Dynamic path following controllers for planar mobile robots

by

Adeel Akhtar

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2011

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In the field of mobile robotics, many applications require feedback control laws that provide perfect path following. Previous work has shown that transverse feedback linearization is an effective approach to designing path following controllers that achieve perfect path following and path invariance. This thesis uses transverse feedback linearization and augments it with dynamic extension to present a framework for designing path following controllers for certain kinematic models of mobile robots. This approach can be used to design path following controllers for a large class of paths. While transverse feedback linearization makes the desired path attractive and invariant, dynamic extension allows the closed-loop system to achieve the desired motion along the path. In particular, dynamic extension can be used to make the mobile robot track a desired velocity or acceleration profile while moving along a path.

Acknowledgements

I would like extend my thanks first and foremost to my supervisor, Dr. Christopher Nielsen for all of his support, suggestions, our many insightful conversations, and the amazing opportunity I have had to work with him and be a part of the Control System Group at the University of Waterloo. I could not have asked for a better arrangement. I am grateful to all the people who helped me in my research, specially Dr. David Wang, Dr. Daniel Miller, Dr. Dana Kulic, Dr. John Thistle and the rest of the Control faculty. I sincerely appreciate all the guidance of Dr. Julie Vale and Darrell Gaudette. Furthermore I appreciate the love, patience, and support of my family and my parents, especially my brother. I would be remiss if I did not also acknowledge my friends Sajid Saleem, Abdul Rehman and Devin Cass for their help and support in completing this thesis.

Dedication

To the four pillars of my life: Almighty Allah, my parents, my siblings and my friends. Without you, my life would fall apart.

I might not know where the lifes road will take me, but I believe Allah has guided me to the right path, in the right direction, and has given me strength to overcome all the difficulties that I have faced.

To my parents, you have given me so much, thank you for your faith in me and for teaching me that I should never surrender.

To my siblings, without your love and support I would not be able to make it.

To my friends, Sajid Saleem, Abdul Rehman, and Devin Cass you have given me motivation when I needed it the most.

We made it!

Table of Contents

List of Tables **ix**

List of Figures **x**

1 Introduction **1**

1.1 Motivation 1

1.2 Literature Review 3

1.2.1 Path Following Using Frenet-Serret Frames 5

1.2.2 Differential Flatness 6

1.2.3 Chain Form 7

1.2.4 Feedback Linearization and Partial Feedback Linearization 7

1.3 Problem Formulation 9

1.4 Organization and Contribution 10

2 Path Following Controllers: Unicycle **11**

2.1 Transverse Feedback Linearization of a Unicycle Following a Circular Path 11

2.1.1 Simulation Results 18

2.2 DTFL of Unicycle Following a Circular Path 18

2.3 Comparison to Other Control Techniques 26

2.3.1 Trajectory Tracking Controller 26

2.3.2 Path Following Using Sliding Mode Control Theory 29

3 Path Following Controllers: Car-like Robot **32**
3.1 Model of a Car-like Robot . . . 33
3.2 Dynamic Extension . . . 34
3.3 Path Following Control Design . . . 36
3.3.1 Transversal and Tangential Controller Design . . . 41
3.4 Implementation Issues . . . 41
3.4.1 Computation of Transversal States . . . 42
3.4.2 Computation of Tangential States . . . 42
3.4.3 Experimental Implementation . . . 45
3.5 Simulation Results . . . 46
3.5.1 Simulation I . . . 47
3.5.2 Simulation II . . . 49
3.6 Robustness of the Proposed Controller . . . 50

4 Approximate Feedback Linearization **56**
4.1 Model of the Standard 1-trailer System . . . 57
4.2 Review of Approximate Feedback Linearization . . . 57
4.3 Approximate Transverse Feedback Linearization . . . 59
4.3.1 Simulation Results . . . 63
4.4 Approximate Dynamic Transverse Feedback Linearization . . . 65
4.4.1 Simulation Results . . . 66
4.5 Conclusion . . . 69

5 Curve Approximation **70**
5.1 Introduction . . . 70
5.2 General Problem . . . 71
5.3 Special Cases of the General Problem . . . 71
5.3.1 Proposed Solution . . . 72
5.4 Solution to the General Problem . . . 77
5.4.1 Implicitization . . . 78
5.4.2 Sylvester Matrix Elimination Method . . . 78

6 Conclusion and Future Work **85**

APPENDICES **86**

A Basic Concepts **87**

A.1 Review of Algebra, Analysis and Differential Geometry 87

A.1.1 Vector fields and their Derivatives 89

A.2 Nonlinear Control Systems 90

A.2.1 Feedback Linearization 91

A.3 Curve Approximation 96

B Matlab Codes **99**

B.1 Transverse Feedback Linearization: code 99

B.2 Dynamic Transverse Feedback Linearization: code 100

Bibliography **107**

List of Tables

5.1 Approximation results of curve (5.8) with $\epsilon = 0.25$. 75
5.2 Approximation results of curve (5.8) with $\epsilon = 0.05$. 76
5.3 Approximation results of curve (5.9) with $\epsilon = 0.45$. 76
5.4 Approximation results of curve (5.9) with $\epsilon = 0.15$. 77
5.5 $\epsilon = 0.25$ 83
5.6 $\epsilon = 0.25$ 83
5.7 Approximation results of the curve (5.22) 83
5.8 $\epsilon = 0.25$ 84
5.9 $\epsilon = 0.25$ 84
5.10 Approximation results of the curve (5.22) 84

List of Figures

1.1 Flow Chart for Motion Control 3
1.2 Frenet-Serret Frames 5

2.1 The kinematic model of Unicycle. 12
2.2 Lift of γ to $\mathbb{R}^3$. 13
2.3 Path Following Manifold, $\Gamma^\star \subset \Gamma$. 15
2.4 Unicycle (2.4) following a circular path 18
2.5 Unicycle (2.4) following non closed paths 19
2.6 DTFL of Unicycle robot (2.1) 24
2.7 Unicycle robot (2.13) following a circular path 25
2.8 Block diagram of Feedback Linearization 26
2.9 Comparison between path following and trajectory tracking 28
2.10 Comparison to sliding mode control 30

