

Nonholonomic Control Utilizing Kinematic Constraints of Differential and Ackermann Steering Based Platforms

Adam Kenneth Shoemaker

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mechanical Engineering

Alexander Leonessa, Chair
Andrew J. Kurdila, Co-Chair
Steve C. Southward

November 30, 2016
Blacksburg, Virginia

Keywords: Nonholonomic Systems, Mobile Robots, Ackermann Control, Differential
Steering Control, Saturation Constraints
Copyright 2016, Adam Kenneth Shoemaker

Nonholonomic Control Utilizing Kinematic Constraints of Differential and Ackermann Steering Based Platforms

Adam Kenneth Shoemaker

ABSTRACT

A nonholonomic tracking controller is designed and adapted to work with both differential steering and Ackermann steering based platforms whose dynamics are represented using a unicycle model. The goal of this work is to find a relatively simple approach that offers a practical alternative to bulky and expensive algorithms, but still bolsters applicability where many other lightweight algorithms are too lax. The hope is that this alternative will offer a straightforward approach for groups interested in autonomous vehicle research, but who do not have the resources or personnel to implement more complex solutions. In the first phase of this work, saturation constraints based on differential drive kinematics are added to ensure that the vehicle behaves intuitively and does not exceed user defined limitations. A new strategy for mapping commands back into a viable envelope is introduced, and the restrictions are accounted for using Lyapunov stability criteria. This stage of work is validated through simulation and experimentation. Following the development of differential drive methods, similar techniques are applied to Ackermann steering kinematic constraints. An additional saturation algorithm is presented, which likewise is accounted for using Lyapunov stability criteria. As with the differential case, the Ackermann design is validated through simulation and experimentation. Overall, the results presented in this work demonstrate that the developed algorithms show significant promise and offer a lightweight, practical solution to the problem of vehicle tracking control.

Nonholonomic Control Utilizing Kinematic Constraints of Differential and Ackermann Steering Based Platforms

Adam Kenneth Shoemaker

GENERAL AUDIENCE ABSTRACT

In this work, a position controller for ground vehicles is developed. The algorithm takes into account the constraints of both Ackermann and differential drive platforms. A simplistic model is used for the initial development of this control algorithm, and more rigid constraints are added based on the intended platform. The goal of this work is to find a relatively simple approach that offers a practical alternative to bulky and expensive algorithms, but still bolsters applicability where many other lightweight algorithms are too lax. The hope is that this alternative will offer a straightforward approach for groups interested in autonomous vehicle research, but who do not have the resources or personnel to implement more complex solutions. Throughout this work, we present the theoretical development as well as simulation and experiments to verify the efficacy of our approach. Overall, the results presented in this work demonstrate that the developed algorithms show significant promise and offer a lightweight, practical solution to the problem of vehicle tracking control.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Contribution and Outline	3
2	Nonholonomic Control Algorithm	4
2.1	System Definition	4
2.2	Controller Design	5
2.3	Simulation Results	8
2.4	Summary	13
3	Saturation Algorithm for Differential Drive Platforms	15
3.1	Differential Drive Kinematics	15
3.2	Differential Drive Constraints	17
3.3	Saturation Algorithm	18
3.3.1	Maintaining Lyapunov Criteria	19
3.4	Differential Drive Simulation Results	20
3.5	Summary	25
4	Experimental Results for Differential Drive Platforms	27
4.1	Experimental Setup	27
4.2	Potential Field Reference System Design	29
4.3	Tracking Performance	31

4.4	Summary	37
5	Saturation Algorithm for Ackermann Steering Based Platforms	39
5.1	Ackermann Steering Kinematics	39
5.2	Ackermann Steering Constraints	41
5.3	Dual Stage Saturation Algorithm	42
5.3.1	First Stage Saturation	43
5.3.2	Second Stage Saturation	45
5.3.3	Maintaining Lyapunov Criteria	49
5.4	Ackermann Steering Simulation Results	50
5.5	Summary	55
6	Experimental Results for Ackermann Steering Platforms	57
6.1	Experimental Setup	57
6.2	Tracking Performance	58
6.3	Application to Obstacle Avoidance	66
6.4	Summary	70
7	Conclusions and Future Research	71
8	Bibliography	74

List of Figures

2.1	Tracking of sinusoidal trajectory with $\alpha = 0.5$ and $\beta = 0.1$	10
2.2	Commanded following distance with $\alpha = 0.5$ and $\beta = 0.1$	10
2.3	Controller commands with $\alpha = 0.5$ and $\beta = 0.1$	11
2.4	Tracking of sinusoidal trajectory with $\alpha = 0.1$ and $\beta = 0.1$	12
2.5	Commanded following distance with $\alpha = 0.1$ and $\beta = 0.1$	13
2.6	Controller commands with $\alpha = 0.1$ and $\beta = 0.1$	13
3.1	Differential platform kinematics.	16
3.2	Differential drive command envelope (shaded).	17
3.3	Differential drive saturation example.	18
3.4	Differential platform tracking of sinusoidal trajectory with $v_{\max} = 5$ and $L_{\text{Diff}} = 0.915$	21
3.5	Differential platform commanded following distance with saturation (black) compared to no saturation (red).	22
3.6	Differential platform controller commands with $v_{\max} = 5$ and $L_{\text{Diff}} = 0.915$	22
3.7	Differential platform tracking of sinusoidal trajectory with $v_{\max} = 2$ and $L_{\text{Diff}} = 0.915$	24
3.8	Differential platform commanded following distance with $v_{\max} = 2$ and $L_{\text{Diff}} = 0.915$	24
3.9	Differential platform controller commands with $v_{\max} = 2$ and $L_{\text{Diff}} = 0.915$	25
4.1	Differential drive test platforms (TURTLES).	28
4.2	Tracking performance of waypoints at $v_{r_{\max}} = 0.5$	32
4.3	Total error, $e_1(t)$, at $v_{r_{\max}} = 0.5$	32

4.4	Controller commands (black) and measured vehicle response (red) at $v_{r_{\max}} = 0.5$.	33
4.5	Tracking performance of waypoints at $v_{r_{\max}} = 1$.	34
4.6	Total error, $e_1(t)$, at $v_{r_{\max}} = 1$.	34
4.7	Controller commands (black) and measured vehicle response (red) at $v_{r_{\max}} = 1$.	35
4.8	Tracking performance of waypoints at $v_{r_{\max}} = 1.5$.	36
4.9	Total error, $e_1(t)$, at $v_{r_{\max}} = 1.5$.	36
4.10	Controller commands (black) and measured vehicle response (red) at $v_{r_{\max}} = 1.5$.	37
5.1	Ackermann steering platform kinematics.	40
5.2	Ackermann steering command envelope (shaded).	42
5.3	Graphical representation of (a) S^- and (b) S^+ .	43
5.4	First stage representation of S with saturation example.	44
5.5	Region classification for Ackermann second stage saturation.	46
5.6	Region 1 examples for second stage saturation of Ackermann platforms.	47
5.7	Region 2 example for second stage saturation of Ackermann platforms.	48
5.8	Ackermann platform tracking of sinusoidal trajectory with $v_{\max} = 10$.	51
5.9	Ackermann platform commanded following distance with $v_{\max} = 10$.	51
5.10	Ackermann platform controller commands with $v_{\max} = 10$.	53
5.11	Ackermann platform tracking of sinusoidal trajectory with $v_{\max} = 2$.	54
5.12	Ackermann platform commanded following distance with $v_{\max} = 2$.	55
5.13	Ackermann platform controller commands with $v_{\max} = 2$.	56
6.1	Ackermann steering test platforms (TURTLES).	58
6.2	Tracking performance of waypoints with $k_{\omega} = 1$ and $v_{r_{\max}} = 2$.	60
6.3	Total error, $e_1(t)$, with $k_{\omega} = 1$ and $v_{r_{\max}} = 2$.	60
6.4	Controller commands (black) and measured vehicle response (red) for $k_{\omega} = 1$ and $v_{r_{\max}} = 2$.	61
6.5	Tracking performance of waypoints with $k_{\omega} = 2.5$ and $v_{r_{\max}} = 5$.	62
6.6	Total error, $e_1(t)$, with $k_{\omega} = 2.5$ and $v_{r_{\max}} = 5$.	62

6.7	Controller commands (black) and measured vehicle response (red) for $k_\omega = 2.5$ and $v_{r_{\max}} = 5$	63
6.8	Tracking performance of waypoints with $k_\omega = 3.5$ and $v_{r_{\max}} = 9$	64
6.9	Total error, $e_1(t)$, with $k_\omega = 3.5$ and $v_{r_{\max}} = 9$	65
6.10	Controller commands (black) and measured vehicle response (red) for $k_\omega = 3.5$ and $v_{r_{\max}} = 9$	65
6.11	Obstacle avoidance at low speeds with $k_\omega = 1$ and $v_{r_{\max}} = 2$	67
6.12	Measured vehicle velocity during obstacle avoidance with $k_\omega = 1$ and $v_{r_{\max}} = 2$	67
6.13	Obstacle avoidance at low speeds with $k_\omega = 2.5$ and $v_{r_{\max}} = 6$	68
6.14	Measured vehicle velocity during obstacle avoidance with $k_\omega = 2.5$ and $v_{r_{\max}} = 6$	68
6.15	Obstacle avoidance at low speeds with $k_\omega = 2.5$ and $v_{r_{\max}} = 9$	69
6.16	Measured vehicle velocity during obstacle avoidance with $k_\omega = 2.5$ and $v_{r_{\max}} = 9$	69

List of Tables

2.1	Parameters of simulation with $\alpha = 0.5$ and $\beta = 0.1$. All quantities expressed using standard SI units.	9
2.2	Parameters of simulation with $\alpha = 0.1$ and $\beta = 0.1$. All quantities expressed using standard SI units.	12
3.1	Differential platform simulation parameters with $v_{\max} = 5$ and $L_{\text{Diff}} = 0.915$. All quantities expressed using standard SI units.	20
3.2	Differential platform simulation parameters with $v_{\max} = 2$ and $L_{\text{Diff}} = 0.915$. All quantities expressed using standard SI units.	23
4.1	Differential platform experimental tracking parameters. All quantities expressed using standard SI units.	31
5.1	Ackermann platform simulation parameters with $v_{\max} = 10$. All quantities expressed using standard SI units.	50
5.2	Ackermann platform simulation parameters with $v_{\max} = 2$. All quantities expressed using standard SI units.	52
6.1	Ackermann platform experimental tracking parameters. All quantities expressed using standard SI units.	59
6.2	Ackermann platform parameters for obstacle avoidance experiments. All quantities expressed using standard SI units.	66

Chapter 1

Introduction

With the increasing demand for driverless technologies, the need for versatile vehicle control algorithms is at an all time high. On the surface, the need appears to be focused around algorithms capable of precision control at both high and low speeds to facilitate modern day automotive driving. These algorithms, however, can tend to be bulky and complicated, often taking substantial resources both in personnel and computation to implement. In many cases, these constraints are prohibitive for groups interested in research in topics outside of vehicle control. Moreover, the use of vehicle control algorithms specifically guided for autonomous road vehicles, while important, can be somewhat short sighted as other fields can benefit from less aggressive forms of this technology. In this work, we develop a controller aimed at filling a gap between highly complicated and demanding algorithms capable of full precision control and simple algorithms that often do not take into account basic vehicle constraints. The result offers a practical lightweight algorithm that has directly contributed to the works of [1, 2, 3] and lead to further development in [4].

1.1 Motivation

Of the more expensive algorithms currently in place for vehicle motion control are state lattice search algorithms ([5, 6]). These algorithms function by generating a set of path primitives, which are essentially a set of actions that describe the vehicles motion from one particular state to another. To implement these actions, the viable control space is discretized so that an infinite set of possible control inputs is simplified into a finite set of discrete actions ([7]). In doing this, the vehicle control problem becomes akin to graph theory in which the vehicle states are individual nodes and the path primitives, which hold all motion information, become edges to the graph, connecting the nodes.

Once posed as a graph search problem, a variety of search methodologies can be used such as A* or its variants, ARA* or AD* for example ([8]). Clearly through the advent of the motion

primitives, this allows the vehicles control to be arbitrarily complex as the goal is to search through a set of possible actions to find the desired behavior. The main limiting factor in this case, however, is that search algorithms can be computationally expensive from a control perspective, and this complexity increases dramatically as the control action discretization size decreases or the dimensionality of the viable control actions increases ([9, 10]). Moreover, the complexity of implementing these techniques may be beyond the resources in personnel available to many groups interested in research scopes beyond motion control.

In an effort to mitigate some computation cost, other researchers look toward optimal control solutions. In the case of the work in [11], the authors use an optimal control approach to path generation. The drawback in this case, though, is that the inverse of the nonlinear dynamics require an iterative search, still adding to the overall complexity and expense for this method ([11]). Other more traditional control techniques to vehicle positioning have been proposed in the literature. There are two major problems with many of these approaches for those not intimately familiar with the advanced techniques of the field. The first problem concerns methodologies that are rooted in immense theory such as optimal control techniques which generally prove too great a burden for small scale practical implementation([12, 13]). Meanwhile the other category lies in simpler approaches that are too highly specialized for a specific task ([14, 15, 16]).

On the other end of the spectrum from expensive and complicated algorithms lies the research of simpler and generally practical algorithms for vehicle planning and control. One approach that lies in this realm is the use of potential field algorithms, which use basic physics to generate intended paths for vehicles ([17, 18, 19, 20]). While typically light weight and straightforward to implement, these methods often do not take into account vehicle capability. Other similar methods use control theory with basic kinematic models to lightly consider vehicle motion in the design ([21, 22, 23]).

In the scope of control approaches that use basic kinematic models, of particular interest to this work are the contributions of [24] and [25]. In [24] the authors introduce a control law designed to drive a vehicle, described with a basic unicycle model, to within some neighborhood of a user defined reference system. This neighborhood is chosen to be sufficiently large enough to allow the vehicle to maintain its proximity to the reference system despite its nonholonomic constraints. The author's approach in [24] is ultimately shown to be stable by invoking Lyapunov stability criteria.

The work of [24] is extended in [25]. In this work, the authors introduce the concept of a time varying distance between the reference system and vehicle. By choosing the time varying distance to correspond with the reference system's speed, the authors are able to improve tracking at lower speeds while still allowing for sufficient reaction time at higher speeds. With this general approach both the works of [24] and [25] offer a light weight control alternative to vehicle positioning.

While useful in some regards, the control methods presented in [24] and [25] do not adequately take vehicle capability constraints into consideration in their designs. This lack of regard

for basic vehicle constraints, in turn, limits the efficacy of their approaches. In an attempt to maintain the simplicity and usability of these approaches, this thesis seeks to expand on the work of [25] by adding constraints based on reasonable vehicle limitations for both differential and Ackermann steering based platforms and incorporating these constraints into the control design.

1.2 Contribution and Outline

The presentation of this work is organized into six subsequent chapters. In Chapter 2 we discuss the algorithm presented in [25]. In addition to this algorithm, we supplement the design added restrictions on the form and dynamics of the system. These modifications are aimed at bolstering the usability and reliability of the algorithm. Moreover, we present observations and limitations about the design of this controller aimed at giving more insight to its workings.

In Chapter 3 we introduce new saturation algorithms based on differential steering kinematics, not previously seen in literature to the best of the author's knowledge. The control law is then updated to maintain Lyapunov stability criteria in cases where the controller is pushed to the edge of its operation envelope. This approach is then validated in Chapter 4 through experiment in which the algorithm is implemented on a small scale differentially steering ground robot.

Following differential drive vehicles, the control design is extended to Ackermann steering platforms in Chapter 5. Similar to the design in Chapter 3, a new saturation algorithm based on Ackermann steering kinematics is introduced. As with the differential drive setup, the Ackermann saturation methodology is validated through experiments in Chapter 6. Lastly, we conclude with a look back on the overall design and contribution in Chapter 7 and offer guides to building on the research in the future.

Chapter 2

Nonholonomic Control Algorithm

In this chapter, we will discuss the controller originally developed in [25]. In this discussion, we introduce additional restrictions placed on the system structure and following distance parameters. These modifications are aimed at bolstering reliability and implementation practicality.

2.1 System Definition

We begin with a unicycle model as it adequately captures the nonholonomic constraints of a variety of ground platforms. While there are admittedly more complex approaches to vehicle dynamics, such as the bicycle and four wheel model variants ([26, 27, 28]), the unicycle model's simplicity directly lends itself to controller design, as seen in several applications in the literature ([29, 30, 24]). The dynamics of the unicycle model are given as follows,

$$\dot{p}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} v(t) \cos(\theta(t)) \\ v(t) \sin(\theta(t)) \end{bmatrix}, \quad (2.1)$$

$$\dot{\theta}(t) = \omega(t), \quad (2.2)$$

where $p(t) \triangleq [x(t) \ y(t)]^T \in \mathbb{R}^2$, is the vehicle's position, $v(t) \in \mathbb{R}$, is the longitudinal velocity, $\theta(t) \in \mathbb{R}$, is the heading, and $\omega(t) \in \mathbb{R}$, is the vehicle's angular velocity. For this work, we will consider the longitudinal and angular velocities to be direct inputs into our system. In practice, we will use low level PI controllers on the platforms to facilitate this assumption.

In order to control the position of the vehicle, we introduce a reference system to represent a virtual target,

$$p_r(t) \triangleq \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix}, \quad (2.3)$$

where $p_r(t) \in \mathbb{R}^2$, is the reference system's position. In the interest of continuous controller commands, we must guarantee that the reference system is composed of class C^1 functions in time, which is a constraint not previously mentioned in [25]. For convenience, the linear velocity of the reference system is defined as,

$$v_r(t) \triangleq \sqrt{\dot{x}_r(t)^2 + \dot{y}_r(t)^2}, \quad (2.4)$$

where $v_r(t) \in \mathbb{R}$.

While the reference system must consist of class C^1 functions, there is no requirement that its trajectory must behave in a manner directly achievable by the vehicle. To compensate for this potential issue, an additional state, $d(t) > 0$, is introduced, which can be thought of as the commanded following distance between the reference system and vehicle. By design we will ensure that the distance between the vehicle and reference system will converge to $d(t)$ such that $\|p(t) - p_r(t)\| \rightarrow d(t)$ as $t \rightarrow \infty$. By ensuring that the actual system converges to a small neighborhood of the reference system, we ensure that the system follows the desired trajectory as well. At the same time, a sufficiently large separation between reference system and vehicle allows a buffer for the vehicle to track the reference, without violating its nonholonomic constraints.

It is foreseeable that the desired distance may vary based on application. For this reason, we allow the user control to define a nominal following distance, $d^*(t) > 0$. In addition to guaranteeing $d^*(t) > 0$, the nominal following distance must also be a class C^1 function in time. Through the controller development in the following section, we will design $d(t)$ such that $d(t) \rightarrow d^*(t)$ as $t \rightarrow \infty$.

2.2 Controller Design

In designing the controller, it is necessary to define the error between the vehicle and reference system. We start by assuming that the origins of the body fixed coordinate frame, \mathcal{B} , and of the global inertial coordinate frame, \mathcal{U} , coincide with the center of mass of the vehicle in the horizontal plane. An orthonormal transformation from \mathcal{B} to \mathcal{U} is then defined,

$$R(\theta(t)) \triangleq \begin{bmatrix} \cos(\theta(t)) & -\sin(\theta(t)) \\ \sin(\theta(t)) & \cos(\theta(t)) \end{bmatrix}. \quad (2.5)$$

Using this transformation along with (2.1) and (2.3), the position error can be expressed in the body coordinate frame as,

$$e(t) \triangleq R^T(\theta(t))(p_r(t) - p(t)), \quad (2.6)$$

where $e(t) \in \mathbb{R}^2$. This position error can be thought of as a vector expression of longitudinal and lateral error in the body frame. Next we introduce the commanded distance vector,

$$\delta(t) \triangleq \begin{bmatrix} d(t) \\ 0 \end{bmatrix}, \quad (2.7)$$

where $d(t) > 0$ is the commanded following distance mentioned in the previous section. By guaranteeing that $e(t) \rightarrow \delta$ as $t \rightarrow \infty$, the longitudinal error will converge to $d(t)$ while the lateral error goes to zero.

To provide this behavior, we introduce the following control law from [25],

$$\begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} = \Delta^{-1}(t) \left(K \tanh(e(t) - \delta(t)) + R^T(\theta(t)) \dot{p}_r(t) - \dot{\delta}(t) \right), \quad (2.8)$$

where,

$$\Delta(t) \triangleq \begin{bmatrix} 1 & 0 \\ 0 & d(t) \end{bmatrix}, \quad K \triangleq \begin{bmatrix} k_v & 0 \\ 0 & k_\omega \end{bmatrix}, \quad (2.9)$$

and $k_v > 0$ and $k_\omega > 0$ are scalar tuning constants used to provide bounded signals to the command longitudinal and angular velocities, respectively. This control law, which is more thoroughly derived in [25], is based on Lyapunov stability theory. In [25], the authors show it to be the natural result that guarantees a negative definite Lyapunov time derivative, given the Lyapunov function, (2.11), introduced in the following theorem.

In order to ensure that $\Delta^{-1}(t)$ is defined for all $t \geq 0$, we must guarantee that $d(t) \neq 0$ for all $t \geq 0$. In practice, if $d(t) < 0$, the vehicle will follow the reference system but in reverse, such that $v(t) < 0$. As such, we restrict the commanded following distance to $d(t) > 0$. For design purposes, a more rigorous constraint of $d(t) > (\beta - \epsilon) > 0$ for all $t \geq 0$, is enforced, where $\beta > \epsilon > 0$ are tuning parameters used to place a minimum bound on $d(t)$. Since $d(t)$ is designed such that $d(t) \rightarrow d^*(t)$ as $t \rightarrow \infty$, the user defined nominal following distance is further restricted such that $d^*(t) \geq \beta$.

At this point, we note the intuitive form of (2.8). For instance, from the $R^T(\theta(t))\dot{p}_r(t)$ term, we see that the reference velocity, projected along the vehicle's longitudinal axis, positively contributes to the commanded velocity, $v(t)$. Likewise the reference velocity, projected along the lateral axis, contributes to the commanded angular rate, $\omega(t)$. In examining the $\dot{\delta}(t)$ term, we see that an increasing distance negatively impacts the commanded velocity, as would be expected. Lastly, with regard to the $K \tanh(e(t) - \delta(t))$ term, we see that the difference between longitudinal error and the commanded distance, $d(t)$, contributes positively to the commanded velocity. Along the same lines, lateral error contributes positively to the angular rate command.

With regard to the $\tanh(\cdot)$ term of (2.8), which is a componentwise operation, we note that while it is unnecessary for theoretical stability, it adds a layer of feedback that can be tuned through K . The $\tanh(\cdot)$ function is chosen for its natural saturation capability. Based on the constants k_v and k_ω , the function directly contributes bounded signals to commanded longitudinal and angular velocities, which are based on the difference between the tracking error $e(t)$ and distance vector $\delta(t)$.

Theorem 2.2.1. *Consider the system described by (2.1) and (2.2), the reference system described by (2.3), and the feedback controller described by (2.8). If the nominal distance is*

restricted such that $d^*(t) \geq \beta$ for all $t \geq 0$, and the distance $d(t)$ is updated according to

$$\dot{d}(t) = \begin{cases} \dot{d}^*(t) - \lambda(d(t) - d^*(t)), & d(t) \geq \beta, \\ \dot{d}^*(t) - \lambda(d(t) - d^*(t)) + \frac{\beta - d(t)}{d(t) - (\beta - \epsilon)}, & d(t) < \beta, \end{cases} \quad d(0) \geq \beta \quad (2.10)$$

where $\lambda > 0$ is a tuning constant, then the commanded distance $d(t)$, between the vehicle and reference system, converges to the desired distance, $d^*(t)$, while guaranteeing that $d(t) > (\beta - \epsilon) > 0$. Meanwhile, the tracking error $e(t)$ converges to the distance vector $\delta(t)$.

Proof. Consider the Lyapunov function candidate

$$V(t) = \frac{1}{2}e_1^T(t)e_1(t) + \frac{1}{2}(d(t) - d^*(t))^2, \quad (2.11)$$

where

$$e_1(t) \triangleq e(t) - \delta(t). \quad (2.12)$$

In order to examine the derivative of the Lyapunov function along the system trajectories, the time derivative of the tracking error, $e(t)$, given by (2.6), is computed as follows,

$$\dot{e}(t) = \dot{R}^T(\theta(t), \omega(t))(p_r(t) - p(t)) + R^T(\theta(t))(\dot{p}_r(t) - \dot{p}(t)). \quad (2.13)$$

The time derivative of the orthonormal transformation, $R(\theta(t))$, is given by,

$$\begin{aligned} \dot{R}(\theta(t), \omega(t)) &= \begin{bmatrix} -\omega(t) \sin(\theta(t)) & -\omega(t) \cos(\theta(t)) \\ \omega(t) \cos(\theta(t)) & -\omega(t) \sin(\theta(t)) \end{bmatrix} \\ &= R(\theta(t))S(\omega(t)), \end{aligned} \quad (2.14)$$

where

$$S(\omega(t)) \triangleq \begin{bmatrix} 0 & -\omega(t) \\ \omega(t) & 0 \end{bmatrix}. \quad (2.15)$$

By substituting (2.1) and (2.14) into (2.13), the error dynamics can be simplified to,

$$\begin{aligned} \dot{e}(t) &= S^T(\omega(t))R^T(\theta(t))(p_r(t) - p(t)) + R^T(\theta(t))\dot{p}_r(t) \\ &\quad - R^T(\theta(t)) \begin{bmatrix} v(t) \cos(\theta(t)) \\ v(t) \sin(\theta(t)) \end{bmatrix} \\ &= -S(\omega(t))e(t) + R^T(\theta(t))\dot{p}_r(t) - \begin{bmatrix} v(t) \\ 0 \end{bmatrix} \\ &= -S(\omega(t))e(t) + R^T(\theta(t))\dot{p}_r(t) - \begin{bmatrix} v(t) \\ d(t)\omega(t) \end{bmatrix} + \begin{bmatrix} 0 \\ d(t)\omega(t) \end{bmatrix} \\ &= -S(\omega(t))e_1(t) - \Delta(t) \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} + R^T(\theta(t))\dot{p}_r(t). \end{aligned} \quad (2.16)$$

By substituting the control law given in (2.8), the error dynamics are then further simplified to

$$\dot{e}(t) = -S(\omega(t))e_1(t) - K \tanh(e_1(t)) + \dot{\delta}(t). \quad (2.17)$$

The error dynamics given by (2.17) are then substituted into the time derivative of the Lyapunov function,

$$\dot{V}(t) = e_1^T(t)\dot{e}_1(t) + (d(t) - d^*(t))(\dot{d}(t) - \dot{d}^*(t)). \quad (2.18)$$

If we consider the case in which $d(t) \geq \beta$, we then substitute (2.10) and (2.17) into (2.18) to obtain

$$\begin{aligned} \dot{V}(t) &= e_1^T(t) \left(-S(\omega(t))e_1(t) - K \tanh(e_1(t)) \right) - \lambda(d(t) - d^*(t))^2, \\ &= -e_1^T(t)K \tanh(e_1(t)) - \lambda(d(t) - d^*(t))^2. \end{aligned} \quad (2.19)$$

From (2.19), it is trivial to show that again $\dot{V}(t) \leq 0$ for all $t \geq 0$, $d(t) \geq \beta$.

