Dawkins, D. M. Bevly, and R. L. Jackson, "Fractal terrain generation for vehicle simulation," *International Journal of Vehicle Autonomous Systems*, vol. 10, no. 1, pp. 3–18, 2012.

[20] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements, and Performance*, 2nd ed. Ganga-Jamuna Press, 2011.