3.1 The kinematic model of the car-like robot. 33
3.2 Argument that minimizes the distance from the curve. 37
3.3 Representation of vector $\sigma'(\lambda^\star)$ orthogonal to $R_{\frac{\pi}{2}} ds^\top_{h(x^\star)}$. 38
3.4 Feedback control system of car-like robot with equation references. 42
3.5 Car-like robot (3.6) following sinusoidal curve and tracking profile (3.26). . 48
3.6 Car-like robot (3.6) following sinusoidal curve and tracking profile (3.27). . 49
3.7 Car-like robot (3.6) following sinusoidal curve and tracking profile (3.28). . 50
3.8 Car-like robot 3.6 following the curve (3.4.1) and tracking $\eta_2^{ref} = 0.5$. . . . 51
3.9 Robustness test of car-like robot. 55

4.1 The kinematic model of standard 1-trailer system. 58
4.2 1-trailer system (4.5) initialized at x_{01}. 64
4.3 1-trailer system (4.5) initialized at x_{02} 65
4.4 Entries of Decoupling Matrix (4.18) 67
4.5 Determinant of decoupling matrix (4.18). 68
4.6 1-trailer system (4.15) initialized at x_{01} 68
4.7 1-trailer system (4.15) initialized at x_{02} 69

5.1 Approximation of curve (5.8) with $\epsilon = 0.25$. 75
5.2 Approximation of curve (5.8) with $\epsilon = 0.05$. 76
5.3 Approximation of curve (5.9) with $\epsilon = 0.45$. 76
5.4 Approximation of curve (5.9) with $\epsilon = 0.15$. 77
5.5 Approximation of curve (5.22) 82
5.6 Approximation of curve (5.23) 83

Chapter 1

Introduction

In this thesis, we study the path following problem for mobile robots. Informally, the path following problem in control theory entails designing control laws to make a system's output approach and move along a pre-specified path with no *a priori* timing law, and to do so starting from initial conditions in a neighborhood of the path.

The problem of maintaining accurate motion along a specified path is one of the major control specifications in the field of mobile robotics and can be roughly classified as either (i) a path following problem or (ii) a reference tracking problem. In a path following problem, the main task of the controller is to follow a path with no *a priori* time parameterization. In a reference tracking problem, the task of the controller is to follow a path with a pre-specified timing law associated with the motion. Thus, tracking can be thought of as a special case of path following. One obvious advantage of path following is that cases exist where the trajectory tracking problem is unsolvable; yet the associated path following problem has a solution [41]. However, the main advantage of adopting the path following approach is that the resulting feedback guarantees the invariance of path. This means that, if the mobile robot is initialized on the path with the appropriate orientation, invariance of path guarantees that it will never leave the path. A trajectory tracking controller cannot ensure path invariance and thus cannot guarantee that the mobile robot will never leave the path [20]. This thesis examines the path following problem for a unicycle, a car-like robot, and a trailer system driven by a car-like robot.

1.1 Motivation

Precise path following is especially desirable when multiple robots must perform a task in tight spatial conditions without colliding, such as small robots moving within a room, or autonomous underwater vehicles moving together from one point in the ocean to another or

even unmanned aerial vehicles working around each other in space. In each of these cases, if there exists defined spatial paths for each of the mobile robots such that the paths do not intersect, then by simply following those paths the robots can be navigated to the intended positions without collision. Precise path following is desirable not only in the use of multiple robots but also for obstacle avoidance using a single robot. More generally, path following is also useful in robotic welding, cutting, drawing, and indoor and outdoor navigation. To further motivate the study of path following, consider the following application examples which illustrate situations where path following is preferable to trajectory tracking.

Example: Car on Road

Consider the application of a path following controller to a car traveling along a road. A roof-mounted camera provides information about the approaching road from which the mathematical representation of a segment of road, such as a spline, can be extracted. This is called the desired path. As discussed above, a tracking controller does not guarantee invariance of the desired path, thus making a tight turn, a tracking controller may lead to unacceptable deviations from the road. The spline determined by the camera can be defined in such that the initial position of the car is on one end of the spline. Since the car is initialized on the path, invariance of the path ensures that the car will never leave the road. In other words, the result is accurate following of the road ahead.

Suppose there is a path defined from a person's home to their office. Suppose the problem is setup as a trajectory tracking problem i.e, we want to follow a virtual reference point that moves along the path and its evolution along the path is a function of time. Suppose that for some reason, such as a road block, the person has to stop on the way to the office. Since the virtual target is parameterized by time it will not stop and it will move with its predefined timing law. If the vehicle does stop, then once the road becomes clear the virtual target will be far ahead and the car will need to either use a shortcut to catch the virtual target or speed up. An available shortcut may not be available and increasing speed may exceed the speed limit which is undesirable. If a path following controller were designed for this application, the path would have been invariant, meaning the car will stay on the path. Secondly by following a speed profile bounded by the speed limit of the road, a speeding ticket can be easily avoided. The car may not reach the office as early as expected, but speeding rules will not be broken.

Example: Computer Numeric Controlled Machines

Consider the application of a path following controller to the automation industry. In computer numeric controlled (CNC) machines like cutting machines, drawing machines, and welding machines the task of the tool is to first approach the desired path and then

cut or weld the job accurately. If the boundary of the part is considered as a path then cutting or welding the part can be considered as a path following problem. If the path can be made invariant, this ensures that once the tool is on the path it will always remain on that path even if it encounters a burr or rough spot that may cause it to slow down.

The above discussion illustrates the suitability of path following for mobile robots and other application areas. There are many more examples where path following controllers are appropriate for mechanical or robotic system when compared to trajectory tracking controllers. These examples include robotic deburring, walking robots, exercise and rehabilitation machines, teleoperations, obstacle avoidance, human robot interaction and robotic manipulators [25].