The Lyapunov time derivative is again examined when $d(t) < \beta$. Again, we substitute (2.10) and (2.17) into (2.18) to obtain

$$\begin{aligned} \dot{V}(t) &= -e_1^T(t)K \tanh(e_1(t)) + (d(t) - d^*(t)) \left(-\lambda(d(t) - d^*(t)) + \frac{\beta - d(t)}{d(t) - (\beta - \epsilon)} \right), \\ &= -e_1^T(t)K \tanh(e_1(t)) - \lambda(d(t) - d^*(t))^2 + (d(t) - d^*(t)) \left(\frac{\beta - d(t)}{d(t) - (\beta - \epsilon)} \right). \end{aligned} \quad (2.20)$$

The commanded following distance, $d(t)$, is restricted such that $d(t) > (\beta - \epsilon)$ by design. This behavior is the result of $\frac{\beta - d(t)}{d(t) - (\beta - \epsilon)} \rightarrow \infty$ as $d(t) \rightarrow (\beta - \epsilon)$. As $d(t)$ decreases below β but still remains larger than $(\beta - \epsilon)$, the additional corrective term pushes $d(t)$ back toward β , eventually guaranteeing $\dot{d}(t) > 0$ as $d(t) \rightarrow (\beta - \epsilon)$. As such, the corrective term $\frac{\beta - d(t)}{d(t) - (\beta - \epsilon)} > 0$ for $d(t) < \beta$. Additionally, since $d^*(t) \geq \beta > d(t)$, we find that $(d(t) - d^*(t)) \left(\frac{\beta - d(t)}{d(t) - (\beta - \epsilon)} \right) < 0$. Since the remainder of the Lyapunov time derivative terms are the same as given in (2.19), we can conclude that $\dot{V} \leq 0$ for all $t \geq 0$. \square

2.3 Simulation Results

To evaluate the performance of the controller, we consider the situation in which the reference system is defined as two decoupled first order systems,

$$\dot{p}_r(t) = \begin{bmatrix} -10x_r(t) + 10r_x(t) \\ -10y_r(t) + 10r_y(t) \end{bmatrix}, \quad (2.21)$$

where $r_x(t)$ and $r_y(t)$ are the desired reference trajectories in x and y , respectively. Furthermore, we choose the reference trajectories to be $r_x(t) = 0.5t$ and $r_y(t) = 10 \sin(0.5t)$, which guarantees the reference system to be smooth, more than satisfying the class C^1 condition mentioned earlier.

In addition to the reference system properties, the nominal following distance is chosen to be,

$$d^*(t) = \alpha v_r(t) + \beta, \quad (2.22)$$

where $\alpha > 0$ and $\beta > 0$ are tuning constants. Via the choice of reference trajectories, we ensure that the choice of $d^*(t)$ given in (2.22) also satisfies the class C^1 constraint. Moreover, we note that this choice of nominal distance offers a fairly intuitive relationship for most ground vehicles. For example, with higher reference velocities, we intuitively allow a greater distance to respond to instantaneous changes in reference direction. Likewise, (2.22) draws the vehicle closer to tightly follow slow trajectories.

Table 2.1: Parameters of simulation with $\alpha = 0.5$ and $\beta = 0.1$. All quantities expressed using standard SI units.

Desired Reference Trajectory	Tuning Parameters	Initial Conditions
$r_x(t) = 0.5t$ $r_y(t) = 10 \sin(0.5t)$	$k_v = 1$ $k_\omega = 1$ $\lambda = 1$ $\alpha = 0.5$ $\beta = 0.1$ $\epsilon = 0.05$	$x_r(0) = 0$ $y_r(0) = 0$ $x(0) = -0.1$ $y(0) = 0$ $\theta(0) = 0$ $d(0) = 0.1$

Using the parameters given in Table 2.1 we arrive at the simulation results shown in Figures 2.1 - 2.3. Figure 2.1 shows the theoretical tracking performance of the system. As intended, the vehicle converges tightly to the reference system's path, while maintaining a varying following distance, shown in Figure 2.2. It is interesting to note that the vehicle's largest deviation from the path of the reference system occurs at the peaks of the sinusoid as well as at the beginning of the simulation. In both cases, this has to do with the commanded distance. Although the commanded distance is at its minimum value at the peaks, the vehicle still reverses its trajectory, while turning, to maintain this value. The vehicle then commences a forward velocity once the reference system starts moving away, ultimately moving back to the desired path.

It is fairly trivial to show that the commanded distance seen in Figure 2.2 is the same as the nominal function given in (2.22) with the parameters from Table 2.1. This behavior is

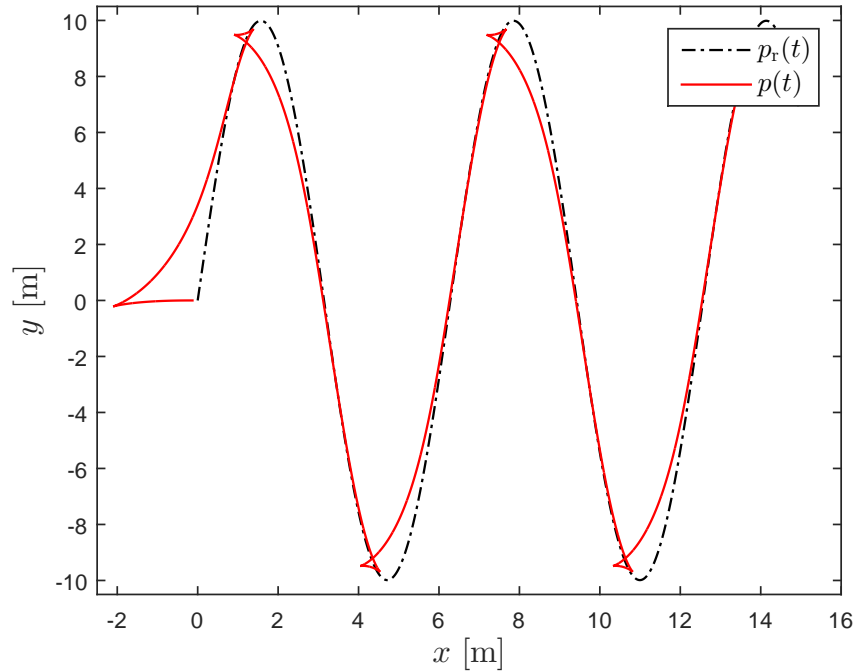


Figure 2.1: Tracking of sinusoidal trajectory with $\alpha = 0.5$ and $\beta = 0.1$.

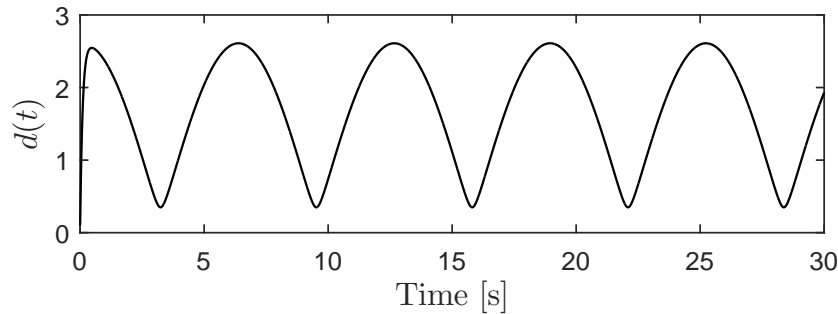


Figure 2.2: Commanded following distance with $\alpha = 0.5$ and $\beta = 0.1$.

simply a result of initializing $d(t)$ such that $d(0) = d^*(0)$. With regard to the controller commands, seen in Figure 2.3, the trends are as expected. The longitudinal velocities oscillates approximately between $-2\frac{\text{m}}{\text{s}}$ and $5\frac{\text{m}}{\text{s}}$, where in this case the negative sign indicates velocity in the reverse direction. As the vehicle approaches the peaks, it slows to a stop then reverses its velocity. From the angular rate command, we can see that it continues to turn in the direction of the reference system, regardless of velocity. Since the controller is based on a unicycle model, there is no obvious correlation between velocity and angular rate commands.

In fact, many of the largest angular rate commands, with regard to magnitude, correspond to the lowest velocity commands. This relationship is exemplified by the third graph which examines the ratio of angular rate to longitudinal velocity. This ratio, which is actually the inverse of the radius of curvature, is part of the motivation behind the saturation algorithms which will be discussed in subsequent chapters. At this point, it is sufficient to note that the ratio is singular at several points along the trajectory. This singularity implies that the vehicle is still turning without actually maintaining a forward velocity, a behavior not achievable by a multitude of ground vehicles.

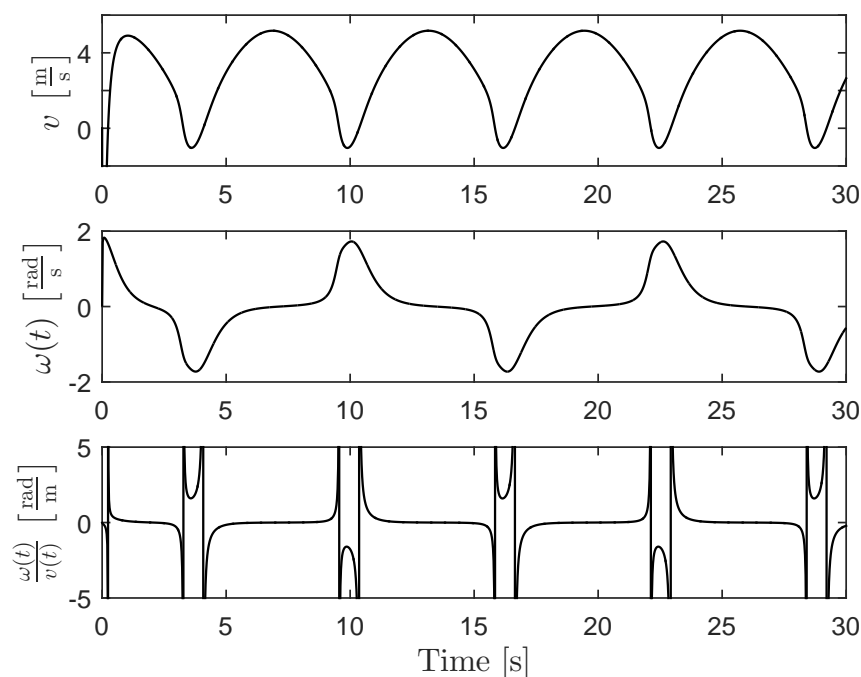


Figure 2.3: Controller commands with $\alpha = 0.5$ and $\beta = 0.1$.

It is interesting to note the impact that the following distance has on the tracking performance. To briefly examine this behavior, we alter the tuning parameter $\alpha = 0.1$. The parameters used in Figures 2.4 - 2.6 are repeated in Table 2.2 for convenience. Looking at Figure 2.4, we see that the tracking performance is dramatically improved. The deviations at the start of the simulation and at the peaks are much more mitigated than those given in Figure 2.1. From Figure 2.5, we see that the main difference is that the magnitude of the commanded following distance is smaller. Apart from this change, the trend is the same.

Overall, the commands shown in Figure 2.6 follow those shown originally in Figure 2.3. The main difference is that the longitudinal velocity never falls below $0 \frac{\text{m}}{\text{s}}$ and the angular rate commands are larger in magnitude. As a result, the ratio of angular rate to longitudinal velocity never becomes singular (apart from the start of the simulation), meaning the radius

Table 2.2: Parameters of simulation with $\alpha = 0.1$ and $\beta = 0.1$. All quantities expressed using standard SI units.

Desired Reference Trajectory	Tuning Parameters	Initial Conditions
$r_x(t) = 0.5t$ $r_y(t) = 10 \sin(0.5t)$	$k_v = 1$ $k_\omega = 1$ $\lambda = 1$ $\alpha = 0.1$ $\beta = 0.1$ $\epsilon = 0.05$	$x_r(0) = 0$ $y_r(0) = 0$ $x(0) = -0.1$ $y(0) = 0$ $\theta(0) = 0$ $d(0) = 0.1$

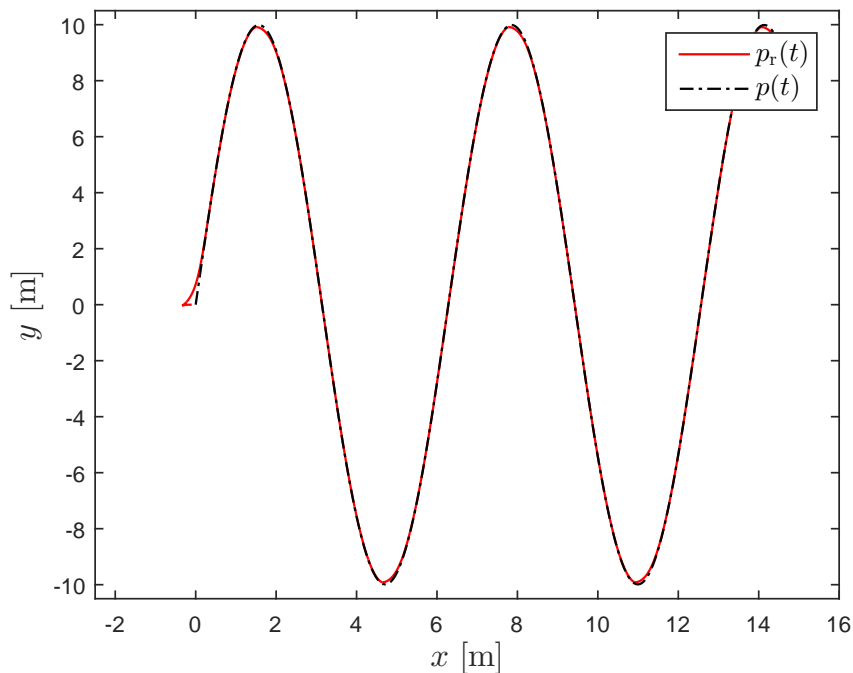


Figure 2.4: Tracking of sinusoidal trajectory with $\alpha = 0.1$ and $\beta = 0.1$.

of curvature is never zero. This observation suggests that the distance function can be chosen so that the control algorithm is more favorable to those platforms not capable of turning independent of longitudinal motion. However, this favorable behavior is still trajectory dependent.

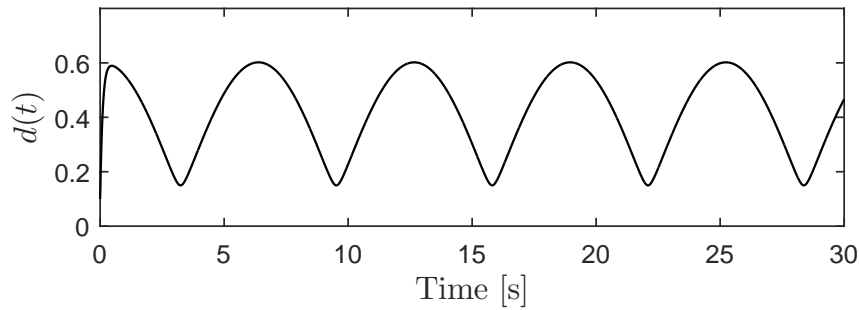


Figure 2.5: Commanded following distance with $\alpha = 0.1$ and $\beta = 0.1$.

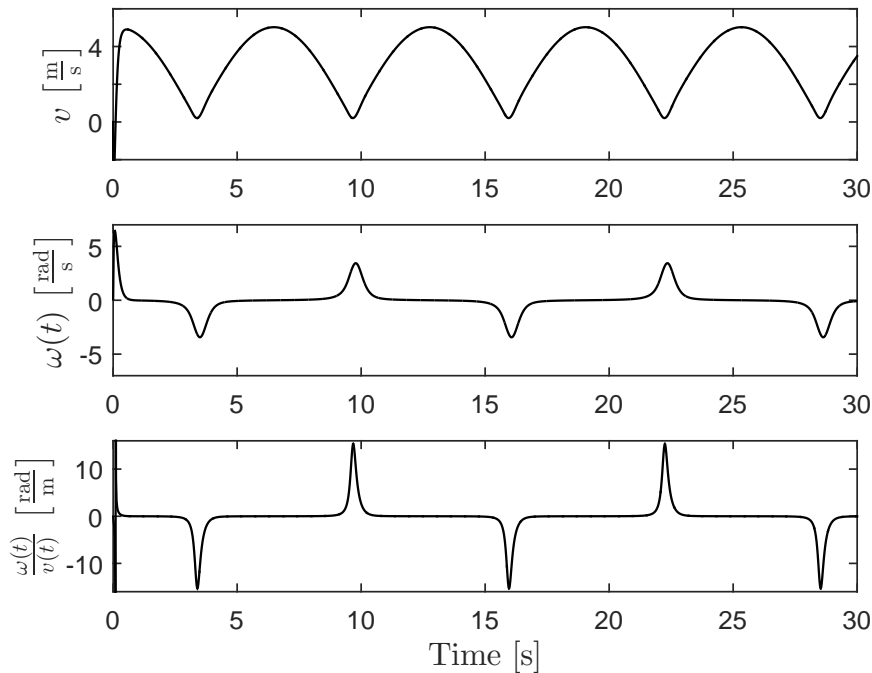


Figure 2.6: Controller commands with $\alpha = 0.1$ and $\beta = 0.1$.

2.4 Summary

In this chapter, we introduce a control algorithm largely developed in [25]. In addition to this algorithm, we place a constraint on the reference system that it must be at least class C^1 in time. This restriction guarantees continuous controller commands and is particularly necessary in the saturation algorithms presented in the next several chapters. Along with constraints on the reference system, the dynamics of the commanded following distance, $d(t)$,

are modified to account for minimum bounds. This modification on the dynamics is used to eliminate numerical issues that may arise during implementation. Using these modifications, simulation results are given to demonstrate the base behavior of the algorithm. As expected, the vehicle behaves as a unicycle with no coupling between longitudinal and angular rates. Additionally, there are no constraints on the magnitude of speed and steering commands which is not typically realistic for the majority of platforms. Lastly in this chapter, we examine the effects of the reference system following criteria. We see that it can be chosen to better suit certain platforms, but while this is the case, it does not guarantee acceptable behavior in all instances.

Chapter 3

Saturation Algorithm for Differential Drive Platforms

In Chapter 2 we discuss the development of the nonholonomic control algorithm. However, we assume nothing about the constraints of the vehicle aside from nonholonomicity which is implicit with the unicycle model. In reality there are other constraints that need to be considered. For example, the velocity and angular rate of a vehicle are typically limited. With the previous control algorithm, if the controller commands a velocity that is not achievable by the vehicle, the Lyapunov criteria are no longer satisfied, and the reference system begins to diverge from the vehicle beyond the commanded following distance.

In this chapter, we introduce saturation constraints to account for this behavior. These constraints are based on the limitations of a differential drive ground platform and will limit the commands to the vehicle. We account for these limitations through Lyapunov stability criteria.

3.1 Differential Drive Kinematics

Differential drive systems are an extraordinarily popular platform used in robotics. These platforms, depicted in Figure 3.1, function by using two independently powered wheels. These wheels are assumed to adhere to a non-slip condition, meaning they are free rolling and have traction at all time. The longitudinal velocities of the left and right wheels, $v_L(t) \in \mathbb{R}$ and $v_r(t) \in \mathbb{R}$ respectively, determine the trajectory of the robot. As it turns out, the wheel velocities have a direct kinematic relationship to longitudinal and angular rates of

the vehicle given as,

$$v(t) = \frac{v_R(t) + v_L(t)}{2}, \quad (3.1)$$

$$\omega(t) = \frac{v_R(t) - v_L(t)}{L_{\text{Diff}}}, \quad (3.2)$$

where $L_{\text{Diff}} > 0$ is the distance separating the center of the two independently driven wheels. It is easy to see from Figure 3.1 that the longitudinal and angular rates of the vehicle directly relate to the radius of curvature, $r(t) \in \mathbb{R}$, about the instantaneous center of curvature, ICC. This relationship is given by,

$$r(t) = \frac{v(t)}{\omega(t)}. \quad (3.3)$$

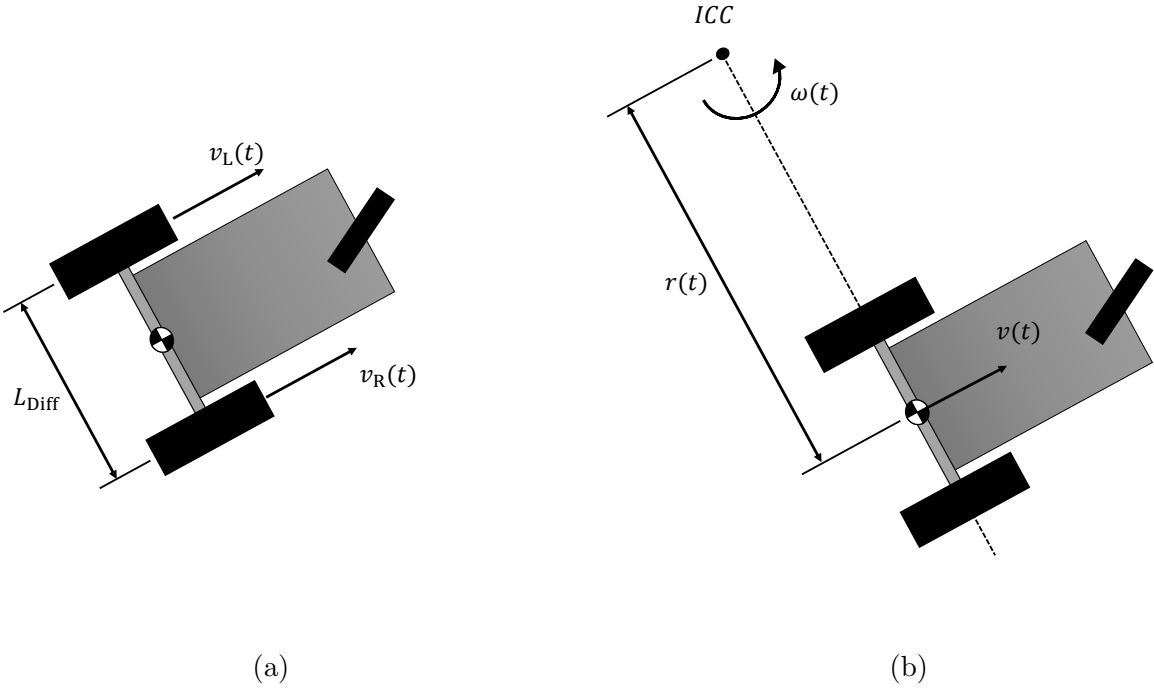


Figure 3.1: Differential platform kinematics.

In practice, we are able to control the wheel velocities directly. By using the relationships given in (3.1) and (3.2), it is possible then to directly control $v(t)$ and $\omega(t)$, which is important given that these are the commands originating from the control algorithm developed in the previous chapter. As mentioned before in the motivation of saturation constraints, the wheel velocities are generally bounded by the physical means of the system, which is a problem not accounted for in the controller development of Chapter 2.

3.2 Differential Drive Constraints

In order to account for the physical limitations of the vehicle, the controller design should be modified such that,

$$-v_{\max} \leq v_L(t) \leq v_{\max}, \quad (3.4)$$

$$-v_{\max} \leq v_R(t) \leq v_{\max}, \quad (3.5)$$

for all $t \geq 0$, where $v_{\max} > 0$ is the physical longitudinal speed limitation of the vehicle's motors. These constraints inherently assume that the wheel speeds have the same magnitude limitation in either the forward or reverse direction and that both drive wheels share the same limitation. While it is possible that these assumptions are not true for some corner cases of vehicle design, most differential drive platforms can be made to function inside these constraints. Considering the effect of these constraints on longitudinal and angular rates, which are the commands given by the controller derived in Section 2.2, we are able to determine a viable command envelope shown in Figure 3.2.

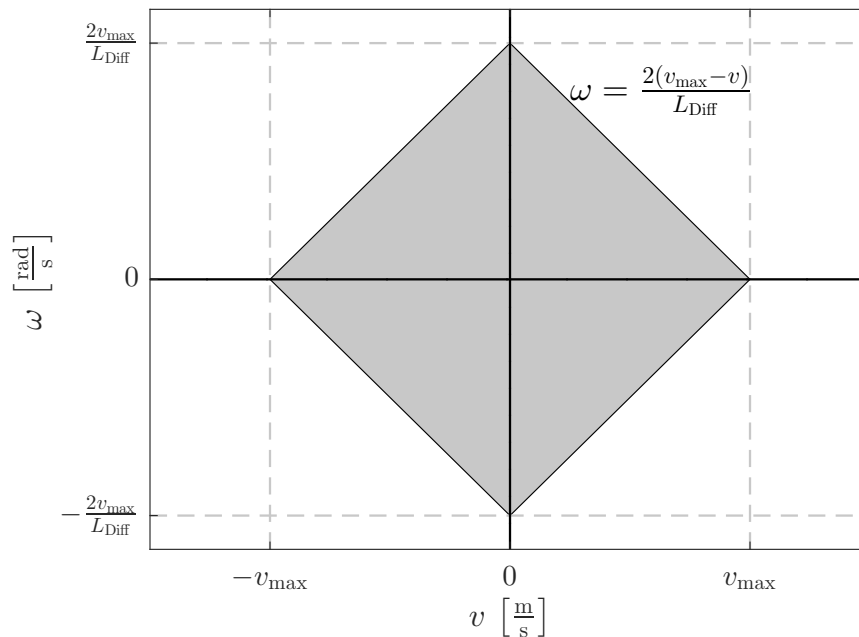


Figure 3.2: Differential drive command envelope (shaded).

The viable envelope is a direct function of motor limitations and vehicle design, v_{\max} and L_{Diff} respectively. As expected, the symmetry of the constraints leads to a symmetric viable

command space about both the v and ω axes. The relationship,

$$\omega = \frac{2(v_{\max} - v)}{L_{\text{Diff}}}, \quad (3.6)$$

for $0 \leq v \leq v_{\max}$, describes the boundary of the envelope in quadrant I and along the positive v and w axes. While there are additional constraints for the remaining quadrants, we will see in the next section that the symmetry will allow us to reflect all considered commands into quadrant I, utilize the saturation algorithm, then reflect the result back into the original quadrant.

3.3 Saturation Algorithm

During operation, the control law given in (2.8) can produce a command anywhere in the $v-\omega$ plane. It is desirable to map these commands back into the envelope depicted in Figure 3.2. There are a variety of saturation behaviors possible. The algorithm could, for example, map commands into the envelope while trying to preserve commanded velocity, or perhaps lateral acceleration. In this case, the saturation algorithm is developed with the intent of preserving the commanded radius of curvature, defined by (3.3), in the hopes of preserving the desired trajectory.

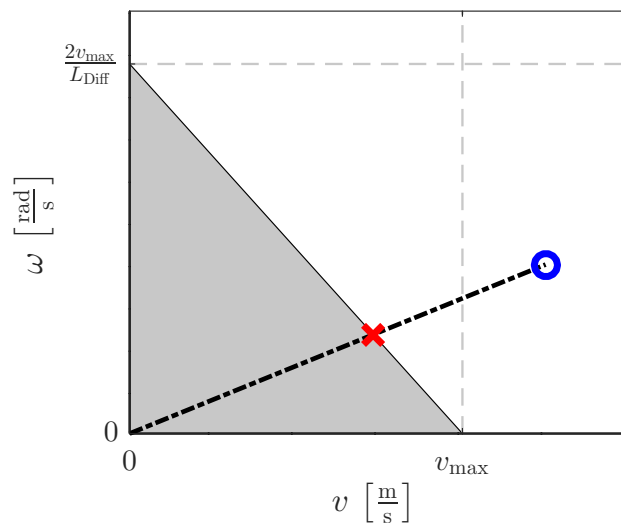


Figure 3.3: Differential drive saturation example.

When the requested command moves outside of the envelope, the saturation algorithm maps the new command to the nearest point on the envelope that provides the same radius of curvature. Figure 3.3 demonstrates this behavior. In this case, the blue circle represents the

original command resulting from (2.8) and the red x represents the command after applying the saturation algorithm. By finding the radius of curvature of the original command according to (3.3) and substituting into (3.6), the saturated command on the envelope border can be found according to,

$$v^{\text{sat}}(t) = \frac{2r(t)v_{\text{max}}}{2r(t) + L_{\text{Diff}}}, \quad (3.7)$$

$$w^{\text{sat}}(t) = \frac{2(v_{\text{max}} - v^{\text{sat}}(t))}{L_{\text{Diff}}}, \quad (3.8)$$

where $v^{\text{sat}}(t) \in \mathbb{R}$ and $w^{\text{sat}}(t) \in \mathbb{R}$ are the saturated velocity and angular rate commands, respectively.

3.3.1 Maintaining Lyapunov Criteria

For the above case, the controller commands are altered without regard to how it affects the tracking performance of the overall system. In order to ensure that the asymptotic stability of the error dynamics is still achieved, we consider the necessary reference system velocity needed to result in the desired saturation commands. By substituting the saturation commands into (2.8) and solving for the reference velocity, we obtain,

$$\dot{p}_r^{\text{sat}}(t) = R(\theta(t)) \left(\Delta(t) \begin{bmatrix} v^{\text{sat}}(t) \\ \omega^{\text{sat}}(t) \end{bmatrix} - K \tanh(e(t) - \delta(t)) + \dot{\delta}(t) \right), \quad (3.9)$$

where $\dot{p}_r^{\text{sat}}(t) \in \mathbb{R}^2$ is the resulting reference time derivative. Since the reference system is entirely user defined, it is acceptable to manipulate it in this manner. The effect is simply that when the reference system affects a command not achievable by the vehicle, the reference system is limited so that the vehicle does not exceed its own capability. Moreover, since the saturation algorithm seeks to maintain the same trajectory as requested, it stands to reason that the saturated reference system will still continue in the desired direction.

With regard to Lyapunov criteria, there is no fundamental relationship change between the saturated commands and reference systems given in (3.9) and the native commands and reference system used in (2.8). As such, it is trivial to show that the Lyapunov criteria for (2.11) are still satisfied using the same procedures given in Section 2.2.

It is important to realize that $d^*(t)$ must be at least a class C^1 function in time to guarantee that both $\dot{d}^*(t)$ exists and also that the control law is continuous. With our choice of $d^*(t)$ in (2.22), this condition was originally met because of the choice in reference system. Since the reference system is no longer guaranteed to be class C^1 in time because of the saturation algorithm, an alternative choice to (2.22) must be used. To elicit this behavior, we modify the nominal following distance to be

$$\ddot{d}^*(t) + 2\zeta_d\omega_d\dot{d}^*(t) + \omega_d^2d^*(t) = \omega_d^2d_{\text{ref}}(t), \quad (3.10)$$

where $\zeta_d > 0$, $\omega_d > 0$ are tuning constants, and $d_{\text{ref}}(t)$ is user desired reference distance.

3.4 Differential Drive Simulation Results

Prior to the discussion of saturation, the nominal following distance, $d^*(t)$, was chosen to follow the behavior (2.22). In order to maintain similar behavior while adhering to new constraints, we chose the reference distance to be,

$$d_{\text{ref}}(t) = \alpha v_r(t) + \beta. \quad (3.11)$$

For the purpose of simulation, the reference behavior given in (2.21) is maintained with the exception that the dynamics adhere to (3.9) during times of saturation. The parameters of the first simulation presented in this section can be found in Table 3.1. In this simulation, the saturation algorithm is triggered only at the beginning when the controller places a relatively high demand on the vehicle to get to an acceptable position, but otherwise, the controller requests efforts well within reason of the given saturation constraints. The intent in this case is to evaluate the basic behavioral changes caused by the addition of the saturation algorithm without regard to the saturation itself.

Table 3.1: Differential platform simulation parameters with $v_{\text{max}} = 5$ and $L_{\text{Diff}} = 0.915$. All quantities expressed using standard SI units.

Desired Reference Trajectory	Saturation Constraints	Tuning Parameters	Initial Conditions
$r_x(t) = 0.5t$ $r_y(t) = 10 \sin(0.5t)$	$v_{\text{max}} = 5$ $L_{\text{Diff}} = 0.915$	$k_v = 1$ $k_\omega = 1$ $\lambda = 1$ $\alpha = 0.5$ $\beta = 0.1$ $\epsilon = 0.05$ $\zeta_d = 0.85$ $\omega_d = 2.5$	$x_r(0) = 0$ $y_r(0) = 0$ $x(0) = -0.1$ $y(0) = 0$ $\theta(0) = 0$ $d(0) = 0.1$

The response to the parameters given in Table 3.1 is given in Figure 3.4. With regard to intent, there should be virtually no difference between the response pictured in Figure 3.4 and the simulation results seen earlier in Figure 2.1. The two simulations, however, have obvious differences. In the first simulation without saturation constraints, the vehicle originally underwent a relatively large magnitude negative velocity as it backed away from its starting location and steered toward the reference system. In the case of the new response, the vehicle does not have this same inclination. Instead, we see from the commands given in Figure 3.6 that it commands a forward velocity while turning sharply toward the reference system. The major contributor to this change in behavior is the additional dynamics placed on the follow distance in (3.10).

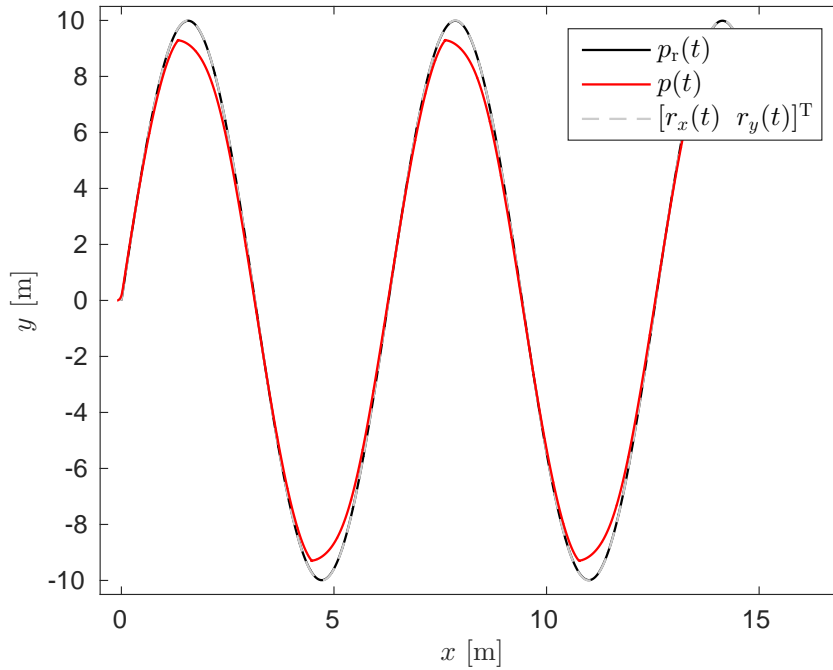


Figure 3.4: Differential platform tracking of sinusoidal trajectory with $v_{\max} = 5$ and $L_{\text{Diff}} = 0.915$.

In the simulation without saturation constraints, the vehicle immediately reversed largely from the distance time derivative, $\dot{\delta}(t)$, contribution to the control law given in (2.8). This observation is corroborated by the following distance comparison shown in Figure 3.5 which gives the non-saturated following distance shown in red alongside the saturated following distance for the current simulation shown in black. Considering the non-saturated following distance at the beginning of the simulation, we see that the magnitude changes relatively quickly which affects a large, positive $\dot{\delta}(t)$, which directly contributes to a negative velocity command. On the other hand, the follow distance resulting from the new saturation algorithm has a milder increase at the start due to the new dynamics. As a result, the vehicle does not experience the quick negative velocity command but is instead more docile, which as it turns out tends to be a favorable trait when dealing with real platforms.

Another notable difference between the simulation shown in Figures 3.4 - 3.6 and the simulation of Figures 2.1 - 2.3 is seen as the vehicle rounds the turns at the peaks of the sinusoid. In the case of the non-saturated simulation, the vehicle stops and backs up before continuing on its trajectory, whereas in the case of saturation, it is more of a stationary turning rather than a noticeable negative velocity command. This behavior is further demonstrated by the commands pictured in Figure 3.6. Again, this is an effect of the added distance dynamics. In the case of no saturation, the distance increases sharply as the reference system rounds

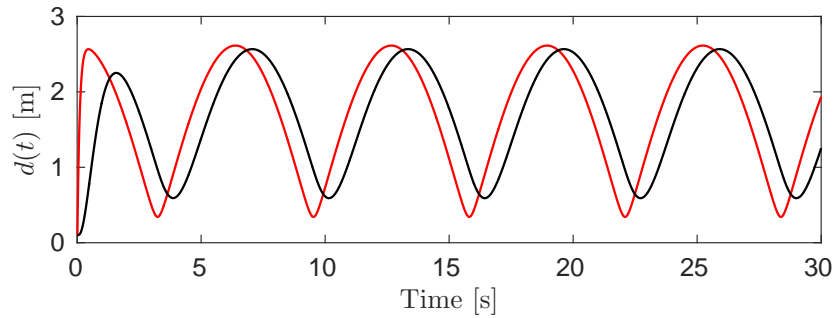


Figure 3.5: Differential platform commanded following distance with saturation (black) compared to no saturation (red).

the peak and begins to increase velocity. This behavior then induces a negative velocity contribution, causing the vehicle to reverse while turning toward the reference system. In the case of the newer saturation dynamics, the response is somewhat less extreme, simply turning into the reference rather than undergoing an additional negative velocity.

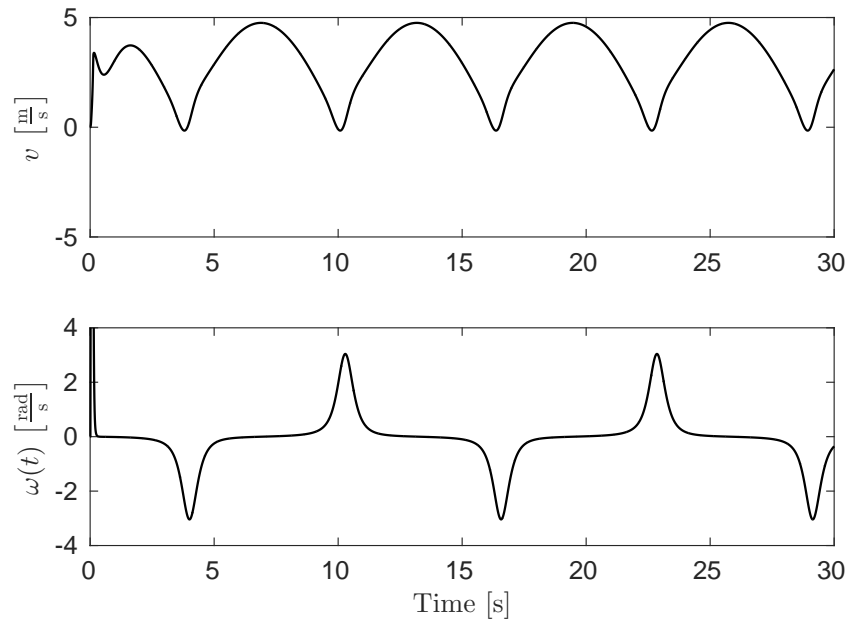


Figure 3.6: Differential platform controller commands with $v_{\max} = 5$ and $L_{\text{Diff}} = 0.915$.

At first glance, it appears that the main behavior change from the addition of the saturation algorithm has been a mitigating effect on controller commands and response. In addition to the obvious and expected phase shift on the following distance, we see from the preceding discussion that some of the more extreme actions such as the immediate negative velocity at

the start of simulation and negative velocities in the turns have been canceled out. Generally, the saturation algorithm has created a more favorable behavior that has less extreme actions and tends to adhere tighter to the reference trajectory.

From the previous simulation, we see there are some behaviors that the saturation algorithm modifies, and in general, the modifications are favorable. The main motivation, however, for this modification is to handle cases in which the vehicle can not physically perform past a certain limitation that is typically dictated by a particular set of platform dependent constraints. In order to simulate this behavior, we introduce the parameters given in Table 3.2. We note that all parameters of the simulation are kept consistent with the previous simulation with the exception of the saturation constraints, $v_{\max} = 2 \frac{\text{m}}{\text{s}}$ and $L_{\text{Diff}} = 0.915\text{m}$.

Table 3.2: Differential platform simulation parameters with $v_{\max} = 2$ and $L_{\text{Diff}} = 0.915$. All quantities expressed using standard SI units.

Desired Reference Trajectory	Saturation Constraints	Tuning Parameters	Initial Conditions
$r_x(t) = 0.5t$ $r_y(t) = 10 \sin(0.5t)$	$v_{\max} = 2$ $L_{\text{Diff}} = 0.915$	$k_v = 1$ $k_\omega = 1$ $\lambda = 1$ $\alpha = 0.5$ $\beta = 0.1$ $\epsilon = 0.05$ $\zeta_d = 0.85$ $\omega_d = 2.5$	$x_r(0) = 0$ $y_r(0) = 0$ $x(0) = -0.1$ $y(0) = 0$ $\theta(0) = 0$ $d(0) = 0.1$

Figure 3.7 shows the resulting trajectory using the parameters given in Table 3.2. The most obvious difference lies in the fact that the simulated vehicle is not nearly as close to the desired trajectory as previous simulations. In fact, the vehicle only achieves approximately 60% of the amplitude in the y direction in comparison to the desired trajectory, causing serious deviations. This relatively poor behavior, however, is largely what is expected based on the modifications made in the earlier sections. Based on the previous simulation, the ideal reference trajectory should be commanding speeds of approximately $5 \frac{\text{m}}{\text{s}}$. Since the speed was limited to less than half of this quantity, there is no feasible way that the vehicle could fully keep up with the reference trajectory.

While the vehicle is incapable of following the ideal path, there are several favorable traits worth noting. First, we note that the overall path still follows the same general trend as the desired reference trajectory. In general, the reference dynamics are chosen to always be pulled toward the reference trajectory. We note, however, the unique path taken by the reference system which is clearly limited by the vehicle capability. The reference path instead, ends up as a tradeoff between desired trajectory and what the vehicle can actually

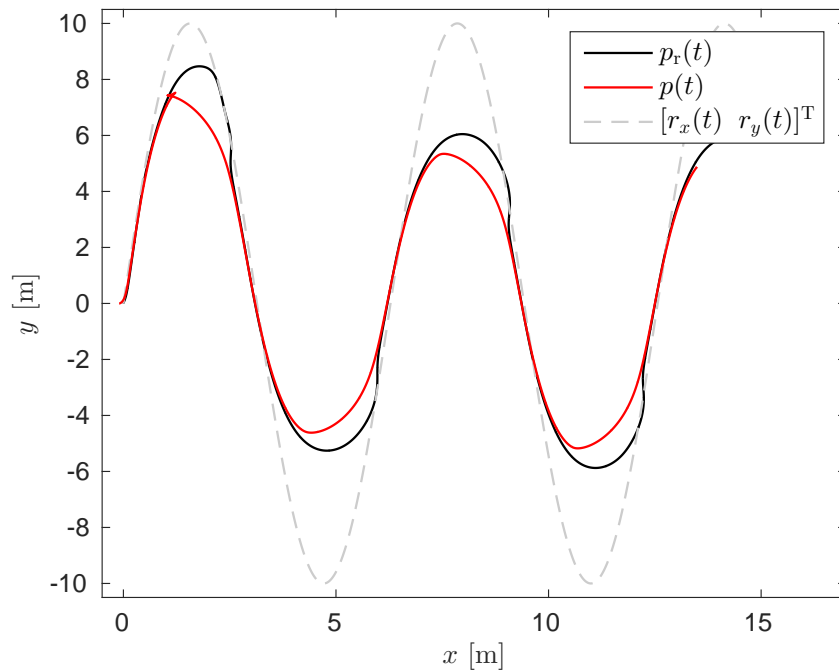


Figure 3.7: Differential platform tracking of sinusoidal trajectory with $v_{\max} = 2$ and $L_{\text{Diff}} = 0.915$.

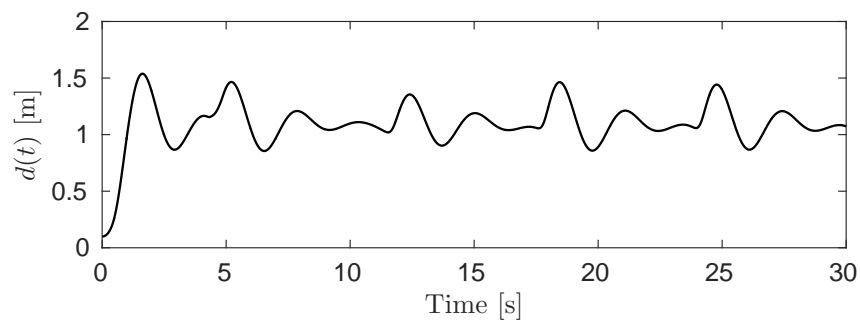


Figure 3.8: Differential platform commanded following distance with $v_{\max} = 2$ and $L_{\text{Diff}} = 0.915$.

accomplish. For our purposes, this trait is favorable since during saturation the best we can hope for is that the system behaves as closely to what is desired as possible.

Examining the following distance given in Figure 3.8, we see that it is somewhat irregular, which is directly related to the irregularities of that reference system. Although irregular, the variation of the following distance is relatively small in comparison to previous simulations.

Since the vehicle velocity is limited and the system is left trying to constantly catch up to the desired trajectory, it would make sense that most of the vehicle's time, and by association the reference system's time, is spent at maximum velocity. As a corollary to this statement, the following distance would have minor deviations due to minor speed changes, which is the behavior observed in Figure 3.8.

Looking at the commands given in Figure 3.9, we see that they are continuous, which was one of the criteria set forth in the saturation algorithm design. Moreover, we note the obvious bounds on velocity, which is the main driving factor in the saturation parameters set forth.

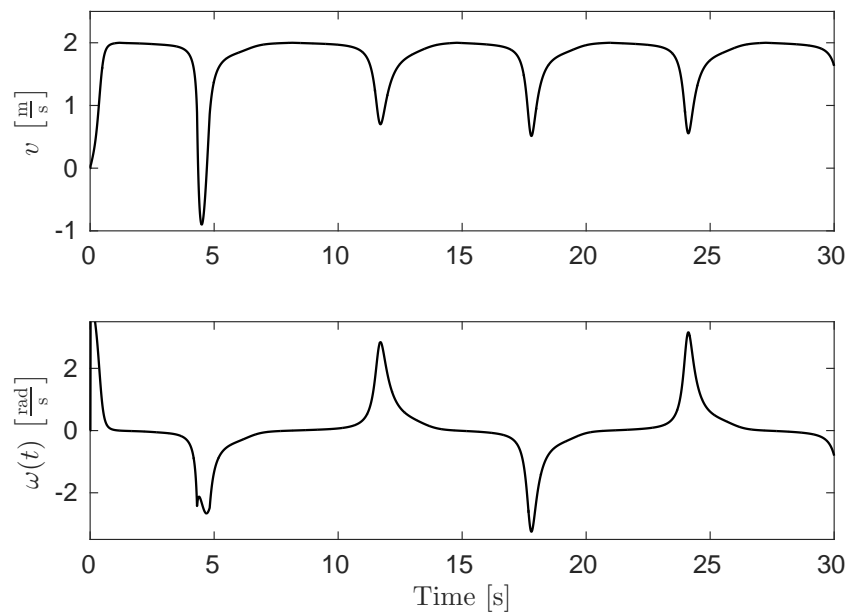


Figure 3.9: Differential platform controller commands with $v_{\max} = 2$ and $L_{\text{Diff}} = 0.915$.

3.5 Summary

In this chapter, saturation constraints are introduced into the control algorithm for differential drive based platforms. To develop these constraints, we introduce a basic kinematic model for a differential drive vehicle. Using this model, constraints are developed that relate to vehicle actuator capabilities. These constraints are simplified with the assumption that typical motors have the same limiting speed in both forward and reverse directions. While this assumption is not always true, it is reasonable and can be made to work for a wide variety of applications. From the saturation constraints, an algorithm is developed which focuses on modifying commands to maintain curvature, with the intent that the vehicle will

follow the desired trajectory to the best of its capabilities during saturation. To facilitate this algorithm, methodologies for maintaining Lyapunov stability criteria are introduced that alter the reference system velocity. Lastly, simulations are conducted using the modified algorithm in both states of normal operation and saturated conditions to demonstrate that the algorithm performs as expected.

Chapter 4

Experimental Results for Differential Drive Platforms

In Chapter 3, we developed the methodology to incorporate vehicle capabilities into the controller developed in Chapter 2 for differentially driven platforms. In this chapter, we will discuss the experimental results obtained by applying this controller to a differentially driven small scale UGV. Before introducing and discussing these results, we will first give an overview of the experimental setup used in these tests. Moreover, additional algorithms based on potential fields are introduced that allow the controller to be applied as a way-point navigation and obstacle avoidance algorithm, increasing its practicality for testing and additional applications.

4.1 Experimental Setup

As part of the available resources in the Center for Dynamic Systems Modeling and Control (DySMAC), there are four UGVs referred to as the Terrestrial Unmanned Robots for Teamed Learning and Exploration (TURTLES), pictured in Figure 4.1. These vehicles, which were designed for both indoor and outdoor capability have a myriad of features that made them ideal for testing. The base design includes a differentially steered setup with two brushless motors located in the rear. Each motor can continually produce 139 lbf-in of torque with peaks up to 257 lbf-in. This corresponds to a continuous draw of over 500 W per motor and speeds of up to 10 mph. The motors are controlled via a RoboteQ brushless controller that allows for RS-232, USB, and PWM signals to modulate speed. The motors and controller are powered by 8 LiFePO4 battery cells in series. These cells have a capacity of 100 A-hr.