1.2 Literature Review

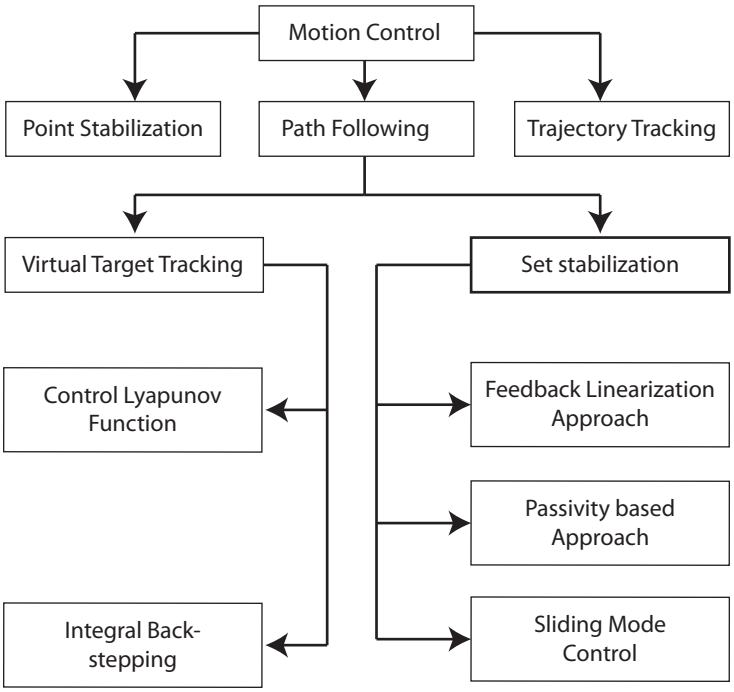


Figure 1.1: Flow Chart for Motion Control

In the last two decades there has been considerable attention paid to the motion control problem, especially in the field of mobile robotics. The problem of the motion control can be classified into three main groups (i) point stabilization (ii) path following (iii) trajectory tracking; see Figure 1.1.

Point stabilization or equilibrium stabilization has been studied extensively for nonholonomic systems. A nonholonomic system is a system with nonholonomic constraint. For example, a unicycle or a car-like robot has a nonholonomic constraint because the robot cannot moves sideways as the wheel rotates only in forward or backward direction but cannot move sideways without slipping. Brockett presents necessary conditions that show that no time-invariant, continuous, feedback stabilizer exists for these systems [11]. The most common approaches used for equilibrium stabilization are discontinuous control and hybrid feedback control [36]. Aicardi et al. [36] showed that, for a unicycle type robot, Brockett's necessary condition can be avoided if a different state space representation is used instead of Cartesian coordinates. There can be an infinite number of different state space representations of a system. A few commonly used in the literature for mobile robots are polar coordinates and Frenet-Serret coordinates representations [32]. Bushnell et al. [13] presented an open loop control law to solve the point stabilization problem for nonholonomic systems.

In this thesis, we study the path following problem. It is important to understand the difference between path following and trajectory tracking. Aguiar et al. [2] highlighted the fundamental difference between path following and reference tracking, i.e in path following, the control objective is to follow a geometric path without a timing law assigned to it. Path following can even be used when a system has unstable zero dynamics. In the case of path following for a linear system with unstable zero dynamics, better performance can be achieved when compared to tracking [2].

A large number of path following control laws are available in the literature that use the concept of control Lyapunov functions [54]. The main disadvantage of using the Lyapunov based controller is that the process is ad hoc. This means that any change in the system or path requires finding a new Lyapunov function which can be very difficult. Sliding mode controllers are another way of solving the path following problem. Sliding mode controller design involves finding a sliding surface on which the system exhibits desirable behavior. Once a sliding surface is identified the designers seek a feedback controller so that the system trajectories converges to the sliding surface in finite time. A comparison between the path following controller designed in this thesis is compared with other existing controllers is presented in Section 2.3. Dagci et al. [17] proposes a controller for the unicycle using sliding mode control theory. One of the main advantages of sliding mode control is that it is robust to modeling errors. The sliding mode controller consists of a discontinuous feedback control that switches on the desired path. Ideally, the switching of control occurs at an infinitely high frequency to eliminate deviations from the path. In practice, the frequency cannot be infinitely fast and so it induces high frequency oscillations, also called chattering, in the control signal. Such control chattering is undesirable since it can damage actuators. Sliding mode path following controllers do not guarantee path invariance. Another approach for path following is passivity based path following technique

using a set stabilization approach. El-Hawwary and Maggiore [20] almost globally solved the problem of path following for the unicycle along the circle using a passivity based control law. Similar to the Lyapunov based techniques, generalizing this method to other systems and arbitrary paths can be very difficult.

Another approach for path following is to express the system in a simpler state space form that suggest easy design of control law. We give a brief overview of some of the interesting coordinate transformation techniques.