With regard to instrumentation and other onboard hardware, the batteries are also interfaced with an onboard voltage regulator that distributes 3.3V, 5V, and 12V to the computers and peripheral devices. This power supply, currently serves to power an onboard computer,



Figure 4.1: Differential drive test platforms (TURTLEs).

network router, GPS/INS system, and 5 GHz radio. The onboard computer is a Pandora Mini PC manufactured by Cappuccino PC. Each unit has an i-7 Intel processor, 80 GB SSD, 8 GB of RAM, and runs a Windows 7 environment.

The 5 GHz radio is part of the Rocket M series manufactured by Ubiquiti Networks. It acts as a wireless Ethernet bridge and has a throughput of 150+ Mbps. Moreover, each radio has a range of several miles allowing for efficient and wide scale implementation. The GPS/INS system is the SPAN-CPT manufactured by NovAtel. This unit is rated to an accuracy of ± 1.5 m and, in practice, is generally accurate to less than ± 1 m. Moreover, the units can be linked to the base station for RTK corrections, allowing for accuracies of ± 2 cm. Heading accuracy falls within $\pm 0.5^\circ$. The GPS/INS communicates over USB or RS-232 and can relay position and IMU data up to 20 Hz and 100 Hz, respectively.

For the purpose of this series of experiments, all controller code and robot manipulation algorithms were written in LabVIEW and run directly on the vehicle. Additionally, all data collected was stored on the vehicle during collection. The controller and data input rates were set at 20 Hz. No RTK corrections were used during testing and no additional filtering or processing was applied to data from the GPS/INS unit. Additionally, two low level PI controllers were used to control velocity and angular rate so that the controller could be implemented as discussed in this document.

4.2 Potential Field Reference System Design

For the purpose of experimental testing, the reference system was manipulated according to a simple potential field algorithm similar to that presented in [17, 18, 19]. This method was chosen primarily because it offers a computationally simple manner to control the reference system that is easily extendable to waypoint navigation and obstacle avoidance. It also provides an intuitive velocity behavior, which is characterized by reducing velocity when obstacles are encountered.

To define the potential field algorithm, we begin by considering the reference system to have the following dynamics,

$$m\ddot{p}_r(t) + c\dot{p}_r(t) = F_a + F_r, \quad (4.1)$$

where $m > 0$, and $c > 0$ are tuning constants for inertia and damping of the virtual system, respectively. The virtual forces $F_a \in \mathbb{R}^2$ and $F_r \in \mathbb{R}^2$ are the contributions from goals and obstacles, respectively, that ultimately drive the motion of the reference system.

To begin understanding how the virtual forces interact with the system, we introduce the distance between the reference system and the i th destination,

$$d_{\text{des},i} \triangleq \sqrt{(x_{\text{des},i} - x_r(t))^2 + (y_{\text{des},i} - y_r(t))^2}, \quad (4.2)$$

where $d_{\text{des},i} \in \mathbb{R}$, $x_{\text{des},i} \in \mathbb{R}$, is the x coordinate of the i^{th} destination waypoint, and $y_{\text{des},i} \in \mathbb{R}$, is the y coordinate of the i^{th} destination waypoint. Furthermore, the distance between the reference system and surrounding j th obstacle is defined as,

$$d_{\text{obs},j} \triangleq \sqrt{(x_{\text{obs},j} - x_r(t))^2 + (y_{\text{obs},j} - y_r(t))^2}, \quad (4.3)$$

where $d_{\text{obs},j} \in \mathbb{R}$, $x_{\text{obs},j} \in \mathbb{R}$, is the x coordinate of the j^{th} obstacle location, and $y_{\text{obs},j} \in \mathbb{R}$, is the y coordinate of the j^{th} obstacle location. Using these distance parameters, the virtual force vectors used in (4.1) are given as follows,

$$F_a \triangleq \frac{F_{\text{ac}}}{d_{\text{des},i}} \begin{bmatrix} x_{\text{des},i} - x_r(t) \\ y_{\text{des},i} - y_r(t) \end{bmatrix}, \quad (4.4)$$

$$F_r \triangleq \sum_{j=1}^{N_{\text{obs}}} -F_{\text{rc}} \left(\frac{W}{d_{\text{obs},j}} \right)^n \begin{bmatrix} x_{\text{obs},j} - x_r(t) \\ y_{\text{obs},j} - y_r(t) \end{bmatrix}, \quad (4.5)$$

where $F_{\text{ac}} > 0$, $F_{\text{rc}} > 0$, $W > 0$, and $n > 1$ are tuning constants and N_{obs} is the number of obstacles within a given radius of the reference system.

From (4.4), we can see that the attractive force, F_a , is essentially a unit vector multiplied by the tuning constant F_{ac} . This vector points from the reference system toward the current

destination at all times. Because of this relationship, the reference system will be pulled toward the goal with some constant force dictated by F_{ac} .

Likewise, we note that (4.5) has a similar form to (4.4). This results in a force vector, F_r , that points from a particular obstacle toward the reference system, repelling it. Because of the exponential relationship caused by n , the repulsive force increases as the reference system nears the obstacle. Consequently, the reference system is largely uninfluenced until it is within the vicinity of an obstacle, at which point it is pushed away. Ultimately, the combination of the attractive and repulsive forces, F_a and F_r respectively, causes the reference system to avoid any obstacles while moving toward the goal.

With the force vectors defined, it is still desirable to find an intuitive manner of choosing both m and c such that the reference system behaves as requested. To define these constants, we start by considering the maximum desired velocity in the presence of only attractive forces. When maximum velocity of the reference system is achieved, the reference acceleration reduces to $\ddot{p}_r(t) = 0$, which yields,

$$c\dot{p}_r(t) = F_a. \quad (4.6)$$

By taking the norm of both sides and considering the maximum reference velocity case we can determine the damping as,

$$c = \frac{F_{ac}}{v_{rmax}}. \quad (4.7)$$

In regard to determining the inertia, we assume the reference system persists at maximum velocity throughout the trajectory. If the kinetic energy is maintained along the path, then the path maintains the same characteristic shape, regardless of velocity. As such, the inertial term is defined as,

$$m = \frac{2E}{v_{rmax}^2}, \quad (4.8)$$

where $E > 0$, is the user defined kinetic energy term of the reference system. By using kinetic energy and maximum reference velocity to determine the tuning characteristics of the system, the behavior becomes much more intuitive. In order to make the vehicle go faster, the velocity is now directly adjustable. Likewise, if the desire is to smooth the reference path, this task can be accomplished by increasing the kinetic energy of the reference system.

The effect of the potential fields is not considered beyond the reference system as the controller is designed to drive the vehicle to the reference system, given that the reference system is composed of class C^1 functions. To better understand the effects of potential fields, the reader is referred to [17, 18, 19], which better explore their behavior and usage.

4.3 Tracking Performance

Using the potential field algorithm described in the previous section, the experimental tracking capabilities of this controller were tested on one of the differential platforms described in Section 6.1 using the parameters laid out in Table 4.1. As seen in the following table, all experimental parameters are kept the same with the exception of the reference velocity, $v_{r_{\max}}$, which is altered to understand the effect of speed on the algorithm. We note at this point that while the vehicle’s motors are capable of much higher speeds, the vehicle itself begins to experience heavy vibrations from its front casters after a certain speed, prohibiting higher speed tests. These effects are even moderately visible in the data of the last experiment, which is presented in the following discussion.

Table 4.1: Differential platform experimental tracking parameters. All quantities expressed using standard SI units.

Saturation Constraints	Tuning Parameters	Potential Field Parameters
$v_{\max} = 2.5$ $L_{\text{Diff}} = 0.915$	$k_{\omega} = 1$ $k_v = 1$ $\lambda = 1$ $\alpha = 0.5$ $\beta = 1.5$ $\epsilon = 0.75$ $\zeta_d = 0.85$ $\omega_d = 2$	$v_{r_{\max}} = 0.5, 1, 1.5$ $F_{\text{ac}} = 10$ $F_{\text{rc}} = 6$ $W = 2$ $n = 1.5$ $E = 14$

Figure 4.2 demonstrates the tracking performance of the differential platform at the lowest reference speed, $v_{r_{\max}} = 0.5 \frac{\text{m}}{\text{s}}$. In this figure, the vehicle starts at the top of the figure in the center and moves clockwise around the waypoints following the reference system. For the most part, it appears that the vehicle trajectory and reference trajectory overlap well, suggesting good tracking performance. There are some notable deviations of the vehicle from the path near the waypoints located at approximately $(10, -26)$ and $(-4, 5)$. This deviation, however, is a natural response of the controller, which allows the vehicle to correct its position as the reference system in front of it begins a turn. As such, it is normal, and in fact encouraging, to see the vehicle diverge from the reference system in turns as this behavior is expected based on the design.

To better assess the performance of the controller, we look to Figure 4.3. Recall from the design that the control law is designed to guarantee that the longitudinal error converges to the commanded following distance, $d(t)$, while the lateral error converges to zero such that such that $\|e(t) - \delta(t)\| \rightarrow 0$ as $t \rightarrow \infty$. This relationship during the experiment is depicted

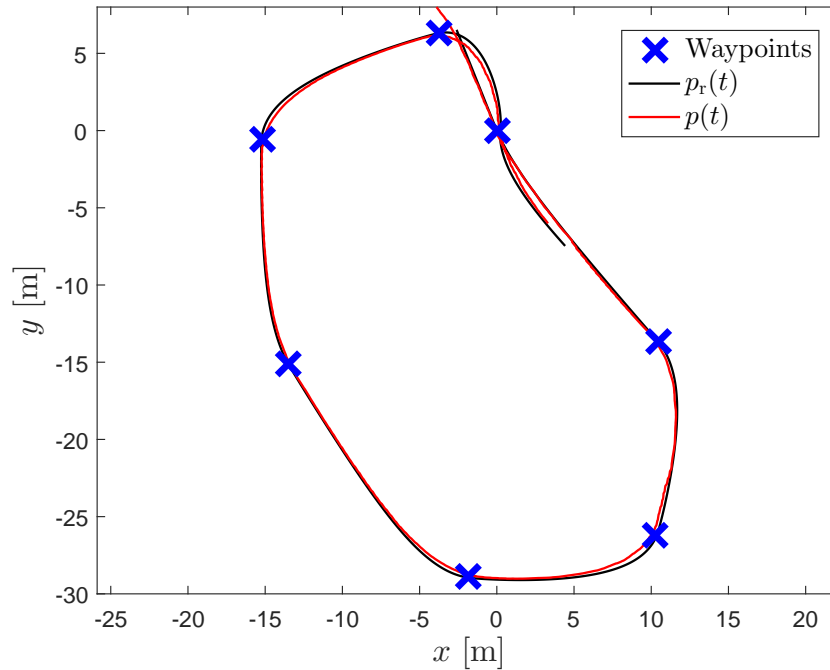


Figure 4.2: Tracking performance of waypoints at $v_{r_{\max}} = 0.5$.

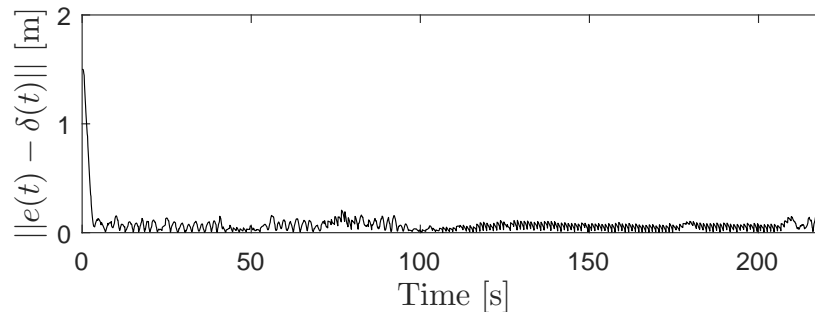


Figure 4.3: Total error, $e_1(t)$, at $v_{r_{\max}} = 0.5$.

in Figure 4.3. From this figure, we see that the tracking performance is highly encouraging. The original error of the system starts at approximately 1.5 m from its intended location and quickly converges to within 0.2 m. This value, in turn, suggests that for the majority of the experiment, the vehicle was no further away from its intended location than 20 cm.

Keeping in mind, the error metric presented intuitively includes lateral error which is influenced by heading. Looking at the command and vehicle response graphs in Figure 4.4, we see that lateral error was likely a source of the majority of the error. While the magnitudes

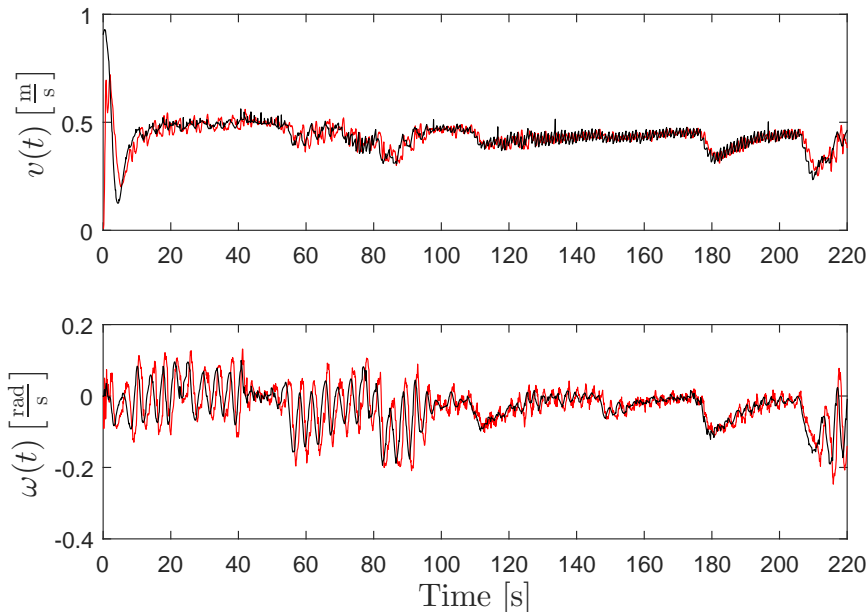


Figure 4.4: Controller commands (black) and measured vehicle response (red) at $v_{\text{rmax}} = 0.5$.

are small, we see that the angular rate commands are highly oscillatory, much more than would be necessary for the path of waypoints given. Conversely, the commands and response for the velocity of the vehicle are relatively constant, suggesting the longitudinal error was generally at the value desired. The behavior of the angular rate commands is the first indication we see in the experiments in which the dynamics of the vehicle effected the performance of the control law which does not otherwise account for the system behind basic kinematics. The source of this oscillation was due to the casters which take a disproportionate effort to initiate turning the the vehicle than is necessary to sustain turning. As a result, the vehicle has a tendency to overshoot target headings when the commands for angular rate and velocity are relatively small, causing the response seen here. Still, the performance of the controller is encouraging in light of these unaccounted for effects.

To further explore the capabilities of the controller, we examine the response of the vehicle to a reference system that is moving at $1 \frac{\text{m}}{\text{s}}$. All parameters are consistent with those in the previous example, with the exception of $v_{\text{rmax}} = 1 \frac{\text{m}}{\text{s}}$. The tracking performance of this test is presented in Figure 4.5. With the exception of initial conditions, we see that the performance between this experiment and that presented in Figure 4.2 is largely unchanged. While arguably redundant, this result is encouraging.

We further explore the efficacy of the controller in Figure 4.6 by looking at the overall error metric for the system. As with the previous experiment, the overall error starts at approximately 1.5 m and converges to a near zero value. In this experiment, the convergence

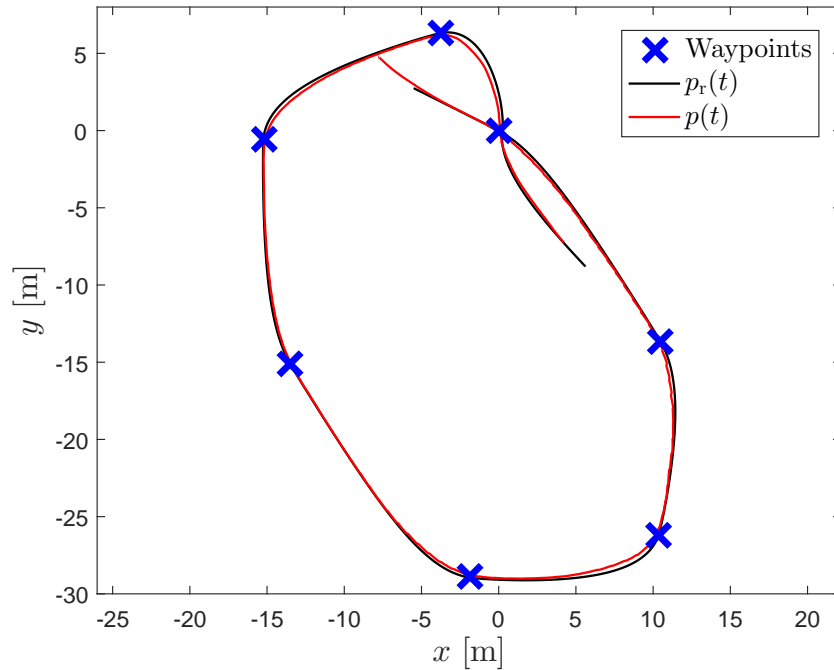


Figure 4.5: Tracking performance of waypoints at $v_{r_{\max}} = 1$.

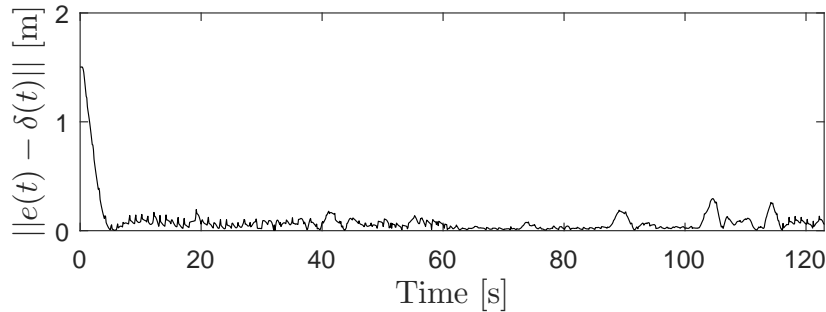


Figure 4.6: Total error, $e_1(t)$, at $v_{r_{\max}} = 1$.

appears somewhat slower than previously presented. Looking at Figure 4.7, the velocity response to the commanded input was slower than before, suggesting that the error early in the experiment is based predominately on longitudinal distance. Throughout the remainder of the experiment, we note that the error experiences several minor rises. These are attributed largely to points in which the reference is in transition between waypoints which causes a velocity reduction and introduces lateral error. Since the vehicle dynamics are not immediate, as they are in simulation, this leads to a natural mild error increase.

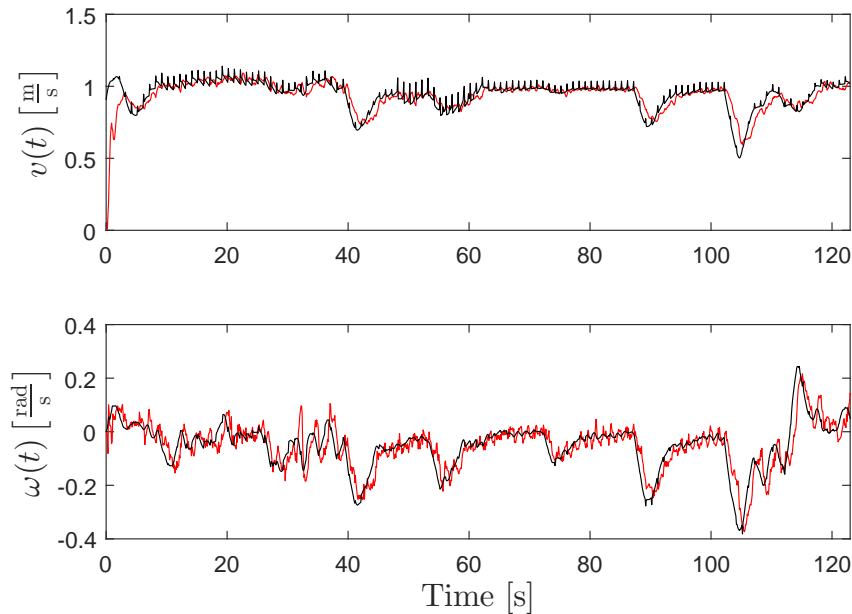


Figure 4.7: Controller commands (black) and measured vehicle response (red) at $v_{r_{\max}} = 1$.

Looking at the commands and vehicle response in Figure 4.7, the main difference between this experiment from the previous is the lack of oscillation in the angular rate command. The inability to affect precise steering at the lower speed setting is actually somewhat mitigated at this slightly higher speed. As a result, the settings of this experiment are actually somewhat closer to ideal as opposed to the previous example.

Using the parameters laid out in Table 4.1, the last experiment explores the behavior of the algorithm with a reference velocity of $v_{r_{\max}} = 1.5 \frac{\text{m}}{\text{s}}$. Figure 4.8 shows the trajectory of the vehicle and reference system at this point. Overall, the response is mostly unchanged from the previous experiments at lower speed settings. Apart from initial conditions, the most notable difference at this setting is that the vehicle tends to move slightly more toward the inside of the path of the reference system than before. This trend is expected, and encouraged, based on the design of the controller and choice of following distance which as the speed increases allows the vehicle more distance and response time to correct its trajectory.

Examining Figure 4.9, we see that this experiment continues the trend of a longer convergence time at higher speeds. Taking nearly 5 seconds to converge, we see a direct correspondence with the rise time of the velocity response given in Figure 4.10. This correspondence corroborates that the majority of the initial error is following distance dependent rather than lateral difference. Looking at Figure 4.9 further, we see that although the error is reasonable at all points in the experiment it has much larger disturbances than seen previously. For

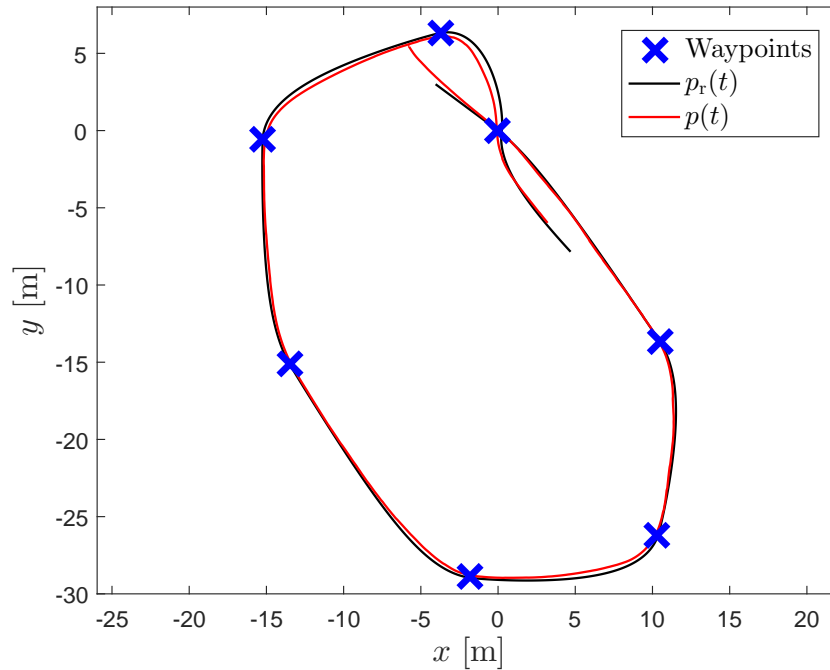


Figure 4.8: Tracking performance of waypoints at $v_{r\max} = 1.5$.

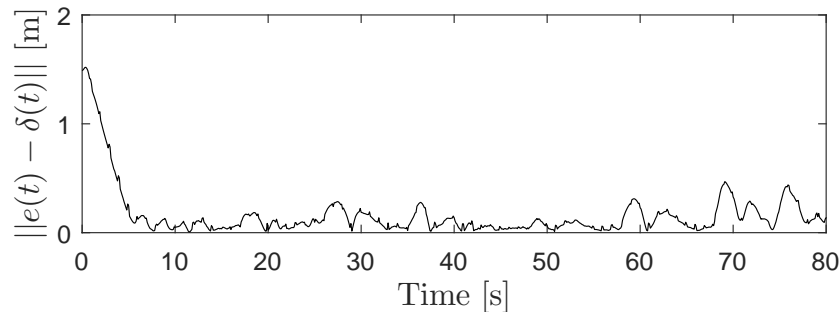


Figure 4.9: Total error, $e_1(t)$, at $v_{r\max} = 1.5$.

example, the deviations in the last 15 seconds of the experiment are nearly twice that of the previous. In this case time period, the vehicle is passing through the last two points in the experiment which are characterized by demanding turns. As a result, we see the angular rate and velocity commands vary largely at this time as well in Figure 4.10. Moreover, the response has a notable lag from the command itself. As such, this lag induces an error that is not accounted for in design and as such shows up in Figure 4.9.

As discussed at the beginning of this section, the vehicle's capability for testing is currently

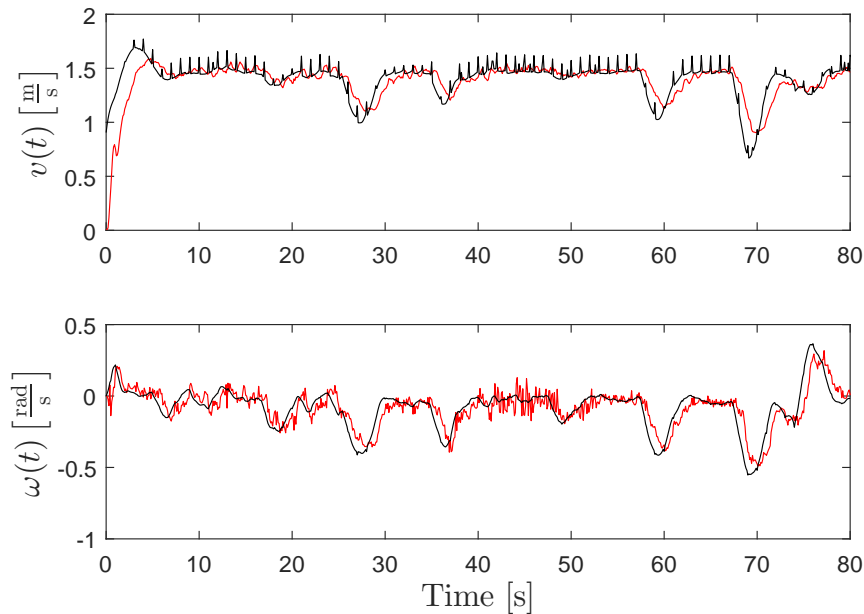


Figure 4.10: Controller commands (black) and measured vehicle response (red) at $v_{\text{rmax}} = 1.5$.

limited in speed by a vibration of the caster wheels. As a side note, this behavior was observed, albeit mildly, in this experiment. In Figure 4.10 in the time period from $t = 40$ to $t = 50$ seconds we see a large amount of noise in the angular rate response. During this period, the caster developed a relatively mild wobble that is obvious from the response. In practice, at speeds higher than this, the effect is increased incrementally until the point that the vehicle is uncontrollable. No experiments beyond this point are included due to the fact that external disturbances are such that the other onboard sensing equipment becomes unreliable, effect the efficacy of the test. Still, in the experiments presented, the control algorithm is shown to be highly effective and capable of controlling the vehicle's position to within an efficient degree of accuracy.

4.4 Summary

In this chapter we present the experimental results obtained using the control algorithm and saturation methods developed in Chapter 2 and Chapter 3, respectively. To conduct these tests, differential steering platforms hosted by DySMAC were used. The hardware on these platforms afforded the ability to implement the algorithm directly without the need for additional observers and other state estimation techniques. To allow testing in a more practical nature, we introduce a modified potential field algorithm to manipulate the reference system. Ultimately, three tests are presented that span the envelope of the vehicles.

Through the discussion of these results, we see that the response is highly favorable with position error converging to within 20 cm of the intended position and typically residing at less than 10 cm.

Chapter 5

Saturation Algorithm for Ackermann Steering Based Platforms

In Chapters 3 and 4 we extend the control algorithm to account for kinematic constraints of differential drive platforms and assess the efficacy of these modifications. While these modifications relate to a wide range of mobile robotic platforms, differential drive is a relatively narrow subset of vehicle kinematics and by far not the most popular. Alternatively, Ackermann based steering platforms are much more widespread, being used in a multitude of industries and applications.

In this chapter, we will explore the use of Ackermann kinematic constraints and how it applies to the control algorithm put forth in this work. A dual stage saturation algorithm will be developed to handle these limitations, and the command changes will be accounted for through Lyapunov stability criteria in much the same way as the differential constraints in Chapter 3. Lastly, simulation results are presented to begin assessing the efficacy of these changes.

5.1 Ackermann Steering Kinematics

Ackermann steering based systems, as mentioned earlier, have a wide variety of use in various industries and applications. Although typically thought of as using four wheels and two axels, the Ackermann kinematics are usually simplified to a bicycle model, pictured in Figure 5.1 ([31, 32]). The general inputs to this kinematic model are a longitudinal velocity, $v(t) \in \mathbb{R}$, and a steering angle, $\phi(t) \in \mathbb{R}$, which determine the trajectory of the vehicle over time. With the current model in question, the wheels are assumed to adhere to a non-slip condition, meaning they are free rolling and have traction at all time.

As we did with the differential steering platform, it is desirable to express the Ackermann

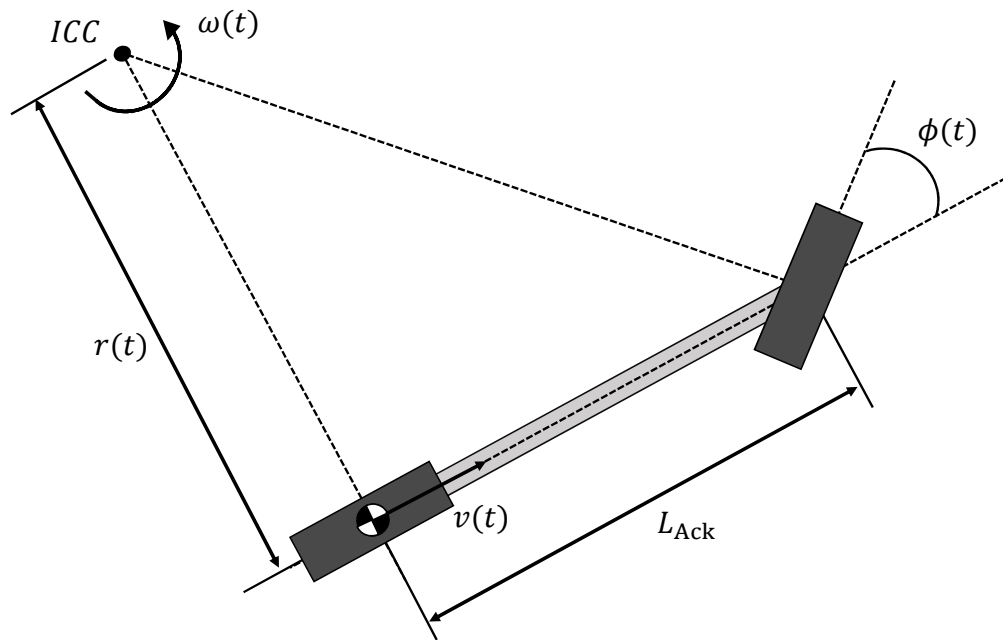


Figure 5.1: Ackermann steering platform kinematics.

inputs in terms of longitudinal and angular rates. By inspection, it is clear that the velocity of the platform pictured in Figure 5.1 is the same longitudinal velocity commanded by the controller, so no additional relationships are necessary. However, it is still necessary to relate the steering angle to the angular rate of the vehicle. Using the assumption that the front and rear tires are both moving on circular arcs centered at the ICC, it is simple to show that the radius of curvature can be determined as,

$$r(t) = \frac{L_{\text{Ack}}}{\tan(\phi(t))}, \quad (5.1)$$

where $L_{\text{Ack}} > 0$, is the distance separating the front and rear axles, commonly referred to as the vehicle's wheelbase. From here, we can substitute (3.3) into (5.1) to yield,

$$\omega(t) = \frac{v(t) \tan(\phi(t))}{L_{\text{Ack}}}. \quad (5.2)$$

Since we are able to control the longitudinal velocity and steering angle, it follows from (5.2) that we can effectively control the angular rate of the vehicle as well. Unlike the differential platform, however, the angular rate and longitudinal velocities are coupled with the implication that the vehicle cannot turn without forward or backward motion. While obvious, this behavior presents certain challenges for saturation that will be covered in the next section. Another interesting caveat of this relationship is that the vehicle has a limited radius of curvature, unlike differential platform. Since the previous saturation algorithm developed in Chapter 3 focused on maintaining the radius of curvature, this change will necessitate fundamental changes in the saturation architecture for Ackermann platforms.

5.2 Ackermann Steering Constraints

In the case of the differential platform, it was sufficient to limit the velocity commands in a way that directly related to the actuator capability. Because angular rates cannot be induced without forward or reverse velocity for Ackermann platforms, the constraints are slightly more stringent. As such, the velocity is limited to some minimum positive value, $v_{\min} > 0$. While it would be possible to design constraints that allow for negative velocities while eliminating arbitrarily small velocities, there was no availability of a vehicle that could distinguish positive from negative velocity for experimentation, and as such, the case is beyond the scope of this work. We further note that imposing a minimum velocity tends to be practical in many situations as most vehicles already have a minimum threshold on controllable velocity due to actuation and sensing constraints. Taking the additional velocity constraints into account, the Ackermann based constraints are as follows,

$$v_{\min} \leq v(t) \leq v_{\max}, \quad (5.3)$$

$$\left| \frac{\omega(t)}{v(t)} \right| \leq \frac{\tan(\phi_{\max})}{L_{\text{Ack}}}, \quad (5.4)$$

where $v_{\min} > 0$, is the minimum longitudinal velocity the vehicle can achieve, $v_{\max} > 0$, is the maximum longitudinal velocity the vehicle can achieve, and $\phi_{\max} > 0$, is the vehicle's maximum steering angle. Note that these constraints assume symmetric steering capability in either direction, an assumption that can easily be applied to the majority of cases. Considering the effect of these constraints on longitudinal and angular rates, which are the commands given by the controller derived in Section 2.2, we are able to determine a viable command envelope shown in Figure 5.2.

As would be expected from the constraint given in (5.4), we see that the envelope in Figure 5.2 is symmetric about the v axis in the command space due to the assumed symmetric steering. As it turns out, this feature makes the saturation algorithm presented in the next section easier to formulate, an added benefit to this assumption. From the experience gained in Chapter 3, it seems intuitive that the top and bottom borders of the envelope are

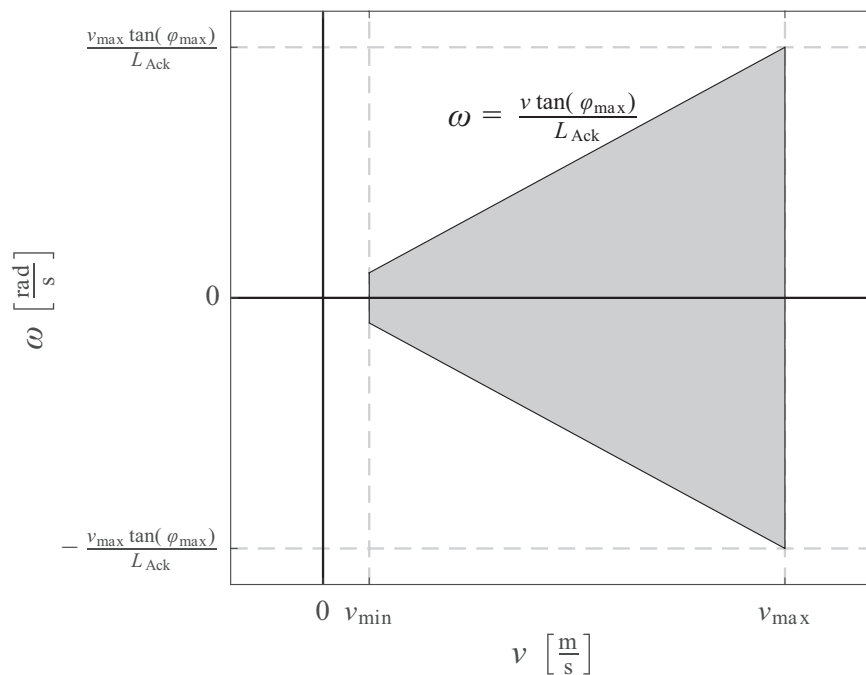


Figure 5.2: Ackermann steering command envelope (shaded).

essentially minimum bounds on the magnitude of the radius of curvature and can be found via the following relation,

$$\omega = \pm \frac{v \tan(\phi_{\max})}{L_{\text{Ack}}}. \quad (5.5)$$

5.3 Dual Stage Saturation Algorithm

During operation, the control law given in (2.8) can viably produce a command anywhere in the $v - \omega$ plane; however, it is desired to map the commands to somewhere within the given envelope while still preserving directional intent. Moreover, any mapping of commands should be continuous, not allowing for instantaneous transitions from one region of the envelope to another. In order to address these constraints, a dual stage saturation algorithm was developed and is presented in this chapter.

5.3.1 First Stage Saturation

Based on the choice of regions in the second stage of saturation, which will be discussed in Section 5.3.2, it is possible for commands to instantaneously change locations on the envelope. Specifically, if the controller commands cross the negative v axis, or pass through the origin, an instantaneous change will occur in the second stage mapping, which is undesirable. In order to prevent these potential problems and preserve directional intent, the first stage is used to prevent commands from entering into a small boundary around the negative v axis and the origin. It should be noted that this saturation stage does not result in commands mapped back to the envelope, which is instead achieved by the second stage.

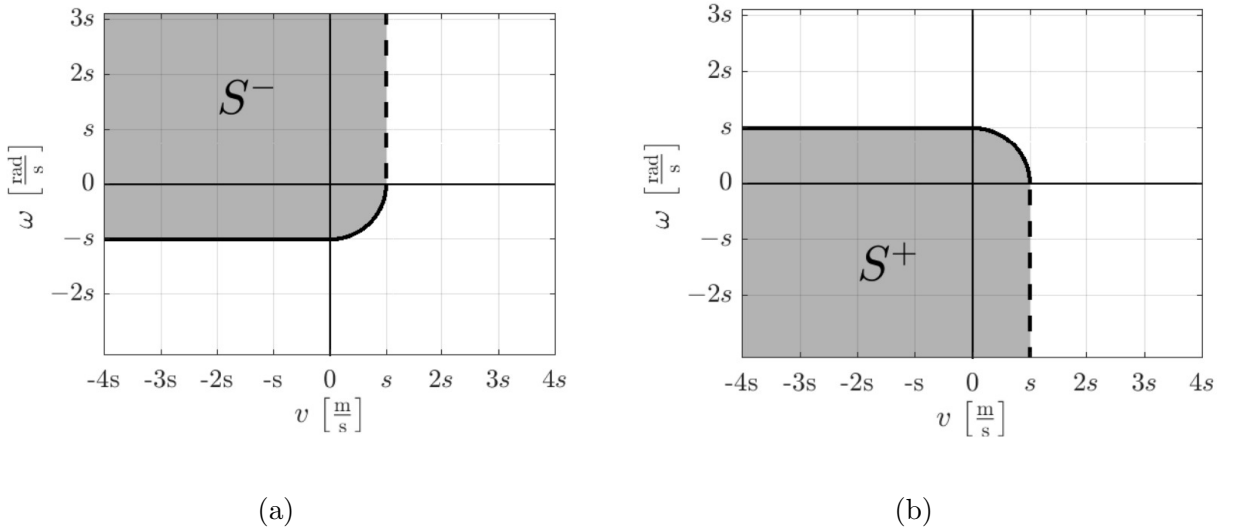


Figure 5.3: Graphical representation of (a) S^- and (b) S^+ .

The first stage ensures that controller commands cannot traverse the region shown in Figures 5.3 - 5.4, which is defined as

$$S \triangleq S^+ \cap S^-, \quad (5.6)$$

where

$$S^+ \triangleq \left\{ (v, \omega) \in \mathbb{R}^2 : 0 \leq \omega \leq s, v \leq \sqrt{s^2 - \omega^2} \right\} \cup \left\{ (v, \omega) \in \mathbb{R}^2 : v < s, \omega < 0 \right\}, \quad (5.7)$$

and

$$S^- \triangleq \left\{ (v, \omega) \in \mathbb{R}^2 : -s \leq \omega \leq 0, v \leq \sqrt{s^2 - \omega^2} \right\} \cup \left\{ (v, \omega) \in \mathbb{R}^2 : v < s, \omega > 0 \right\}, \quad (5.8)$$

where $s \in (0, v_{\min})$ defines the size of this small region around the negative v axis. In practice, s can be arbitrarily small as its only purpose is to provide a boundary around the

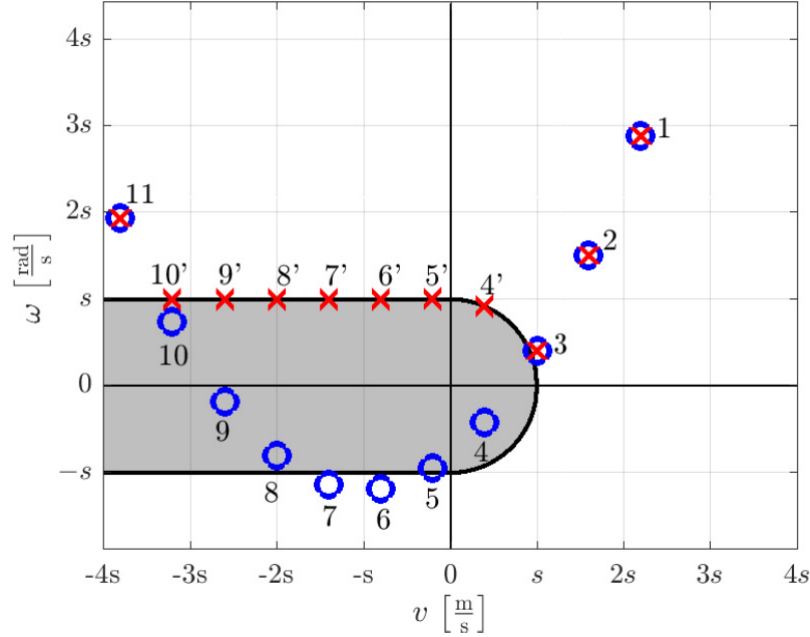


Figure 5.4: First stage representation of S with saturation example.

negative v axis. In general, small values, such that $s \ll v_{\min}$, are preferred to provide the least interference with controller intent. Figure 5.4 gives an example of the saturation implementation, which is detailed in the following discussion.

During implementation, if at time $t_e > 0$ the controller issues a command such that $(v(t_e), \omega(t_e)) \in \partial S$, then $\omega_e \triangleq \omega(t_e)$ is recorded. If $\omega_e \geq 0$, then for all subsequent controller commands such that $(v(t), \omega(t)) \in S^+$ the algorithm maps the commands to $(v(t), \omega_s(t)) \in \partial S^+ \cap S^+$. Likewise in the case of $\omega_e < 0$, for all subsequent commands such that $(v(t), \omega(t)) \in S^-$ the algorithm maps the command to $(v(t), \omega_s(t)) \in \partial S^- \cap S^-$. Note that $\omega_s(t)$, the resultant angular rate command, is defined by the point on the boundary that corresponds to the original velocity command.

Figure 5.4 demonstrates a parabolic command evolution before and after the first saturation stage. In this case, we consider the commands evolving from right to left in the order they are numbered. The blue circles represent the original commands from the controller, and the red crosses demonstrate the resulting command after the first saturation stage. Examining the figure, we see that commands 1-3 are left unaltered, as they are outside of the region S . At some point between 3 and 4, the command crosses the boundary of S , at which point ω_e is recorded to be some positive value. Since $\omega_e \geq 0$, all subsequent commands such that $(v(t), \omega(t)) \in S^+$ are mapped to $\partial S^+ \cap S^+$, while preserving the velocity command,

$v(t)$. This trend is the case for commands 4-10 which are mapped to 4'-10', respectively. Of particular interest are commands 6 and 7. Although these commands exit the region S , they are still contained in S^+ . As such, the saturation algorithm continues to map these commands to ∂S^+ using the same methodology. Lastly, we look at command 11, which occurs at some point after the commands have exited S^+ . In this case, since $(v(t), \omega(t)) \notin S^+$ the saturation algorithm is no longer active.

5.3.2 Second Stage Saturation

Once the commands are passed through the first stage of saturation, they are categorized to a specific region based on their location in the $v - \omega$ plane. Figure 5.5 shows four separate regions which use various methodologies to map the commands into the viable envelope. In Region 1, we focus on preserving the radius of curvature of the original commands as this directly relates to the intended path of the controller. Due to the limitations of the Ackermann platform, however, not all curvatures are achievable. As such, in Region 2, the second stage saturation algorithm maps commands to the nearest achievable radius of curvature while preserving lateral acceleration. We choose to preserve lateral acceleration so that the acceleration perceived by the vehicle and its sensors will remain unchanged. In the case of Regions 3 and 4, commands originate outside of any achievable radius of curvature or lateral acceleration. As a result, these commands are mapped to the closest achievable points that preserve intent. As expected, all commands originating within the envelope are left unaltered. For the remainder of this discussion, we only consider cases of $\omega(t) \geq 0$, as the other case is simply a reflection about the v axis.

Region 1 Methodology

Ideally, during saturation it is desirable to reproduce the same trajectories that are commanded by the controller. In order to elicit this behavior, we examine the radius of curvature given in (3.3) and repeated here for convenience,

$$r(t) = \frac{v(t)}{\omega(t)}. \quad (5.9)$$

When the requested command lies within Region 1, which is given by the dark blue regions on the left and right side of the envelope in Figure 5.5, the second stage saturation algorithm maps the new command to the nearest point on the envelope that provides the same radius of curvature. By maintaining the same radius of curvature, we ensure that the intended path is followed to the best of the vehicles capability. We note that the origin is included as part of Region 1, however, this presents a potential for discontinuous mapping depending on command evolution. For this reason, the first stage saturation method is used to prevent transitions through the origin.

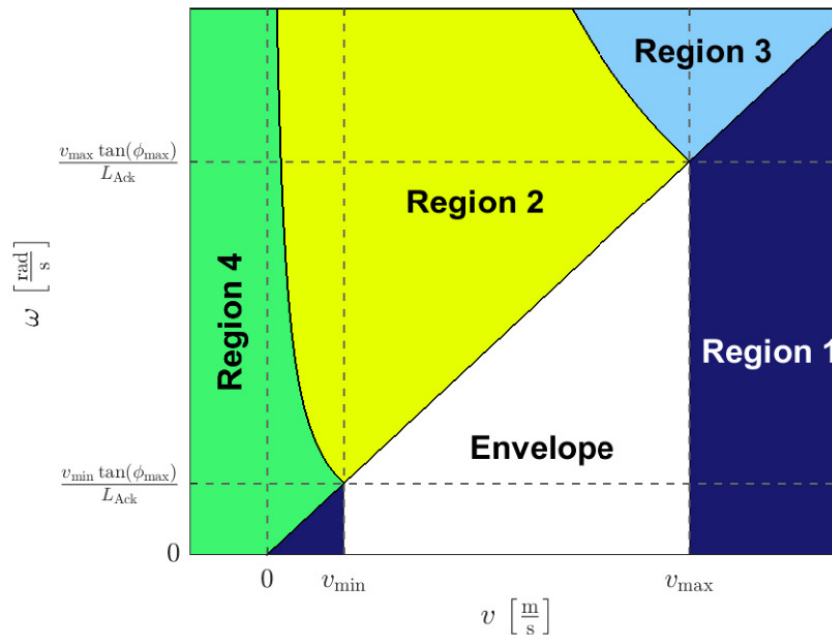


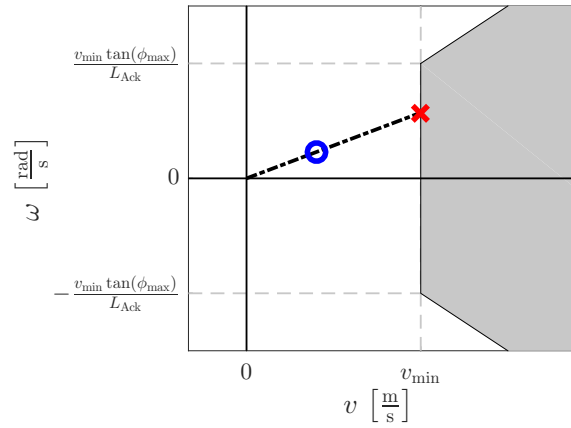
Figure 5.5: Region classification for Ackermann second stage saturation.

Figure 5.6 provides several examples of Region 1 saturation. In Figure 5.6, the blue circles represent the command received from the first stage of saturation, and the red crosses represent the command following the second saturation stage. In the first example, the command is mapped to the minimum velocity while still preserving the $\frac{\omega(t)}{v(t)}$ ratio. The second and third examples exhibit the same behavior, instead mapping to the maximum velocity. Of particular interest is the third example, which maps along the v axis. In this case, the radius of curvature is infinite, but by using the inverse of curvature during implementation, the methodology is preserved. For this reason, the reciprocal of the radius is typically used to prevent singularities in computation.

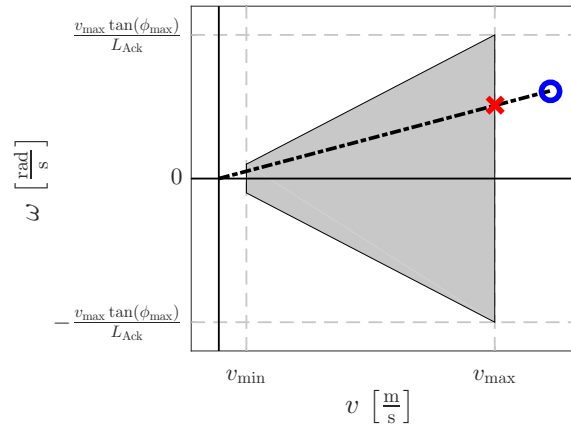
Region 2 Methodology

Due to the constraints of an Ackermann platform, there are a variety of conditions in which the desired radius of curvature is not achievable, as is the case of Region 2. To find a reasonable alternative, we start by considering the lateral acceleration of the vehicle given by

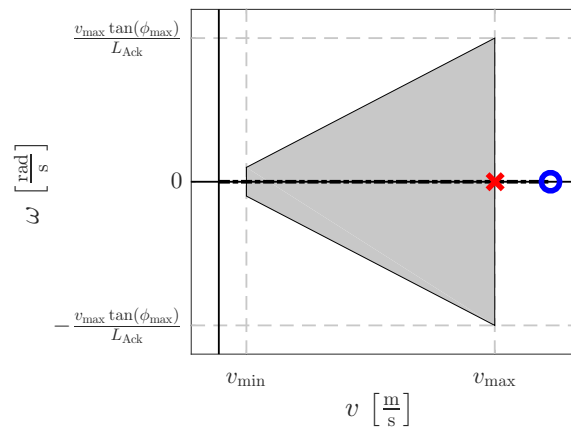
$$a_L(t) = v(t)\omega(t). \quad (5.10)$$



(a)



(b)



(c)

Figure 5.6: Region 1 examples for second stage saturation of Ackermann platforms.