1.2.1 Path Following Using Frenet-Serret Frames

Many path following problems in the literature are analyzed under the setting of Frenet-Serret frames (FS frames). The most common approach is to convert the system from Cartesian coordinates to FS coordinates. The Frenet-Serret formulas describe a particle which moves along a differentiable path in three dimensional space. The formulas describe the derivatives of the so called tangent, normal, and binormal unit vectors in terms of each other. A FS frame is an orthonormal basis $\{T, N, B\}$ of $\mathbb{R}^3$ which moves along the path, see Figure 1.2. Here T is the unit vector tangent to the curve, pointing in the direction of motion, N is the derivative of T with respect to the arc length parameter of the curve, divided by its length and B is the cross product of T and N. The evolution of a FS frame

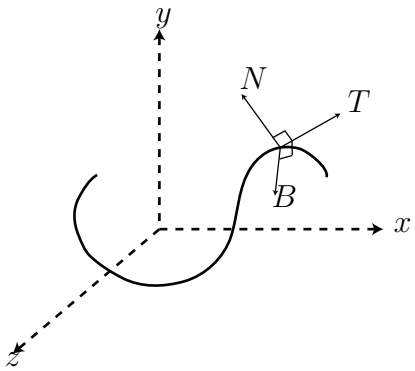


Figure 1.2: Frenet-Serret Frames

is completely determined by the curvature and torsion of the path via the FS formulas. Interested readers are referred to [45]. One drawback in representing a nonholonomic system like the unicycle in FS coordinates (e.g., Micaelli et al. [1],) is that the coordinate transformation is not global and singularities are introduced in the model. Singularities arise when the position of the virtual vehicle is defined simply by projecting the position of the actual vehicle to the closest point on the path. In other words, a singularity occurs when the vehicle is located at the center of curvature of the path. In this situation, the

closest point from the vehicle to the path is not unique. The approach followed in this thesis suffers from a similar limitation. Lapierre et al. [32] shows that by explicitly controlling the rate of progression of a virtual target to be tracked along the path and treating it as a new control input, a singularity can be avoided. Lapierre et al. gives a Lyapunov based control law for path following for the unicycle. In general, it is not easy to find candidate Lyapunov functions for generic paths and for general nonholonomic system or systems with multiple inputs.

1.2.2 Differential Flatness

A useful property of nonlinear system specifically from the point of view of path following is flatness. Fliess et al. [21] introduced the notion of differential flatness. A large class of nonlinear systems fall in this category. Roughly speaking, a nonlinear system is differentially flat if there exist a set of outputs (equal to the number of the inputs) such that all states and inputs can be uniquely determined from the desired output. More precisely, if the system has states $x \in \mathbb{R}^n$, and the inputs $u \in \mathbb{R}^m$ then the system is flat if we can find outputs $y \in \mathbb{R}^m$ of the form,

$$y = y(x, u, \dot{u}, \cdots, u^{(p)}) \tag{1.1}$$

such that,

$$x = x(y, \dot{y}, \cdots, u^{(q)})$$
$$u = (y, \dot{y}, \cdots, u^{(q)})$$

where $y^{(i)} := \frac{d^i y(t)}{dt^i}$ and $u^{(i)} := \frac{d^i u(t)}{dt^i}$. Differentially flat systems were first introduced in [21] using differential algebra and later described using a Lie-Bäcklund transformation [22]. In [59] differential flatness was introduced under the setting of differential geometry. Finding a flat output involves finding a function that satisfies the conditions given in [47]. The search for a flat output can be simplified by noting that they often have strong geometric interpretations [55]. For example in case of unicycle and car-like robot, presented in the following chapters, we use outputs that are physically meaningful and the outputs turnout to be differentially flat. From a trajectory tracking point of view, differentially flat systems are useful. Since the behavior of the flat system and its inputs are completely determined by the flat outputs, trajectories can be planned in the output space, and can then be mapped to appropriate inputs. It is shown in the literature that the standard n-trailer system, discussed in Chapter 4, pulled by a car-like robot, is a differentially flat system [47]. The procedure to find a flat output for a general system is still unknown. Murray et al. has shown that every system which is feedback linearizable via dynamic extension is differentially flat [40].

1.2.3 Chain Form

Bushnell et al. [13] give sufficient conditions for converting multiple input systems to a simpler chain form, through a local coordinate transformation. For a drift free[1] control system of the form

$$\dot{\xi} = g_0(\xi)u_0 + g_1(\xi)u_1 + \cdots + g_m(\xi)u_m, \tag{1.2}$$

where, $\xi \in \mathbb{R}^n$ are the states, $u(\xi, t) \in \mathbb{R}^m$ are the inputs and g_i are the smooth linearly independent vector fields, Walsh et al. gave necessary and sufficient conditions under which the system (1.2) can be transformed into chain form [61]. In chain form (1.2) reads,

$$\begin{array}{cccc} \dot{x}_0^0 = v_0, & \dot{x}_1^0 = v_1, & \dot{x}_2^0 = v_2, & \cdots, \quad \dot{x}_m^0 = v_m, \\ & \dot{x}_{10}^1 = x_1^0 v_0, & \dot{x}_{20}^1 = x_2^0 v_0, & \cdots, \quad \dot{x}_{m0}^1 = x_m^0 v_0, \\ & \vdots & \vdots & \ddots \quad \vdots \\ & \dot{x}_{10}^{n_1} = x_{10}^{n_1-1} v_0, & \dot{x}_{20}^{n_2} = x_{20}^{n_2-1} v_0, & \cdots, \quad \dot{x}_{m0}^{n_m} = x_{m0}^{n_m-1} v_0, \end{array}$$

the chain form provides some advantages in analysis as the system becomes less complicated as compared to the original form. In [13] the authors give sufficient conditions for transforming a three-input driftless system into a chain form via a coordinate transformation and state feedback. In the chain form the system was shown to be completely controllable. The control law proposed in [13] is an open-loop control law that is not robust to plant modeling uncertainties. The authors in [35] propose the integration of path planning and feedback control design. The procedure to convert a general system to chain form is an unsolved problem.

1.2.4 Feedback Linearization and Partial Feedback Linearization

A lot of work has been done in the field on nonlinear control using the concept of feedback linearization. We consider feedback linearization from the point of view of path following. Skjetne et al. [53] proposed a method for path following for a class of general nonlinear systems. The authors in [53] divide the path following problem into two tasks: a geometric task and a dynamic task. The geometric task forces the system to converge to the desired path while the dynamic task involves objectives such as tracking speed or velocity profiles. One of the drawbacks in the work [53] is that the system must be feedback linearizable. Since most mobile robots are not fully feedback linearizable, this work is not directly applicable to mobile robots.

When feasible, exact feedback linearization allows us to represent a given nonlinear system as a fully linear system using a coordinate and feedback transformation. Necessary

[1] Informally, a drift free system stops when all the control inputs are set to zero.

and sufficient conditions for a system to be feedback linearizable can be found in [29], [51]. It is not always possible to fully feedback linearize a given nonlinear system. If the system is not fully feedback linearizable, it may still be possible to partially feedback linearize the system. An interesting approach to solve path following problem is by using transverse feedback linearization. Banaszuk and Hauser [9] use the notion of transverse feedback linearization to linearize the dynamics of a system transverse to a closed orbit in the state space. They provide necessary and sufficient conditions for generating a coordinate and feedback transformation to accomplish this. The authors in [9] propose an autonomous feedback control providing exponential stability for periodic orbits. Altafini [5] casts the path following problem as an output regulation problem. In [5] a general n-trailer system is considered and it is shown that the trailer system is input-output feedback linearizable. It was further shown that the path following problem can be solved on the partially linearized system.

An interesting approach for path following based on the work in [9] is via transverse feedback linearization. In [43], the authors showed that in many cases it possible to make a desired path invariant via transverse feedback linearization. An invariant path means that when the system starts on the path, with velocity tangent to the path, the system will remain on the path for all future time. The authors in [42] provide necessary and sufficient conditions for the linearization of dynamics transverse to the the curve. In [14], the authors consider the path following problem for the planar vertical takeoff and landing of aircraft. A path following controller was designed that follows a class of smooth Jordan curves. The proposed controller enjoys the property of path invariance. It in interesting to note that it was shown using the approach outlined in [14] that the controller works for nonminimum phase systems. In [42], it was shown by the authors that using transverse feedback linearization path following can be achieved for a maglev positioning system. This thesis extends the approach proposed by the authors in [42]. The controller is designed in two steps. First, a transversal controller is designed and then a tangential controller. The transversal controller drives the output of the system to the path, while the tangential controller meets the application specific requirements on the path. Path following controller design for a mechanical systems is discussed in [27]. In this paper, the authors show that using a coordinate transformation and feedback, a mechanical system can be represented in a convenient form that simplifies controller design. An input-output feedback linearization approach is applied to a planar five-bar linkage robot and an underactuated five-bar robot with a flexible link. In this thesis, we will follow the technique of [42] for specific class of systems and arbitrary path. A detailed review of this approach is elaborated in Chapter 2. Moreover, in Section 2.3 we compare our controller with few other existing controllers.

1.3 Problem Formulation

In this thesis, we design path following controllers for kinematic models of a unicycle, a car-like robot and a car with trailers. Some of the terminologies used is defined in Appendix A. A nonlinear system with m inputs and p outputs can be modeled as,

$$\dot{x} = f(x) + \sum_{i=1}^{m} g_i(x)u := f(x) + g(x)u, \tag{1.3}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$, $g_i : \mathbb{R}^n \to \mathbb{R}^n$, $i \in \{1, \ldots, m\}$, are smooth functions[2]. The output[3] of (1.3), with $p = 2$ is modeled by an equation of the form

$$y = h(x) \tag{1.4}$$

with $h : \mathbb{R}^n \to \mathbb{R}^2$ smooth. Suppose we are given a path to follow in the output space $\mathbb{R}^2$ of (1.3) as a regular parameterized curve

$$\begin{aligned} \sigma : \mathbb{D} &\to \mathbb{R}^2 \\ \lambda &\mapsto \begin{bmatrix} \sigma_1(\lambda) \\ \sigma_2(\lambda) \end{bmatrix}, \end{aligned} \tag{1.5}$$

where $\mathbb{D}$ is either $\mathbb{S}$ if the curve is closed or $\mathbb{R}$ if the curve is not closed[4], and $\sigma \in C^r$ is sufficiently smooth. Since σ is regular, without loss of generality, we can assume that it has a unit-speed parameterization, i.e.,

$$(\forall\, \lambda \in \mathbb{D}) \;\; \|\sigma'(\lambda)\| = 1.$$

Under this assumption, the curve σ is parameterized by its arc length. For Jordan curves with finite length L this means that $\mathbb{D} = \mathbb{R} \bmod L$ and σ is L-periodic, i.e., for any $\lambda \in \mathbb{D}$, $\sigma(\lambda + L) = \sigma(L)$. When the curve is not closed $\mathbb{D} = \mathbb{R}$. We impose geometric restrictions on the class of curves considered [26].

Assumption 1. *The path, $\sigma(\mathbb{D})$, is an embedded submanifold of $\mathbb{R}^2$ with dimension 1.*

Assumption 2. *There exists a smooth map $s : \mathbb{R}^2 \to \mathbb{R}^1$ such that 0 is a regular value of s and $\sigma(\mathbb{D}) = s^{-1}(0)$. Let $\gamma := s^{-1}(0)$.*

[2]Informally, functions that have derivatives of all orders are called smooth

[3]We restrict $p = 2$ in this thesis as we are dealing with planer mobile robots.

[4]The notation $\mathbb{S}$ means $\mathbb{R}\text{mod}L$, meaning the curve is periodic with a period of L. Thus $\mathbb{S}$ has the geometric structure of a circle.

Assumption 1 imposes that the path has no self intersections, no "corners", and does not approach itself asymptotically. Assumption 2 asks that the entire path be represented as the zero level set of the function s in the output space of system (1.3). This is always possible, locally, if Assumption 1 holds. We seek a smooth control law such that the closed-loop system satisfies,

PF1 For each initial condition in the neighborhood, the output (1.4) along solutions of the closed-system (1.3) asymptotically approaches the path, i.e., $y(t) \to \sigma(\mathbb{D})$ as $t \to \infty$.

PF2 The level set $s(y)$ is output invariant, i.e., if the system is initialized on the path with the velocity vector tangent to the curve, the system remain on the path $\sigma(\mathbb{D})$ for all $t \geq 0$.