Commands originating in Region 2 are mapped to the point on the envelope that is characterized by the same lateral acceleration. This point also corresponds to the nearest achievable radius of curvature. Figure 5.7 demonstrates saturation behavior for Region 2. Again, the blue circle represents the command received from the first stage of saturation, and the red cross represents the command following the second saturation stage. In this figure, the black dashed line represents the laws of constant lateral acceleration. As explained, both commands have the same lateral acceleration, with one being in the achievable command space.

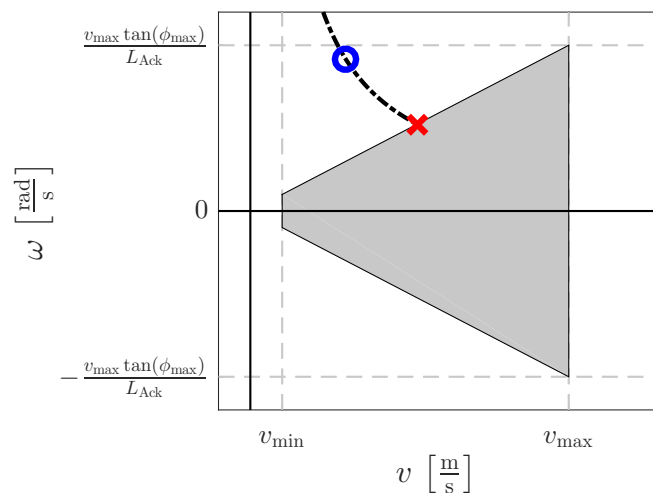


Figure 5.7: Region 2 example for second stage saturation of Ackermann platforms.

Region 3 Methodology

While lateral acceleration offers an intuitive choice for mapping commands to an obtainable radius of curvature, Ackermann platforms inherently have a maximum lateral acceleration, given by

$$a_{L_{\max}} \triangleq v_{\max}^2 \frac{\tan(\phi_{\max})}{L}. \quad (5.11)$$

When the requested command exceeds the minimum radius of curvature and the maximum lateral acceleration, the command is mapped to the maximum velocity and maximum turning angle, represented by the corner of the envelope. Region 3 of Figure ?? depicts the area in question. By holding the command at the corner of the envelope in Region 3, we ensure a continuous mapping from Region 2 to Region 3 and from Region 3 to Region 1.

Region 4 Methodology

Although lateral acceleration offers a good methodology for mapping commands, it does not always elicit optimal behavior. Recalling from Section 2.3, the controller can command high angular rates while also commanding little to no longitudinal velocity. In these cases, the lateral acceleration is small. If we were to map to the envelope maintaining this lateral acceleration, the result would yield a forward velocity, with a nearly zero angular rate, which would be opposite to desired. Instead, the vehicle should ideally turn as sharp as possible.

In Region 4, we examine the situation in which the radius of curvature is not obtainable and the command lies below the minimum lateral acceleration threshold given by

$$a_{L_{\min}} \triangleq v_{\min}^2 \frac{\tan(\phi_{\max})}{L}. \quad (5.12)$$

In this case, the command is mapped to the minimal velocity and maximum angular rate, characterized by the left corner of the envelope given in Figure 5.5. For the situations in which the vehicle has a large requested angular rate with small velocity, the algorithm ensures that the vehicle turns as sharp as possible at a minimal velocity until the controller requests more achievable behavior.

We extend Region 4 to encompass all negative velocity commands as well since the goal is to turn as much as possible until the controller requests positive velocities. We note that while Region 4 includes portions of the first stage of saturation, these portions will not be reached due to the effects of the first stage of saturation. If the controller commands were to cross the v axis while the requested velocity is negative, the second stage saturation algorithm would fully reverse the steering command. However, the first level of saturation restricts this potential chattering behavior by keeping the command in the same quadrant until the controller requests positive velocities.

5.3.3 Maintaining Lyapunov Criteria

In the case of the first and second stages of saturation, the commands are altered without consideration of how the stability criteria for the controller may be expected. To account for this behavior, the same methodology utilized in Section 3.3.1 is used in this case. As was done in (3.9) the new reference system time derivative is found using the saturated commands resulting from the presented algorithms. This equations is repeated here for convenience,

$$\dot{p}_r^{\text{sat}}(t) = R(\theta(t)) \left(\Delta(t) \begin{bmatrix} v^{\text{sat}}(t) \\ \omega^{\text{sat}}(t) \end{bmatrix} - K \tanh(e(t) - \delta(t)) + \dot{\delta}(t) \right), \quad (5.13)$$

where $\dot{p}_r^{\text{sat}}(t) \in \mathbb{R}^2$ is the resulting reference time derivative, $v^{\text{sat}}(t) \in \mathbb{R}$ is the saturated longitudinal velocity command, and $\omega^{\text{sat}}(t) \in \mathbb{R}$ is the saturated angular rate command.

As also mentioned in Section 3.3.1, it is necessary to guarantee that $d^*(t)$ must be at least a class C^1 function in time. This constraint is also accounted for using the modification presented in (3.10) and repeated here for convenience as,

$$\ddot{d}^*(t) + 2\zeta_d\omega_d\dot{d}^*(t) + \omega_d^2d^*(t) = \omega_d^2d_{\text{ref}}(t), \quad (5.14)$$

where $\zeta_d > 0$, $\omega_d > 0$ are tuning constants, and $d_{\text{ref}}(t)$ is user desired reference distance.

5.4 Ackermann Steering Simulation Results

In order to keep the simulation consistent with previous findings, the reference following distance is given the same relationship as (3.11), repeated here for convenience as

$$d_{\text{ref}}(t) = \alpha v_r(t) + \beta. \quad (5.15)$$

Moreover, all other simulation parameters, listed in Table 5.1, are kept as consistent as possible with the first set of simulations in Section 2.3 and Section 3.4, with the exception of Ackermann specific saturation parameters.

Table 5.1: Ackermann platform simulation parameters with $v_{\text{max}} = 10$. All quantities expressed using standard SI units.

Desired Reference Trajectory	Saturation Constraints	Tuning Parameters	Initial Conditions
$r_x(t) = 0.5t$ $r_y(t) = 10 \sin(0.5t)$	$v_{\text{max}} = 10$ $v_{\text{min}} = 1$ $L_{\text{Ack}} = 0.3556$ $\phi_{\text{max}} = 0.4363$ $s = 0.01$	$k_v = 1$ $k_\omega = 1$ $\lambda = 1$ $\alpha = 0.5$ $\beta = 0.1$ $\epsilon = 0.05$ $\zeta_d = 0.85$ $\omega_d = 2.5$	$x_r(0) = 0$ $y_r(0) = 0$ $x(0) = -0.1$ $y(0) = 0$ $\theta(0) = 0$ $d(0) = 0.1$

Figure 5.8 demonstrates the resulting trajectory of using the parameters of Table 5.1. Overall, the vehicle still follows the trajectory given by $\begin{bmatrix} r_x(t) & r_y(t) \end{bmatrix}^T$, which is desired. However, in the case of sharp turns, the vehicle continues forward, locking itself into a circular trajectory until it is able to re-obtain the desired path. This limit on curvature is the most notable difference between the saturation algorithm performance seen in Figure 3.4 and the Ackermann saturation method. To facilitate this behavior, the reference system is pushed away from the trajectory detailed by $\begin{bmatrix} r_x(t) & r_y(t) \end{bmatrix}^T$, overriding the dynamics described by

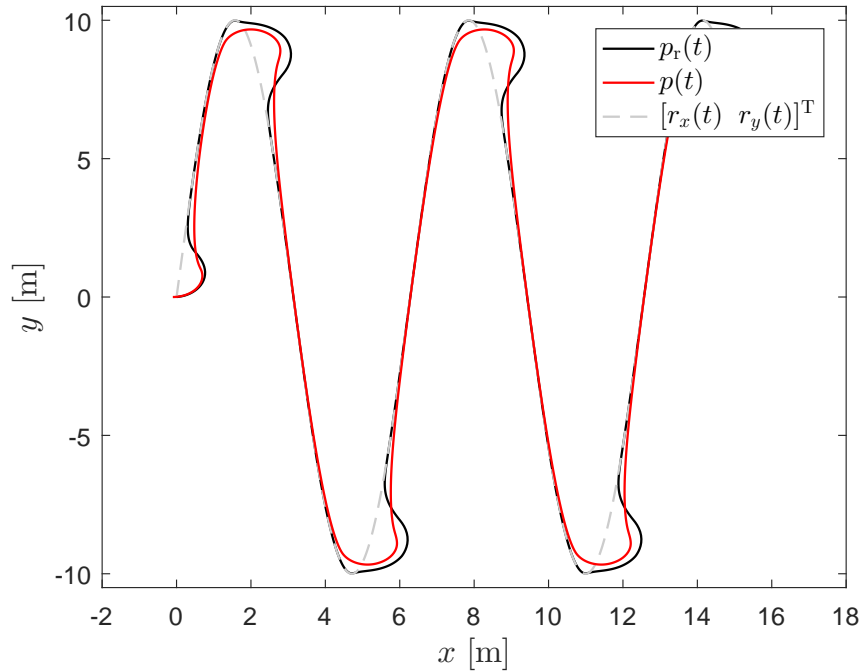


Figure 5.8: Ackermann platform tracking of sinusoidal trajectory with $v_{\max} = 10$.

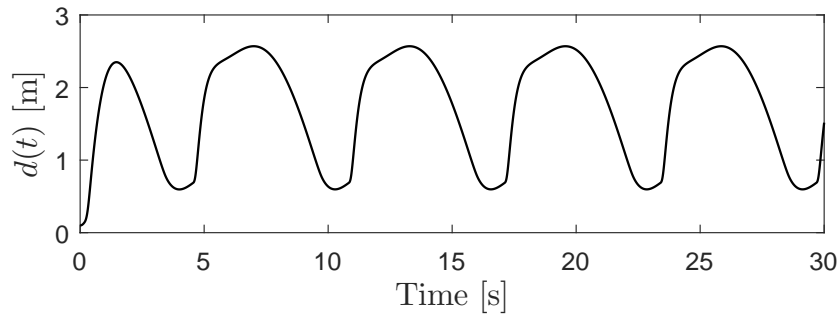


Figure 5.9: Ackermann platform commanded following distance with $v_{\max} = 10$.

(2.21). In addition to following the path, the saturation algorithm also eliminated the prior initial behavior seen in Figure 2.1, characterized by an immediate reversal of trajectory to satisfy distance requirements. Instead, we observe a situation in which the vehicle continues smoothly in a forward direction while steering back into the path.

In addition to tracking performance, we see that the follow distance behaves largely as expected in Figure 5.9. In fact, in comparing this distance to the response seen in Figure 3.5, we see that the different saturation methods affect little change on the following distance,

Table 5.2: Ackermann platform simulation parameters with $v_{\max} = 2$. All quantities expressed using standard SI units.

Desired Reference Trajectory	Saturation Constraints	Tuning Parameters	Initial Conditions
$r_x(t) = 0.5t$ $r_y(t) = 10 \sin(0.5t)$	$v_{\max} = 2$ $v_{\min} = 1$ $L_{\text{Ack}} = 0.3556$ $\phi_{\max} = 0.4363$ $s = 0.01$	$k_v = 1$ $k_\omega = 1$ $\lambda = 1$ $\alpha = 0.5$ $\beta = 0.1$ $\epsilon = 0.05$ $\zeta_d = 0.85$ $\omega_d = 2.5$	$x_r(0) = 0$ $y_r(0) = 0$ $x(0) = -0.1$ $y(0) = 0$ $\theta(0) = 0$ $d(0) = 0.1$

which in general is desirable. This minor effect allows the intuitive choosing and tuning of the following distance so that the vehicle can better track the reference system at low speeds while maintaining sufficient tracking distance at high speeds.

Further examination of the behavior is given in Figure 5.10. Clear limitations in the velocity are observed between $1 \frac{\text{m}}{\text{s}}$ and $10 \frac{\text{m}}{\text{s}}$. Additionally, the implemented algorithm provides a smooth velocity command, as expected. This behavior is a result of a continuous mapping of saturation commands back to a region on or within the envelope.

Looking further at Figure 5.10, we see that the angular rate commands are significantly different from those obtained without saturation constraints. In general, the commanded angular rates achieve higher magnitudes than previously observed. From the third graph, however, it is obvious that there are clear bounds on the ratio of $\frac{\omega(t)}{v(t)}$. As such, the vehicle achieves higher angular rates, while still following the criteria of an Ackermann based platform.

As with the simulation in Section 3.4 the data represented in Figures 5.8 - 5.10 behaves favorably. Unlike those simulations, however, we see more obvious modifications in the trajectory due to the minimum curvature restrictions now placed on the platform. With that said, however, another key motivation of this work is to limit the reference system based not only on the restrictions in steering, but the restrictions in velocity as well. To explore this scenario, another simulation is introduced in which the maximum velocity is changed from $v_{\max} = 10 \frac{\text{m}}{\text{s}}$ to $v_{\max} = 2 \frac{\text{m}}{\text{s}}$. This parameter and all additional parameters of this simulation, which are consistent with the previous, are given in Table 5.2.

Figure 5.11 shows the resulting trajectory in the case that longitudinal velocity is a major limiting factor. At first glance, we see that the resulting trajectory is similar to that in Figure 3.7 in which the velocity also served as a limiting factor. This behavior is ideal based

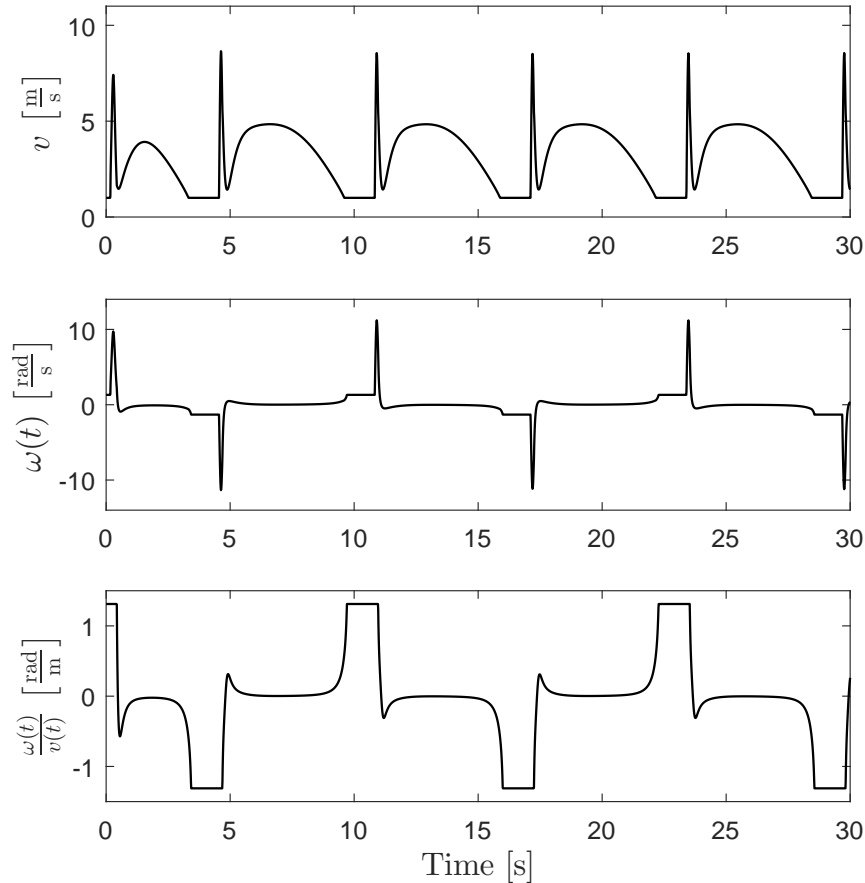


Figure 5.10: Ackermann platform controller commands with $v_{\max} = 10$.

on our design and requirements. As the desired trajectory goes beyond vehicle capability, we see that the reference system is limited to whatever the vehicle is capable of, while still preserving directional intent.

Along with other similarities with the differential saturation algorithm, we note the commanded following distance given in Figure 5.12. While mildly fluctuating throughout the simulation, we see that the distance does not largely vary and follows much the same trend as the differential drive simulation in Figure 3.8. Again, this result is favorable as it shows that our following distance design remains largely unchanged between the differential and Ackermann saturation designs. This quality enables more intuitive tuning and understanding across platforms.

While similar, a notable difference between the Ackermann and differential platform responses can be seen in Figure 5.11 after each turn away from the peaks. In each of these

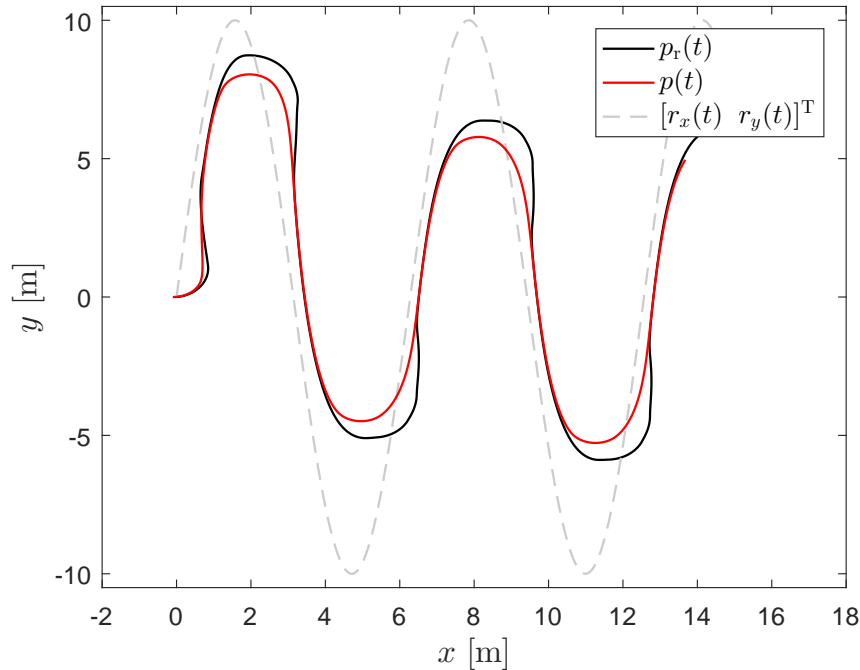


Figure 5.11: Ackermann platform tracking of sinusoidal trajectory with $v_{\max} = 2$.

cases, the vehicle never fully turns back into the path, unlike the differential platform response in Figure 3.7, which managed to make it back to the desired path for a portion of its journey. The reason behind this difference is directly related to the designed behavior of the Ackermann platform when requested to undergo a negative velocity. Instead of following through with that command, it ultimately goes to the extent of its steering while continuing forward at its slowest speed. This behavior is corroborated by the circular paths at the peaks in Figure 5.11 and the commands of Figure 5.13. As a result, when the vehicle is allowed to speed up, it is much further away from the desired trajectory than the differential restrictions were in Section 3.4. As such, it is left driving toward a point that is further down the path and has a greater x coordinate value.

Looking further at the commands presented in Figure 5.13, we see that the vehicle is abiding by the restrictions put forth in Table 5.2. As seen in Figure 5.10 the longitudinal velocity is bounded to the set capabilities as well as the ratio of angular rate to longitudinal velocity. At first glance, it appears that there may be discontinuities in the velocity command as it goes from v_{\min} to v_{\max} , however, this is simply due to the fact that there is a relatively small envelope, and by extension an unusually quick transition from the minimum lateral acceleration to the maximum achievable lateral acceleration as the command moves from Region 4, to Region 2, and then ultimately to Region 3, as laid out in Figure 5.5 by the second stage methodology. The general brevity of this transition combined with the relatively

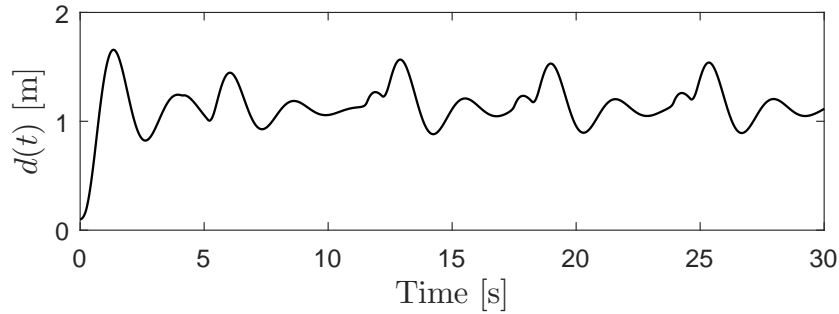


Figure 5.12: Ackermann platform commanded following distance with $v_{\max} = 2$.

large time scale of Figure 5.13 can cause the commands to appear discontinuous, although this is not the case.

5.5 Summary

In this chapter, saturation constraints are introduced based on Ackermann steering platform kinematics. By examining a bicycle kinematic model, constraints and a desired command envelope are determined which is based on typical vehicle limiting factors such as speed and steering angle. As is done in Chapter 3, a saturation methodology is developed that focuses on mapping commands from the base controller back into this desired envelope. This task is accomplished through dual stage algorithm that focuses on preserving directional intent, desired steering curvature, and desired lateral acceleration when possible. Again, as seen in Chapter 3, the reference system is altered so that Lyapunov stability criteria are still met. Lastly, simulation results are presented that demonstrate the saturation algorithm for Ackermann platforms behaves as intended.

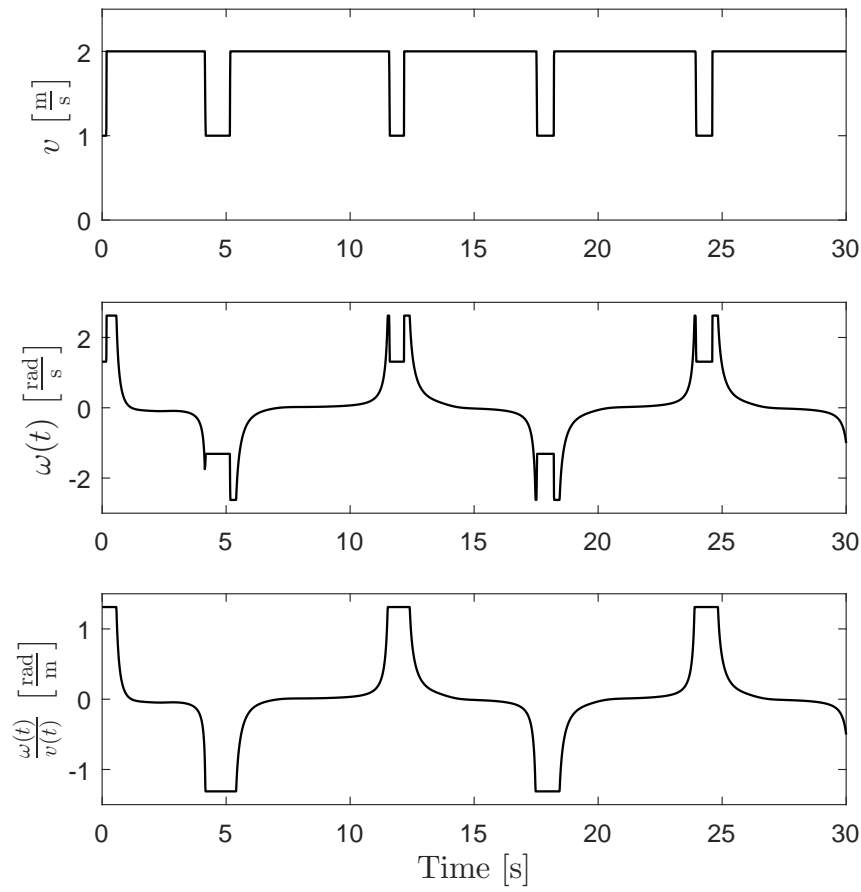


Figure 5.13: Ackermann platform controller commands with $v_{\max} = 2$.

Chapter 6

Experimental Results for Ackermann Steering Platforms

In Chapter 5, we developed the methodology to incorporate vehicle capabilities into the controller developed in Chapter 2 for Ackermann steering based platforms. In this chapter, we will discuss the experimental results obtained by applying this controller to a Ackermann steering driven small scale vehicle. As was introduced and used in Chapter 4, we will use potential field algorithms to manipulate the reference system. Before introducing and discussing the results, we will first give an overview of the experimental setup used in these tests.

6.1 Experimental Setup

Along with the TURTLEs used in Chapter 4, the Center for Dynamic Systems Modeling and Control allowed access to a small scale Ackermann steering based platform pictured in Figure 6.1. The base platform of this vehicle consists of an electric RC vehicle manufactured by Traxxas. This platform natively provides a brushless electric motor and Electronic Speed Control (ESC) that provides acceleration and braking capability for speeds up to $11 \frac{\text{m}}{\text{s}}$, and a standard hobby servo used for steering. The motor and servo are powered using a 5000 mAh 6S lithium polymer battery and controlled using a Pololu 6 channel USB servo controller.

With regard to sensing, computing, and peripheral devices, a separate 5000 mAh 6S lithium polymer battery is interfaced with an onboard voltage regulator that distributes 3.3V, 5V, and 12V to any additional equipment. This regulator serves to power an onboard computer which hosts an Intel Atom 2.4 GHz processor, 2 GB of RAM, and 64 GB SSD running Windows XP. The computer, which directly controls the servo controller, also connects to a 2.4 GHz radio manufactured by Ubiquiti Networks. In regard to sensing, the vehicle has a Trimble BD970 GPS which has an accuracy of ± 1 m and up to ± 1 cm using RTK corrections.