PF3 On the path, the mobile robot tracks a desired velocity or acceleration profile.

1.4 Organization and Contribution

The organization of this thesis is as follows. In Chapter 2 the path following problem for the unicycle is analyzed. A brief overview of the approach from [42] is presented. Transverse feedback linearization for path following is reviewed for the unicycle and a circular path. We then introduce the procedure for transverse feedback linearization with dynamic extension (henceforth dynamic transverse feedback linearization). In Chapter 3, the path following problem is solved for the car-like robot using dynamic transverse feedback linearization for a large class of embedded curves. Chapter 4 investigates the trailer system driven by a car-like robot. An overview of non-regular systems is given. The 1-trailer system is an example of such a system. We show, largely by simulations, that the results of transverse feedback linearization and dynamic transverse feedback linearization are applicable to non-regular systems. In Chapter 5 a procedure is proposed for finding an implicit representation of a given parametric curve. Certain nonlinear control applications, for example, dynamic transverse feedback linearization, require a parametric as well as an implicit representation of the given path. A solution to the implicitization problem is proposed based on Weierstrass approximation theorem and elimination theory.

The main contribution of this thesis is that it gives a solution to path following problem for a large class of curves for the systems such as unicycle and car-like robot via dynamic extension. Moreover, a procedure is outlined to implicitize a given parameterized curve.

Chapter 2

Path Following Controllers: Unicycle

In this chapter we present a solution to the path following problem for the kinematic unicycle. The path following problem is formulated as a special case of a set stabilization problem, where stabilizing an appropriate set in the state space of the mobile robot causes the output of the system to lie on the desired path. In [41], [43] authors stabilize the desired path using transverse feedback linearization. Using a special coordinate and feedback transformation, the authors show that the problem of forcing the system's output to approach the desired path is equivalent to stabilizing a linear time-invariant (LTI) subsystem. When feasible, this approach is attractive because it simplifies control design. However, as we will show in this chapter, one of the limitations of that approach is that it is not possible to control the motion of the unicycle along the path. We outline the deficiency of that approach and then present a procedure to overcome the deficiency with the help of dynamic extension.

2.1 Transverse Feedback Linearization of a Unicycle Following a Circular Path

In [41], [43] the authors propose a path following controller that satisfies objectives **PF1** and **PF2** for the class of curves introduced in Section 1.3. One of the limitations of that controller is that the speed of the robot is fixed along the path. Our work is an extension of [41] that overcomes this limitation. We begin with an illustrative example of a unicycle robot following a circular path of unit radius. We show that the aforementioned limitation is caused by fixing the speed of the unicycle. Later in this chapter we provide a procedure to overcome this limitation. The following example is based on the work in [41].

Consider the kinematic model of a unicycle mobile robot, see Figure 2.1,

$$\dot{x} = \begin{bmatrix} \cos x_3 & 0 \\ \sin x_3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{2.1}$$

where $x \in \mathbb{R}^3$ is the state, the input $v \in \mathbb{R}$ is the translational speed and $\omega \in \mathbb{R}$ is the angular velocity of the steering angle. We take the unicycle's position in the plane as the output of (2.1)

$$y = h(x) = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\top. \tag{2.2}$$

Remark 2.1.1. *We only consider kinematic models in this thesis. A dynamic model of the unicycle robot can be found in many places including [38]. We argue that, from the point of view of controller design, all theoretical issues arise at the kinematic level. Of course, when implementing a control law on a physical system, a dynamic model is necessary. However, since the relationship between a kinematic and dynamic model is essentially an integrator, any controller that works on the kinematic model can be implemented on the dynamic model using, for instance, backstepping [31]. Furthermore, since kinematic models are generally simpler then dynamical models, this choice simplifies the exposition. For these reasons, in this thesis we consider only kinematic models of the mobile robots.*

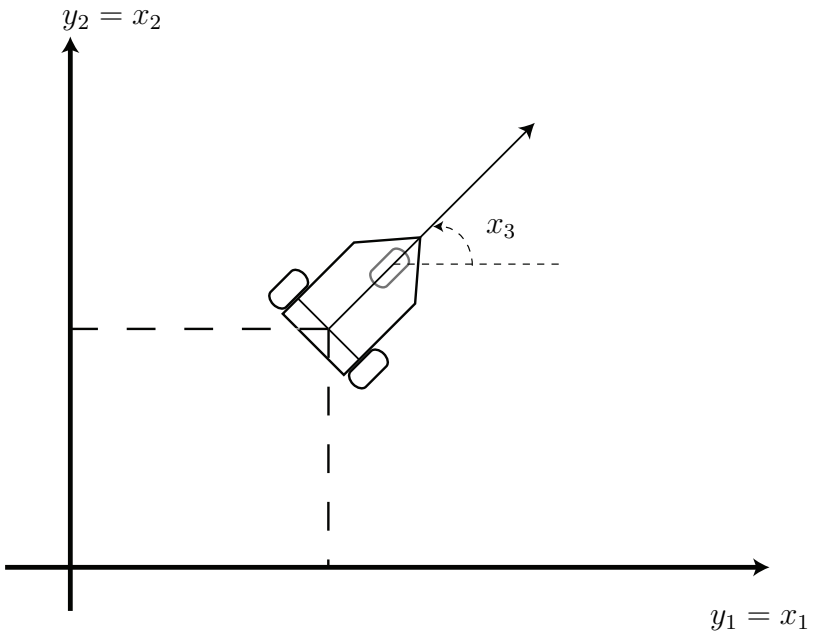


Figure 2.1: The kinematic model of Unicycle.

In this example, the desired path is a unit circle and is given as a regular parameterized curve. It is interesting to note that the path in parameterized by a path parameter λ not specifically time t.

$$\begin{aligned} \sigma : \mathbb{S} &\to \mathbb{R}^2 \\ \lambda &\mapsto \begin{bmatrix} \cos\lambda \\ \sin\lambda \end{bmatrix}, \end{aligned} \tag{2.3}$$

The path is a regular unit-speed curve,

$$(\forall\, \lambda \in \mathbb{S}) \;\; \|\sigma'(\lambda)\| = \sqrt{\cos^2\lambda + \sin^2\lambda} = 1.$$

This curve satisfies Assumption 2 and so there exists a smooth map $s : \mathbb{R}^2 \to \mathbb{R}^1$ such that 0 is a regular value of s and $\sigma(\mathbb{D}) = s^{-1}(0)$. Let $\gamma := s^{-1}(0)$. The map $h : \mathbb{R}^3 \to \mathbb{R}^2$ is transversal [23] to γ and therefore the *lift* of γ to $\mathbb{R}^3$

$$\Gamma := (s \circ h)^{-1}(0) = \left\{ x \in \mathbb{R}^3 : s(h(x)) = 0 \right\}$$

is a submanifold of $\mathbb{R}^3$ having the shape of a cylinder, as shown in Figure 2.2. Define

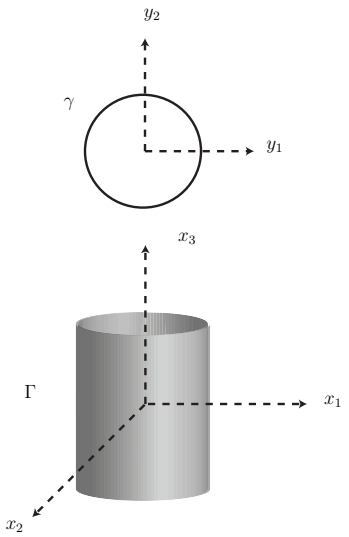


Figure 2.2: Lift of γ to $\mathbb{R}^3$.

$\alpha(x) := s \circ h(x) = x_1^2 + x_2^2 - 1$. Intuitively, making $x \to \Gamma$ is equivalent to making $y \to \gamma$. In [41] the path following problem is solved by fixing the translational speed $v \neq 0$. Let the input $u := \omega$. The system (2.1) can be written as,

$$\dot{x} = \begin{bmatrix} v\cos x_3 \\ v\sin x_3 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u. \tag{2.4}$$

Define

$$f(x) := \begin{bmatrix} v\cos x_3 \\ v\sin x_3 \\ 0 \end{bmatrix}, g(x) := \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},$$

so that (2.4) can be written compactly as $\dot{x} = f(x) + g(x)u$. We start differentiating the function α with respect to time

$$\begin{aligned} \dot{\alpha} &= \frac{\partial \alpha(x)}{\partial x}\dot{x} \\ &= \frac{\partial \alpha(x)}{\partial x}f(x) + \frac{\partial \alpha(x)}{\partial x}g(x)u \\ &= L_f\alpha(x) + L_g\alpha(x)u, \end{aligned}$$

where,

$$\begin{aligned} L_f\alpha(x) &= 2v\left(x_1\cos x_3 + x_2\sin x_3\right), \\ L_g\alpha(x) &= 0. \end{aligned}$$

Differentiating the function $\dot{\alpha}(x)$ with respect to time

$$\begin{aligned} \ddot{\alpha} &= \frac{\partial \dot{\alpha}(x)}{\partial x}\dot{x} \\ &= \frac{\partial L_f\alpha(x)}{\partial x}f(x) + \frac{\partial L_f\alpha(x)}{\partial x}g(x)u \\ &= L_f^2\alpha(x) + L_gL_f\alpha(x)u, \end{aligned}$$

where

$$\begin{aligned} L_f^2\alpha(x) &= 2v^2, \\ L_gL_f\alpha(x) &= 2v(x_2\cos x_3 - x_1\sin x_3). \end{aligned}$$

By assumption $v \neq 0$ and so the function $L_gL_f\alpha(x)$ equals zero if and only if

$$\tan(x_3) = \frac{x_2}{x_1}.$$

Physically, this condition means that the unicycle is oriented along a line passing through the origin of the output space. In other words, the unicycle is normal to the desired path. We claim that this condition cannot occur on the set

$$\begin{aligned} \Gamma^\star &= \{x \in \mathbb{R}^3 : \alpha(x) = \dot{\alpha}(x) = 0\} \\ &= \{x \in \mathbb{R}^3 : x_1^2 + x_2^2 - 1 = x_1\cos x_3 + x_2\sin x_3 = 0\}. \end{aligned}$$

Suppose, by way of contradiction, that the function $L_g L_f \alpha(x) = 0$ on $\Gamma^\star$. Then, since $\dot{\alpha}(x) = 0$, we have that

$$\begin{bmatrix} \cos(x_3) & \sin(x_3) \\ -\sin(x_3) & \cos(x_3) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0.$$

The above 2×2 matrix is nonsingular so the only way that this equation holds is if $x_1 = x_2 = 0$. This is not possible since $x_1^2 + x_2^2 - 1 = 0$ on $\Gamma^\star$. This shows that the function $\alpha(x)$ yields a well defined relative degree of 2 at each point on the set $\Gamma^\star$. The set $\Gamma^\star$ is the path following manifold for this particular example. It is the largest controlled invariant subset of the set Γ. In this case, $\Gamma^\star$ can be visualized as a spring wrapped around a cylinder as shown in Figure 2.3.

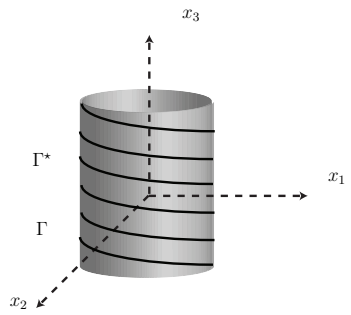


Figure 2.3: Path Following Manifold, $\Gamma^\star \subset \Gamma$.

We will use the function α to stabilize the circle. The output (2.2) of the system (2.4) can be made to converge to the path by making $\alpha(x)$ and $\dot{\alpha}(x)$ converge to zero. Thus path following is essentially a output-zeroing problem, a well know problem in control [51].