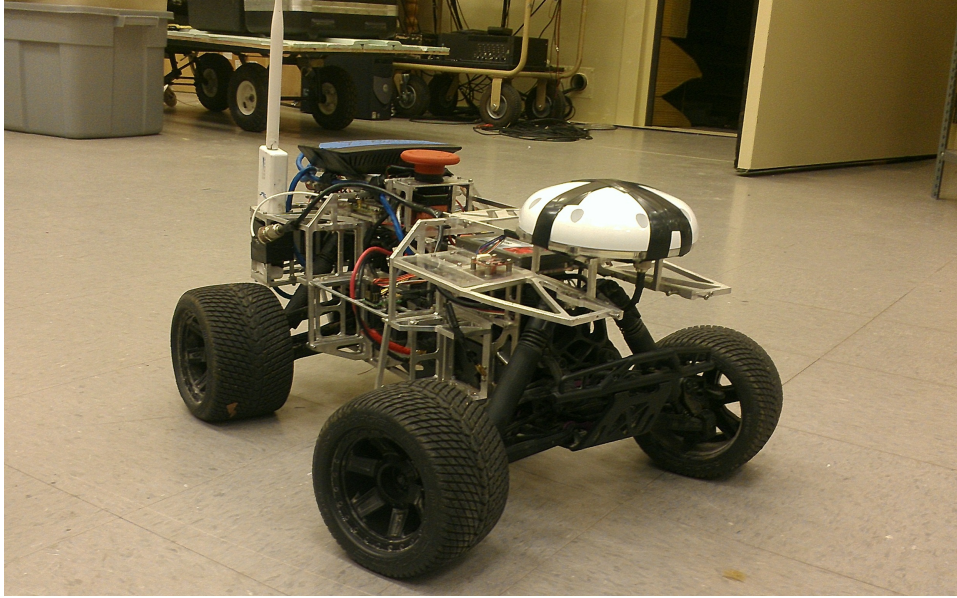


Figure 6.1: Ackermann steering test platforms (TURTLEs).

Along with the GPS, the vehicle incorporates a Microstrain 3DM-GX2 Inertial Measurement Unit (IMU) that provides Euler angles at $\pm 2^\circ$, acceleration on 3 axes at $\pm 0.01 g$, and angular rates at $\pm 0.2^\circ/\text{sec}$.

For the purpose of this series of experiments, all controller code and robot manipulation algorithms were written in LabVIEW and run directly on the vehicle. Additionally, all data collected was stored on the vehicle during collection. The controller and data input rates were set at 40 Hz. No RTK corrections were used during testing. Due to the inadequacies of direct measurement, an Extended Kalman Filter was written for the purpose of this study to provide reliable feedback information. Additionally, two low level PI controllers were used to control velocity and angular rate so that the controller could be implemented as discussed in this document.

6.2 Tracking Performance

To test the tracking performance, the reference system was manipulated using the potential field algorithm described in Section 4.2, as was the case in Section 4.3. With the exception of the controller tuning parameter k_ω and the potential field reference velocity $v_{r_{\max}}$, all saturation and tuning parameters were kept the same for the following experiments. These parameters are listed in Table 6.1. In the case of $v_{r_{\max}}$, the specific variations, listed at the top of the third column in Table 6.1, the variation is simply to test the performance across a range of speeds. On the other hand, the variations in k_ω , which are listed at the top of the

Table 6.1: Ackermann platform experimental tracking parameters. All quantities expressed using standard SI units.

Saturation Constraints	Tuning Parameters	Potential Field Parameters
$v_{\max} = 10$	$k_{\omega} = 1, 2.5, 3.5$	$v_{r_{\max}} = 2, 5, 9$
$v_{\min} = 1$	$k_v = 0.5$	$F_{ac} = 10$
$L_{Ack} = 0.3556$	$\lambda = 1$	$F_{rc} = 6$
$\phi_{\max} = 0.4363$	$\alpha = 1.3$	$W = 2$
$s = 0.01$	$\beta = 1$	$n = 1.5$
	$\epsilon = 0.5$	$E = 14$
	$\zeta_d = 0.85$	
	$\omega_d = 2.5$	

second column, are used to account for vehicle dynamics which are not fully accounted for in the control design.

The low speed tracking results are given in Figure 6.2. In this figure, the reference system and vehicle are proceeding from waypoint to waypoint in a clockwise fashion, with the first waypoint positioned at approximately (0,-5) and the last point at (-15, 5). As expected, the vehicle (red) tracks the reference trajectory (black) tightly. This is the typical behavior at low speeds, as the vehicle lags the reference system by a relatively small amount. While there are some slight deviations at the bottom and top of the graph, the overall performance is favorable.

While following the trajectory is arguably the most important performance aspect of the controller, another key feature is the ability to maintain the commanded following distance $d(t)$ such that $\|e(t) - \delta(t)\| \rightarrow 0$ as $t \rightarrow \infty$. We use Figure 6.3 to better elaborate on this aspect of the performance. At the start of the experiment, the error starts relatively high at almost 2.5 m. It drops quickly, however, to within 1 m and while fluctuating, stays under 1.6 m for the duration of the experiment. It is important to realize that the quantity pictured in Figure 6.3 is a reflection of both the error along the longitudinal axis of the vehicle as well as along the lateral axis. As it turns out, the higher frequency fluctuations are attributed to longitudinal error and velocity dynamics whereas the lower frequency fluctuations, such as the one occurring from the time interface $t = 60$ s to $t = 100$ s, result largely from lateral error influenced predominately by steering.

This fact is further corroborated by the controller commands and vehicle response graphs given in Figure 6.4. Looking at the velocity control response in the top graph, we see that the fluctuations occur largely at the same frequency as the faster fluctuations in Figure 6.3. In the case of the guiding control law, this response is the case of the vehicle trying

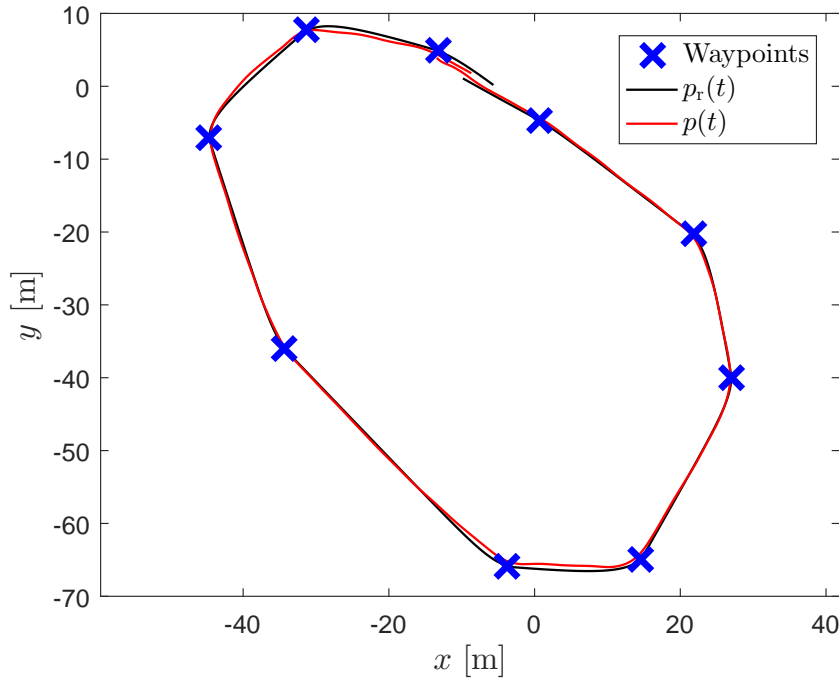


Figure 6.2: Tracking performance of waypoints with $k_\omega = 1$ and $v_{r_{\max}} = 2$.

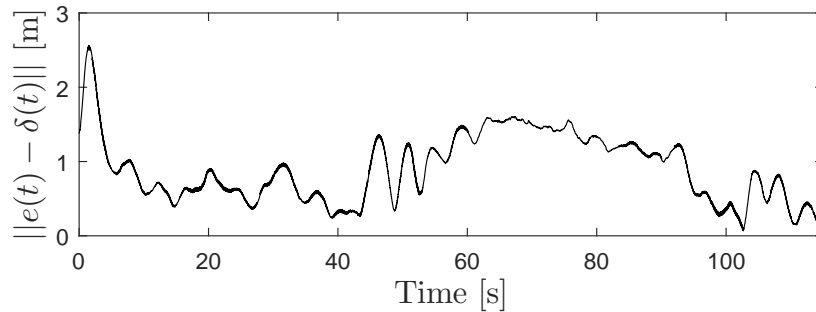


Figure 6.3: Total error, $e_1(t)$, with $k_\omega = 1$ and $v_{r_{\max}} = 2$.

to maintain the distance behind the reference system, strengthening the claim that the higher frequency deviations of Figure 6.3 occur because of longitudinal error, whereas low frequency deviation is a resulting from steering deviation. Had the low frequency deviation been a response to following distance increasing, this would have been reciprocated by an increase in the commanded velocity. Instead, the velocity settles out to around $2 \frac{\text{m}}{\text{s}}$ around that time period, the same velocity as the reference system.

Looking further at Figure 6.4, we see that the response is typical of many dynamic systems

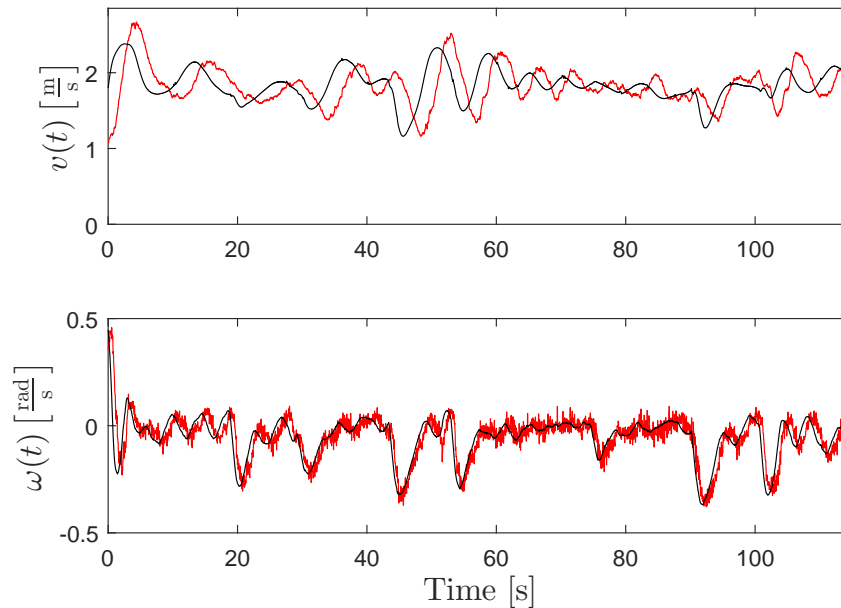


Figure 6.4: Controller commands (black) and measured vehicle response (red) for $k_\omega = 1$ and $v_{\text{rmax}} = 2$.

with obvious phase shifts occurring between the commands, seen in black, and the actual response, seen in red. The main takeaway from this response is that it is a behavior not directly accounted for in the control design, although the algorithm in place has thus far shown to be capable of handling this scenario. Another key note about this data is that it was taken in a non ideal environment. The experiments were performed in a parking lot on the Virginia Tech campus that was heavily sloped and had a variety of irregularities, introducing many disturbances into the system throughout the tests. In light of these facts, the experiments presented in Figures 6.2 - 6.4 show highly favorable results for control of Ackermann platforms.

To further explore the capabilities of the controller, we examine the response of the vehicle to a reference system that is moving at $5 \frac{\text{m}}{\text{s}}$. All parameters are consistent with those in the previous example, with the exception of $k_\omega = 2.5$ and $v_{\text{rmax}} = 5 \frac{\text{m}}{\text{s}}$. The tracking performance of this test is presented in Figure 6.5. At first glance, it is clear that the vehicle is cutting to the inside of the reference system's path, as seen in the bottom of the graph and in the top. This performance is ideal based on the derivation of the controller. As the vehicle moves faster, the following distance will increase. As a result, the vehicle begins correcting its trajectory earlier in the path. Consequently, the vehicle moves to the inside of the reference system's path.

Although ideal to move to the inside of the path, that is not always the case in this experi-

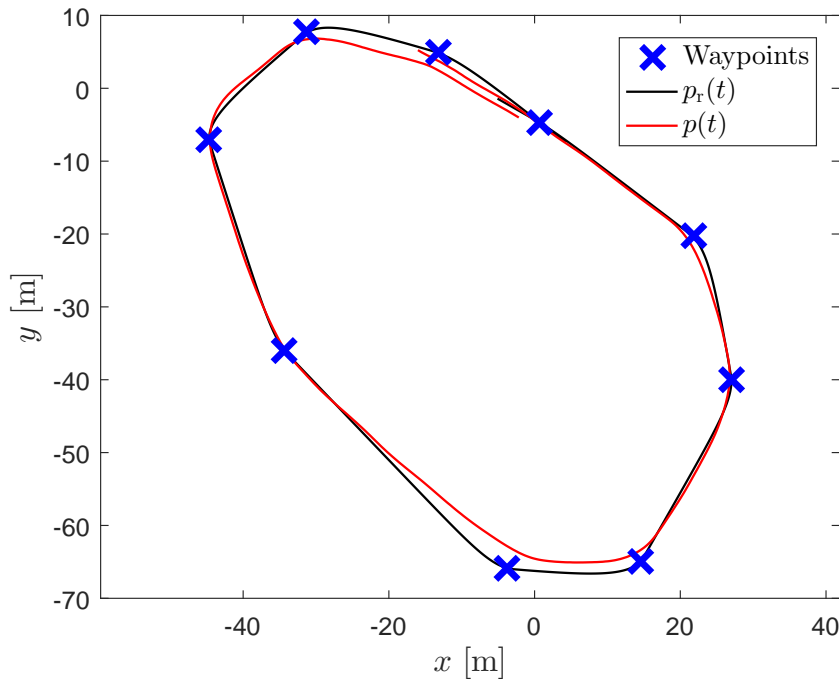


Figure 6.5: Tracking performance of waypoints with $k_\omega = 2.5$ and $v_{r_{\max}} = 5$.

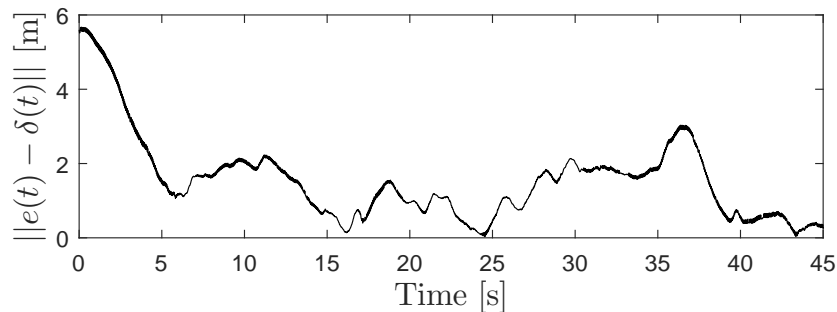


Figure 6.6: Total error, $e_1(t)$, with $k_\omega = 2.5$ and $v_{r_{\max}} = 5$.

ment. We see that as the vehicle moves past the waypoint located at approximately $(-40, -8)$ in Figure 6.5 it continues beyond the reference system's path. The cause of this deviation is linked to actuator dynamics, which is somewhat mitigated by an increased value of k_ω from the previous experiment.

To further assess the efficacy of the algorithm at $5 \frac{\text{m}}{\text{s}}$, we examine Figure 6.6. At the start of data collection for this experiment, the vehicle is almost 6 m from its intended location. This value drops to under 2 m and stays near this value for the majority of the experiment.

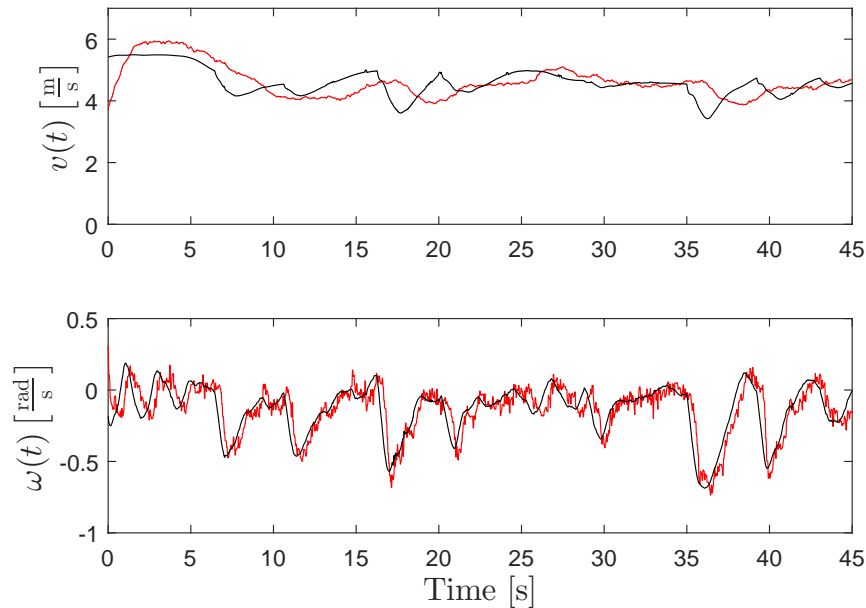


Figure 6.7: Controller commands (black) and measured vehicle response (red) for $k_\omega = 2.5$ and $v_{r_{\max}} = 5$.

We see that during the time period, approximately $t = 15\text{s}$ to $t = 25\text{s}$, in which the vehicle was cutting to the inside of the turn as the bottom of Figure 6.5, the deviation is actually at its smallest values. This behavior strengthens the claim that the controller naturally biases toward cutting to the inside of the turn when the reference system more heavily leads the vehicle. Along the same lines, in the instance that the vehicle moves out beyond the reference path at approximately $t = 35\text{s}$ to $t = 40\text{s}$, the error increases sharply. Looking at Figure 6.7, we further attribute this large increase to dynamic constraints based on the fact that angular rate command is undergoing a large change at this point in time. Moreover, while the angular rate response matches it, the phase lag is enough to allow the vehicle to continue past its target point increasing the total error of the system.

Looking further at the commands of the controller given in Figure 6.7, we see the behavior is as expected. With regard to the velocity command and response, it appears to have somewhat less fluctuation than the data in Figure 6.4, but the difference is marginal and somewhat misleading due to the difference in scaling of the graphs. From the velocity command, we also see that the small deviations in error are largely following distance dependent while the large deviations are steering dependent, as was the case with the first experiment in this section. This observation is supported by the increase in error at approximately $t = 16\text{s}$ which then affects an increasing velocity command around the same time. Likewise, we previously established that the large increase in error corresponded with a quickly varying angular rate command.

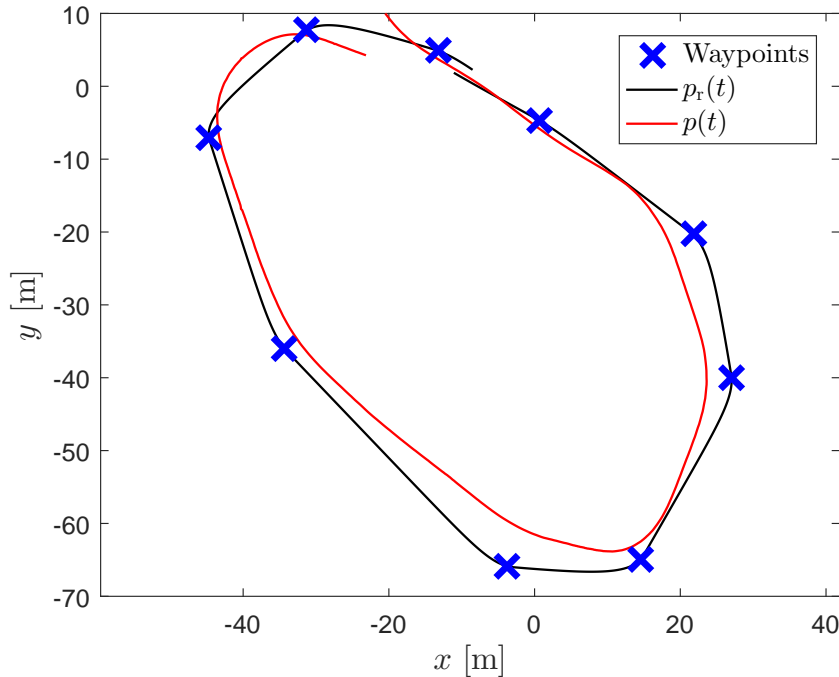


Figure 6.8: Tracking performance of waypoints with $k_\omega = 3.5$ and $v_{r_{\max}} = 9$.

Overall, the performance of the experiments in Figures 6.5 - 6.7 is encouraging. While there are some instances of actuator dynamics perhaps unnecessarily increasing the error of the system, we are still able to track a reference trajectory without serious deviation. Keeping in mind, some of these effects were also mitigated by increasing k_ω at higher speeds. In actuality, the performance would have suffered more using the same parameters as the first experiment of this section. To further extend the analysis of the control law, we lastly consider the case in which the reference velocity is set to $9 \frac{\text{m}}{\text{s}}$.

As before, all parameters are consistent with those in the previous example, with the exception of $k_\omega = 3.5$ and $v_{r_{\max}} = 9 \frac{\text{m}}{\text{s}}$. The tracking performance of this test is presented in Figure 6.8. At first glance, we see that this case is basically an extreme case of the performance seen in Figure 6.5. While in the previous test, the vehicle veered to the inside of the turn in a small range of instances, it appears that in this case, the vehicle spends the majority of its time on the inside of the reference system's path. As was said earlier, this behavior is desirable given the design of the controller. At the requested speeds and given parameters, the following distance should be 12.7 m behind the reference system which would cause the vehicle to start turning much earlier in its path, affecting a noticeable "short cutting" of the path.

While it is ideal if the vehicle stays to the inside of the path, we see that as it passes the

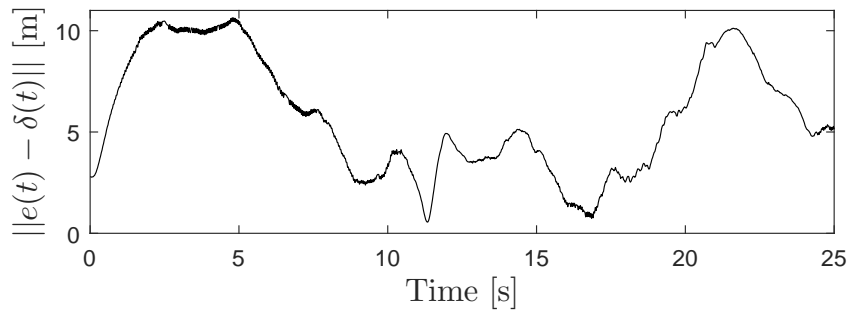


Figure 6.9: Total error, $e_1(t)$, with $k_\omega = 3.5$ and $v_{r_{\max}} = 9$.

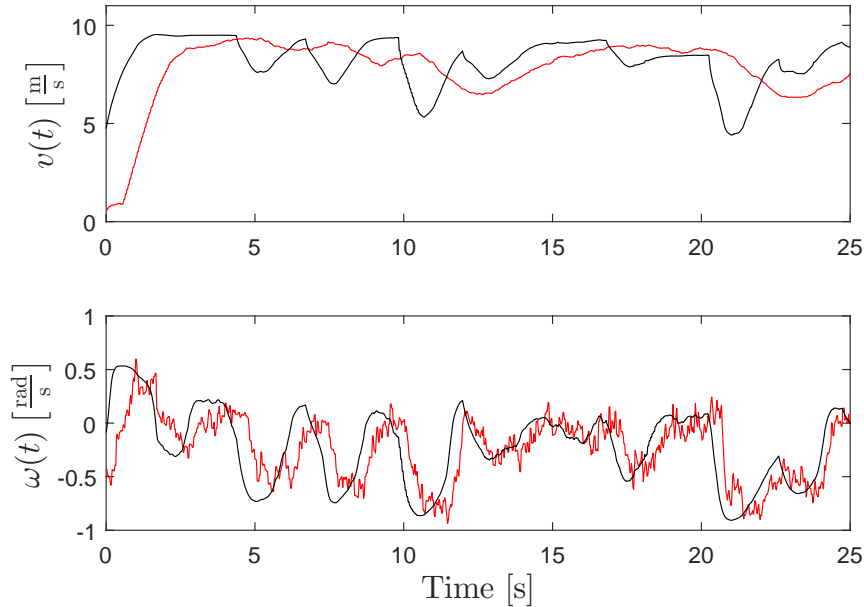


Figure 6.10: Controller commands (black) and measured vehicle response (red) for $k_\omega = 3.5$ and $v_{r_{\max}} = 9$.

waypoint located at approximately $(-40, -8)$, it deviates even more from the reference system than in the previous test. This deviation is substantiated by the error increase seen in Figure 6.9 starting at approximately $t = 18$ s. Somewhat less obvious is that the vehicle has a relatively high amount of error between the waypoints located at $(25, -40)$ and $(18, -65)$, which is seen in Figure 6.9 at roughly $t = 12$ s to $t = 15$ s. Although still inside the path of the reference system, its near intersection is still notable in Figure 6.9.

Looking at Figure 6.10 we see that the vehicle response to commands is not drastically changed from earlier experiments. A key takeaway, however, is that the phase lags are

much more substantial relative to the overall time scope of this experiment. As such, the overall decreased performance at this speed is largely attributed to the relatively large delays in dynamic responses for this experiment. Although we are still capable of tracking the reference system, it is clear that at this level the tracking performance is heavily diminished in comparison to the experiments conducted at $2\frac{\text{m}}{\text{s}}$ and $5\frac{\text{m}}{\text{s}}$.

6.3 Application to Obstacle Avoidance

The control algorithm presented can be conceptualized as a position filter, which smoothly follows a reference system despite sudden changes in direction. Potential fields, on the other hand, have natural obstacle avoidance properties, but they lack smoothness, which is particularly necessary for high speed situations. Given that Ackermann platforms are arguably the best suited to handle high speed usage, it is interesting to assess the algorithms applicability to these scenarios with the Ackermann platform. In this section, we explore this relationship further using the parameters put forth in Table 6.2.

Table 6.2: Ackermann platform parameters for obstacle avoidance experiments. All quantities expressed using standard SI units.

Saturation Constraints	Tuning Parameters	Potential Field Parameters
$v_{\max} = 10$	$k_{\omega} = 1, 2.5, 2.5$	$v_{r_{\max}} = 2, 6, 9$
$v_{\min} = 1$	$k_v = 0.5$	$F_{ac} = 10$
$L_{Ack} = 0.3556$	$\lambda = 1$	$F_{rc} = 6$
$\phi_{\max} = 0.4363$	$\alpha = 1.3$	$W = 2$
$s = 0.01$	$\beta = 1$	$n = 1.5$
	$\epsilon = 0.5$	$E = 14$
	$\zeta_d = 0.85$	
	$\omega_d = 2.5$	

As was the case in the previous section, the speed parameter of the reference system, $v_{r_{\max}}$, and the gain on angular rate, k_{ω} , are changed between each experiment. Figures 6.11 - 6.12 examine the case of obstacle avoidance at low speed with $k_{\omega} = 1$ and $v_{r_{\max}} = 2\frac{\text{m}}{\text{s}}$. In Figures 6.11, the triangles and circles denote the location of the reference system and vehicle, respectively, at certain instances in time. Looking at Figure 6.11, the vehicle tightly tracks the reference system despite a sharp, awkward corner around the obstacle. Since the speed is small, seen in Figure 6.12, this behavior is acceptable. Of particular interest is the fact that the velocity drops to only $1\frac{\text{m}}{\text{s}}$ as the system avoids the obstacle. This is a natural result of the potential field system, which we utilize to obtain an intuitive response from the

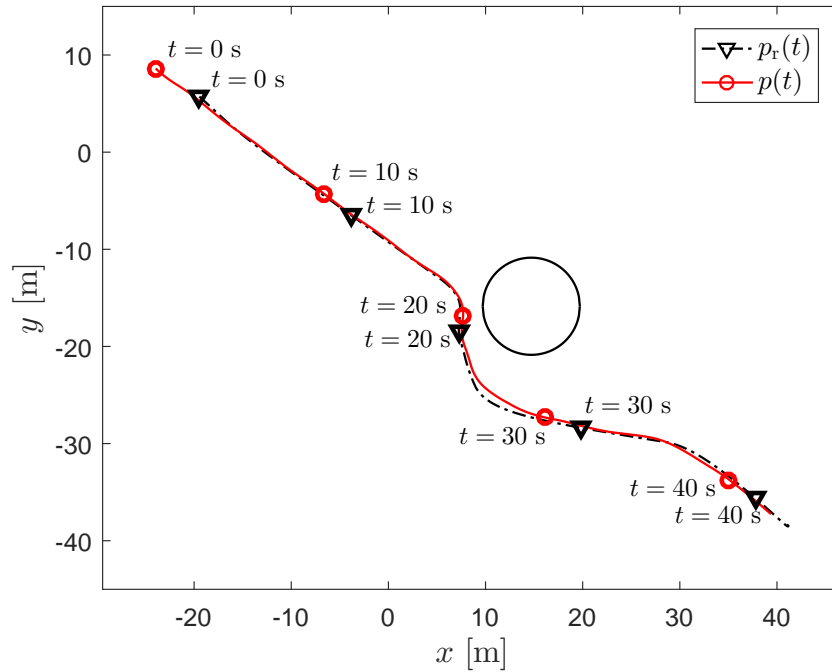


Figure 6.11: Obstacle avoidance at low speeds with $k_\omega = 1$ and $v_{r_{\max}} = 2$.

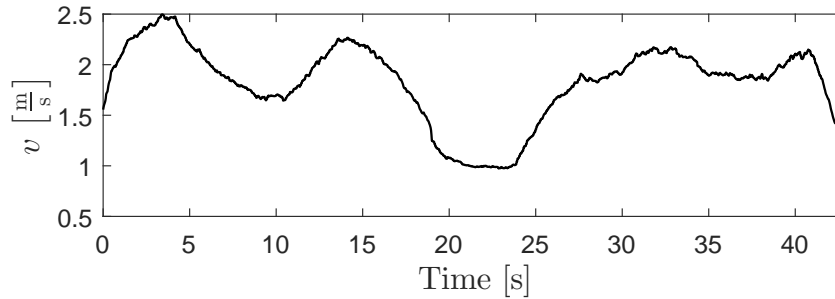


Figure 6.12: Measured vehicle velocity during obstacle avoidance with $k_\omega = 1$ and $v_{r_{\max}} = 2$.

platform. Additionally, we note that this lower velocity is also bounded by the saturation algorithm which maintains at least a $1 \frac{\text{m}}{\text{s}}$ speed. Consequently, by examining the time stamps of Figure 6.11, we see that the distance between the reference and vehicle reduces during the cornering, allowing the vehicle to better track the turn.

The results of obstacle avoidance at medium speeds are given in Figures 6.13 - 6.14. Here the tuning gains are altered such that $k_\omega = 2.5$ and $v_{r_{\max}} = 6 \frac{\text{m}}{\text{s}}$. Like Figure 6.11, the triangles and circles in Figure 6.13 denote the location of the reference system and vehicle,

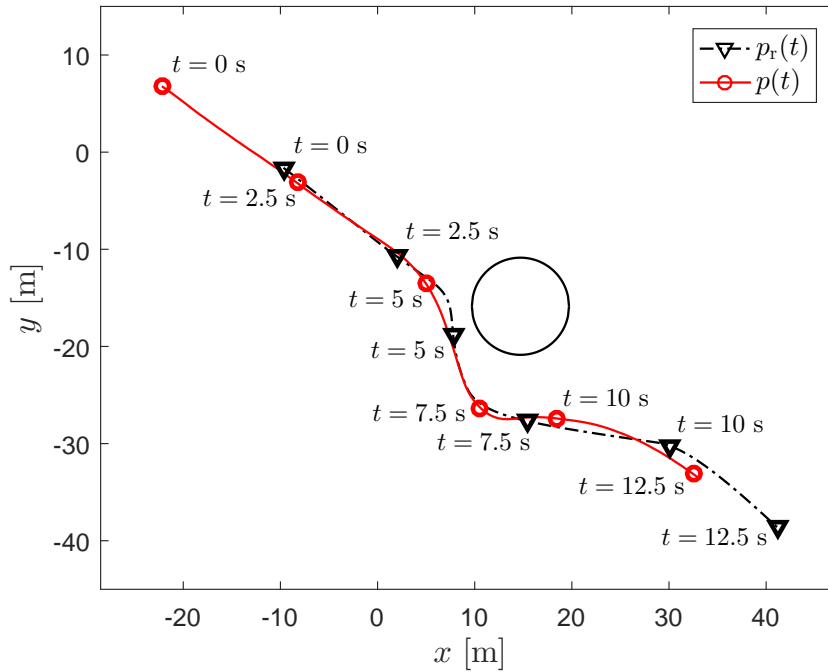


Figure 6.13: Obstacle avoidance at low speeds with $k_\omega = 2.5$ and $v_{r_{\max}} = 6$.

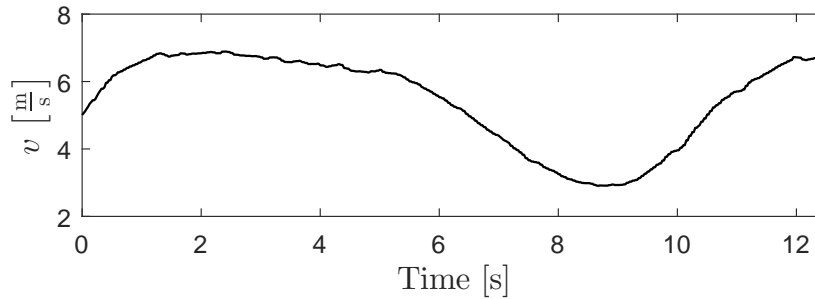


Figure 6.14: Measured vehicle velocity during obstacle avoidance with $k_\omega = 2.5$ and $v_{r_{\max}} = 6$.

respectively, at certain instances in time. As with the medium speed case in the previous section, the vehicle does not follow the reference system as tightly. This behavior, however, has distinct advantages in dealing with obstacles. At the 5 second mark, the vehicle begins to diverge from the reference path in order to avoid the obstacle, as opposed to waiting until the last instant.

By steering earlier, the vehicle achieves a less severe curve during avoidance, facilitating higher speeds throughout. This action is more intuitive for humans as well, as a human

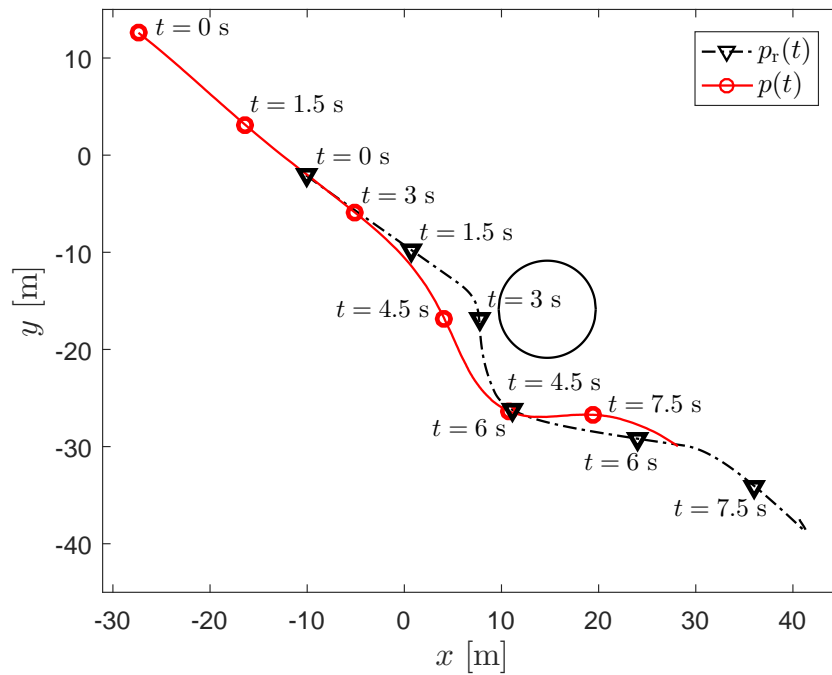


Figure 6.15: Obstacle avoidance at low speeds with $k_\omega = 2.5$ and $v_{r_{\max}} = 9$.

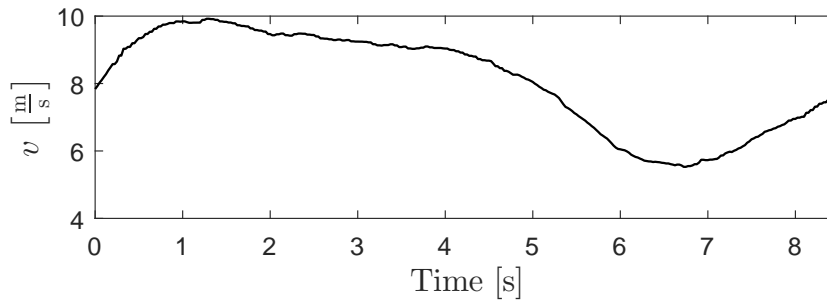


Figure 6.16: Measured vehicle velocity during obstacle avoidance with $k_\omega = 2.5$ and $v_{r_{\max}} = 9$.

driver would avoid immediately upon the observation of an obstacle, rather than waiting. The human's path of avoidance would likewise be based on speed. Looking at Figure 6.14, the speed decreases from $6 \frac{\text{m}}{\text{s}}$ to $3 \frac{\text{m}}{\text{s}}$ during the turn. Again, this is a natural behavior as the average driver would naturally slow down in order to safely avoid the obstacle. By slowing down, the distance decreases, which then facilitates better tracking of the reference system.

Lastly, we consider the case of obstacle avoidance at high speeds, seen in Figures 6.15 - 6.16. For this case, the tuning gains are altered such that $k_\omega = 2.5$ and $v_{r_{\max}} = 9 \frac{\text{m}}{\text{s}}$. Again, the

triangles and circles in Figure 6.15 denote the location of the reference system and vehicle, respectively, at certain instances in time. Looking at Figure 6.15, the algorithm clearly smooths the trajectory of the reference system. In the same way that a human driver may seek to avoid an obstacle at high speeds, the vehicle departs from the reference path at approximately 3 seconds, at a point much earlier than the reference system begins to avoid. During this outward swerve, Figure 6.16 indicates that the velocity begins to drop off at 5 seconds. This drop results in a decrease from $9 \frac{\text{m}}{\text{s}}$ to $6 \frac{\text{m}}{\text{s}}$, a 33% dip opposed to the 50% change in the previous scenario. With a smoother trajectory, the algorithm allows for more preservation of speed than previously seen. Lastly, we note the distance between the reference system and vehicle decreases, allowing for a more aggressive turn after passing the obstacle.

6.4 Summary

In this chapter we present the experimental results obtained using the control algorithm and saturation methods developed in Chapter 2 and Chapter 5, respectively. These tests were performed on a small scale Ackermann steering base platform hosted by DySMAC. In addition to the base vehicle and sensing hardware, low level PI controllers and an Extended Kalman filter were used to facilitate use of the developed control algorithm. As with Chapter 6, potential field algorithms were used to better facilitate testing. Initially three separate tracking tests that span the operational envelope are presented. With regard to previous tests, we see that the Ackermann platform is not nearly as capable of maintaining low position error as the differential platform. In fact, it was not uncommon to see errors grow to as large as 2 m and up to 5+ m in other cases. Through further examination, it seems likely that these errors were caused largely by dynamic effects not considered in control development. Lastly in this chapter, we explore applications to obstacle avoidance which show promising results due to the natural filtering that occurs as the vehicle trails a reference system.

Chapter 7

Conclusions and Future Research

In Chapter 2, we introduce a control algorithm largely developed in [25]. In addition to this algorithm, we place a constraint on the reference system that it must be at least class C^1 in time. This restriction guarantees continuous controller commands and is particularly necessary in the saturation algorithms presented in subsequent chapters. Along with constraints on the reference system, the dynamics of the commanded following distance, $d(t)$, are modified to account for minimum bounds. This modification on the dynamics is used to eliminate numerical issues that may arise during implementation.

Using these modifications, simulation results are given to demonstrate the base behavior of the algorithm. As expected, without modification, the vehicle behaves as a unicycle with no coupling between longitudinal and angular rates. While within the bounds of initial development, this behavior is later shown to be undesirable, especially in dealing with Ackermann steering platforms in which the velocity and angular rate are tightly related. In addition to no coupled behavior in commands, there are no constraints on the magnitude of the commands, which is not typically realistic for the majority of platforms. Lastly in Chapter 2, we examine the effects of the reference system following criteria, a key feature of the control algorithm. We see that the following distance can be chosen to better suit certain platforms, but while this is the case, it does not guarantee acceptable behavior in all instances, further motivating the need for additional modifications.

To better account for platform limitations, in Chapter 3, saturations constraints are introduced into the control algorithm for differential drive based platforms. To develop these constraints, we introduce a basic kinematic model for a differential drive vehicle. Using this model, constraints are developed that relate to vehicle actuator capabilities. These constraints are simplified with the assumption that typical motors have the same limiting speed in both forward and reverse directions. While this assumption is not always true, it is reasonable and can be made to work for a wide variety of applications. From the saturation constraints, an algorithm is developed which focuses on modifying commands to maintain curvature, with the intent that the vehicle will follow the desired trajectory to the best of

its capabilities during saturation. To facilitate this algorithm, methodologies for maintaining Lyapunov stability criteria are introduced that alter the reference system velocity. We conclude Chapter 3 with simulations using the modified algorithm in both states of normal operation and saturated conditions, demonstrating that the algorithm performs as expected by following the desired trajectory to the best of its ability.

To expound on the performance, Chapter 4 presents experimental results obtained using the control algorithm and saturation methods developed in Chapter 2 and Chapter 3, respectively. To conduct these tests, differential steering platforms hosted by DySMAC were used. The hardware on these platforms afforded the ability to implement the algorithm directly without the need for additional observers and other state estimation techniques. To allow testing in a more practical nature, a modified potential field algorithm is introduced to manipulate the reference system. Ultimately, three tests are presented that span the envelope of the vehicles. Through the discussion of these results, we see that the response is highly favorable with position error converging to within 20 cm of the intended position and typically residing at less than 10 cm. In terms of the minor discrepancies seen, these are generally attributed to actual vehicle dynamic constraints not factored into controller design.

While differential platforms cover a large range of vehicles in many mobile robotic applications, the majority of vehicle platforms in general are Ackermann based. In Chapter 5, saturation constraints are introduced based on Ackermann steering platform kinematics. By examining a bicycle kinematic model, constraints and a desired command envelope are determined which is based on typical vehicle limiting factors such as speed and steering angle. As is done in Chapter 3, a saturation methodology is developed that focuses on mapping commands from the base controller back into this desired envelope. This task is accomplished through a dual stage algorithm that focuses on preserving directional intent, desired steering curvature, and desired lateral acceleration when possible. Again, as seen in Chapter 3, the reference system is altered so that Lyapunov stability criteria are still met. To examine the efficacy of this design, Chapter 5 concludes with simulation results that demonstrate the intended behavior for Ackermann platforms.

In Chapter 6 we present the experimental results obtained using the control algorithm and saturation methods developed in Chapter 2 and Chapter 5, respectively. These tests were performed on a small scale Ackermann steering base platform hosted by DySMAC. In addition to the base vehicle and sensing hardware, low level PI controllers and an Extended Kalman filter were used to facilitate use of the developed control algorithm. As with Chapter 6, potential field algorithms were used to better facilitate testing. Initially three separate tracking tests that span the operational envelope are presented. With regard to previous tests, we see that the Ackermann platform is not nearly as capable of maintaining low position error as the differential platform. In fact, it was not uncommon to see errors grow to as large as 2 m and up to 5+ m in other cases. Through further examination, it seems likely that these errors were caused largely by dynamic effects not considered in control development. While these errors were mildly present in the differential case, they are much more exaggerated by the Ackermann platform. Lastly in this chapter, we explore applications to

obstacle avoidance which show promising results due to the natural filtering that occurs as the vehicle trails a reference system.

Overall, the control algorithm developed in this work shows promise for both differential and Ackermann steering based platforms. Through both simulation and experiment, successful position control is demonstrated that offers relatively low tracking error. Moreover, the control algorithm itself has proven practical in a wide variety of situations and been used to directly contribute to the publication of [1, 2, 3]. Currently, the main drawback of this algorithm is the lack of vehicle dynamic characteristics taken into account during the design. As such, future work will focus on using backstepping methods to better account for these phenomenon. Additionally, significant effort will be put toward adding adaptive capabilities to future versions to better account for a wide range of vehicle applications.

Chapter 8

Bibliography

- [1] A. Shoemaker and A. Leonessa, “Bio-inspired nonholonomic tracking control,” in *ASME 2013 Dynamic Systems and Control Conference*, pp. V001T07A004–V001T07A004, American Society of Mechanical Engineers, 2013.
- [2] A. Shoemaker and A. Leonessa, “Bioinspired tracking control of high speed nonholonomic ground vehicles,” *Journal of Robotics*, vol. 2015, p. 11, 2015.
- [3] G. Christie, A. Shoemaker, K. Kochersberger, P. Tokekar, L. McLean, and A. Leonessa, “Radiation search operations using scene understanding with autonomous uav and ugv,” *arXiv preprint arXiv:1609.00017*, 2016.
- [4] L. Quaiyum, “Model reference adaptive backstepping control of an autonomous ground vehicle,” Master’s thesis, Virginia Polytechnic Institute and State University, 2015.
- [5] M. Pivtoraiko and A. Kelly, “Efficient constrained path planning via search in state lattices,” in *International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, pp. 1–7, 2005.
- [6] M. Pivtoraiko and A. Kelly, “Kinodynamic motion planning with state lattice motion primitives,” in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2172–2179, Sept 2011.
- [7] S. Pancanti, L. Pallottino, D. Salvadorini, and A. Bicchi, “Motion planning through symbols and lattices,” in *Robotics and Automation, 2004. Proceedings. ICRA ’04. 2004 IEEE International Conference on*, vol. 4, pp. 3914–3919 Vol.4, April 2004.
- [8] N. Limpert, S. Schiffer, and A. Ferrein, “A local planner for ackermann-driven vehicles in ros sbpl,” in *Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech), 2015*, pp. 172–177, Nov 2015.

- [9] S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 354–363, 2005.
- [10] D. Ferguson, M. Likhachev, and A. Stentz, "A guide to heuristic-based path planning," in *Proceedings of the international workshop on planning under uncertainty for autonomous systems, international conference on automated planning and scheduling (ICAPS)*, pp. 9–18, 2005.
- [11] A. Kelly and B. Nagy, "Reactive nonholonomic trajectory generation via parametric optimal control," *The International Journal of Robotics Research*, vol. 22, no. 7-8, pp. 583–601, 2003.
- [12] J. Reuter, "Mobile robots trajectories with continuously differentiable curvature: an optimal control approach," in *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, vol. 1, pp. 38–43 vol.1, Oct 1998.
- [13] F. Biral, M. D. Lio, and E. Bertolazzi, "Combining safety margins and user preferences into a driving criterion for optimal control-based computation of reference maneuvers for an adas of the next generation," in *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, pp. 36–41, June 2005.
- [14] R. Y. Hindiyeh and J. C. Gerdes, "Design of a dynamic surface controller for vehicle sideslip angle during autonomous drifting," *IFAC Proceedings Volumes*, vol. 43, no. 7, pp. 560–565, 2010.
- [15] R. Y. Hindiyeh and J. C. Gerdes, "A controller framework for autonomous drifting: Design, stability, and experimental validation," *Journal of Dynamic Systems, Measurement, and Control*, vol. 136, no. 5, p. 051015, 2014.
- [16] T. Hiraoka, O. Nishihara, and H. Kumamoto, "Automatic path-tracking controller of a four-wheel steering vehicle," *Vehicle System Dynamics*, vol. 47, no. 10, pp. 1205–1227, 2009.
- [17] J. Borenstein and Y. Koren, "High-speed obstacle avoidance for mobile robots," in *Intelligent Control, 1988. Proceedings., IEEE International Symposium on*, pp. 382–384, Aug 1988.
- [18] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 19, pp. 1179–1187, Sept 1989.
- [19] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *IEEE Conference on Robotics and Automation*, pp. 1398–1404, April 1991.

- [20] S. Waydo and R. M. Murray, "Vehicle motion planning using stream functions," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 2, pp. 2484–2491, IEEE, 2003.
- [21] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," in *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pp. 384–389, IEEE, 1990.
- [22] C. C. De Wit, H. Khenouf, C. Samson, and O. J. Sordalen, "Nonlinear control design for mobile robots," *Recent trends in mobile robots*, vol. 11, pp. 121–156, 1993.
- [23] D. DeVon and T. Bretl, "Kinematic and dynamic control of a wheeled mobile robot," in *International Conference on Intelligent Robots and Systems*, 2007.
- [24] R. Carona, A. P. Aguiar, and J. Gaspar, "Control of unicycle type robots," in *IV Jornadas de Engenharia Electronica e Telecomunicacoes e de Computadores*, pp. 180–185, November 2008.
- [25] V. Medina-Garciadiego and A. Leonessa, "Tracking control of a nonholonomic ground vehicle," in *American Control Conference*, June 2011.
- [26] J. Ryu, E. J. Rosseter, and J. C. Gerdes, "Vehicle sideslip and roll parameter estimation using gps," in *Proceedings of the AVEC International Symposium on Advanced Vehicle Control*, 2002.
- [27] J. Li and J. Yi, "Vehicle motion stability with two vehicle dynamics models," in *ASME 2011 Dynamic Systems and Control Conference and Bath/ASME Symposium on Fluid Power and Motion Control*, pp. 893–900, American Society of Mechanical Engineers, 2011.
- [28] E. Velenis, D. Katzourakis, E. Frazzoli, P. Tsiotras, and R. Happee, "Steady-state drifting stabilization of rwd vehicles," *Control Engineering Practice*, vol. 19, pp. 1363–1376, August 2011.
- [29] R. Fierro and F. Lewis, "Control of a nonholonomic mobile robot: Backstepping kinematics into dynamics," *Journal of Robotic Systems*, vol. 14, pp. 149–163, 1997.
- [30] K. Pathak and S. Agrawal, "Planning and control of a nonholonomic unicycle using ring shaped local potential fields," in *American Control Conference, 2004. Proceedings of the 2004*, vol. 3, pp. 2368–2373 vol.3, June 2004.
- [31] C. E. Beal, C. G. Bobier, and J. C. Gerdes, "Controlling vehicle instability through stable handling envelopes," in *ASME 2011 Dynamic Systems and Control Conference and Bath/ASME Symposium on Fluid Power and Motion Control*, pp. 861–868, American Society of Mechanical Engineers, 2011.

- [32] M. Gerard, M. Corno, M. Verhaegen, and E. Holweg, “Force-based abs control using lateral force measurement,” in *ASME 2011 Dynamic Systems and Control Conference and Bath/ASME Symposium on Fluid Power and Motion Control*, pp. 831–837, American Society of Mechanical Engineers, 2011.