The function α and its derivative $\dot{\alpha}$ can be used to partially define a local coordinate transformation for the unicycle. To complete the coordinate transformation, we need a third function. For circular paths one possible choice is [42], [26],

$$\pi(x) := tan^{-1}\left(\frac{x_2}{x_1}\right). \tag{2.5}$$

In order to show that these functions determine a local diffeomorphism, we use the inverse function theorem.

Theorem 2.1.2. *(Inverse Function Theorem [46]) Let U be an open subset of $\mathbb{R}^n$ and $f : U \to \mathbb{R}^n$, a C^∞ mapping. If the Jacobian, $df_{x^\star}$, is nonsingular at some $x^\star$ in U, then there exists an open neighborhood V of $x^\star$ in U such that $W = f(U)$ is open in $\mathbb{R}^n$ and $f|_V$ is a diffeomorphism onto W.*

Corollary 2.1.3. *Let $x^\star \in \Gamma^\star$. There exists a neighbourhood $U \subset \mathbb{R}^3$ containing $x^\star$ such that the mapping $T : U \subset \mathbb{R}^3 \to T(U) \subset \mathbb{R}^3$, defined by*

$$\begin{bmatrix} \eta_1 \\ \xi_1 \\ \xi_2 \end{bmatrix} = T(x) = \begin{bmatrix} \pi(x) \\ \alpha(x) \\ L_f\alpha(x) \end{bmatrix}, \tag{2.6}$$

is a diffeomorphism onto its image.

Proof. Let $x^\star \in \Gamma$. By direct calculations the Jacobian $dT_{x^\star} := \frac{\partial T}{\partial x}\big|_{x=x^\star}$ is given by

$$dT_{x^\star} = \begin{bmatrix} \frac{\partial \eta_1}{\partial x_1} & \frac{\partial \eta_1}{\partial x_2} & \frac{\partial \eta_1}{\partial x_3} \\ \frac{\partial \xi_1}{\partial x_1} & \frac{\partial \xi_1}{\partial x_2} & \frac{\partial \xi_1}{\partial x_3} \\ \frac{\partial \xi_2}{\partial x_1} & \frac{\partial \xi_2}{\partial x_2} & \frac{\partial \xi_2}{\partial x_3} \end{bmatrix} = \begin{bmatrix} -x_2 & x_1 & 0 \\ 2v\cos x_3 & 2v\sin x_3 & 2v(x_2\cos x_3 - x_1 \sin x_3) \\ 2x_1 & 2x_2 & 0 \end{bmatrix}. \tag{2.7}$$

The determinant of (2.7) is given by,

$$\det\left(dT_{x^\star}\right) = -4v(x_2\cos x_3 - x_1 \sin x_3).$$

As we have already shown, on $\Gamma^\star$ this determinant equals zero if and only if $v = 0$. Since we assume that $v \neq 0$, $dT_{x^\star}$ is nonsingular. By Theorem (2.1.2), T is diffeomorphism onto its image. □

Using the coordinate transformation T from Corollary 2.1.3, in the neighborhood of any point $x^\star \in \Gamma^\star$, the system (2.4) in (η, ξ)-coordinates reads

$$\begin{aligned} \dot\eta_1 &= f^0(\eta, \xi) \\ \dot\xi_1 &= \xi_2 \\ \dot\xi_2 &= L_f^2\alpha + L_gL_f\alpha u\big|_{x=T^{-1}(\eta,\xi)}. \end{aligned} \tag{2.8}$$

When $\xi = 0$, i.e., when $\alpha(x) = \dot\alpha(x) = 0$, the system is restricted to evolve on the path following manifold. Thus stabilizing the ξ-states is equivalent to getting the unicycle on the desired path with heading velocity tangent to the path. This motivates us to call the ξ-subsystem the transversal subsystem and the ξ-states the transversal states. When the robot is on the path following manifold, i.e., $\xi = 0$ then η_1 determines the position of the robot on the path. The dynamics of η_1 restricted to the path following manifold are given by

$$\dot\eta_1|_{\Gamma^\star} = f^0(\eta, 0) = -\left.\frac{v(x_2\cos x_3 - x_1\sin x_3)}{x_1^2 + x_2^2}\right|_{x=T^{-1}(\eta,0)}.$$

Note that on $\Gamma^\star$

(i) $x_1^2 + x_2^2 = 1$

(ii) $x_1 \cos(x_3) = -x_2 \sin(x_3)$.

Solving (i) and (ii) for x_1 and x_2 yields two valid solutions

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -\sin(x_3) \\ \cos(x_3) \end{bmatrix}$$

or

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \sin(x_3) \\ -\cos(x_3) \end{bmatrix}.$$

Substituting these solutions into the expression for $\dot{\eta}_1$ yields

$$\dot{\eta}_1 = \pm v. \tag{2.9}$$

In either case, the dynamics restricted to the path are unstable. In this application, this is desirable because it means that the unicycle will traverse the entire circle. On the other hand, since v is constant, we cannot affect the motion of the vehicle on the circle. The minus and plus signs correspond to, respectively, clockwise and counterclockwise path traversal.

Next consider the regular feedback transformation

$$u = \frac{1}{L_g L_f \alpha}(-L_f \alpha^2 + v^{\pitchfork}), \tag{2.10}$$

where $v^{\pitchfork}$ is an auxiliary control inputs. The controller is well defined in a neighborhood of $x^\star \in \Gamma^\star$ because $L_g L_f \alpha(x) \neq 0$ there. In a neighbourhood of $x^\star$ the closed-loop system becomes

$$\begin{aligned} \dot{\eta}_1 &= f^0(\eta, \xi) \\ \dot{\xi}_1 &= \xi_2 \\ \dot{\xi}_2 &= v^{\pitchfork} \end{aligned} \tag{2.11}$$

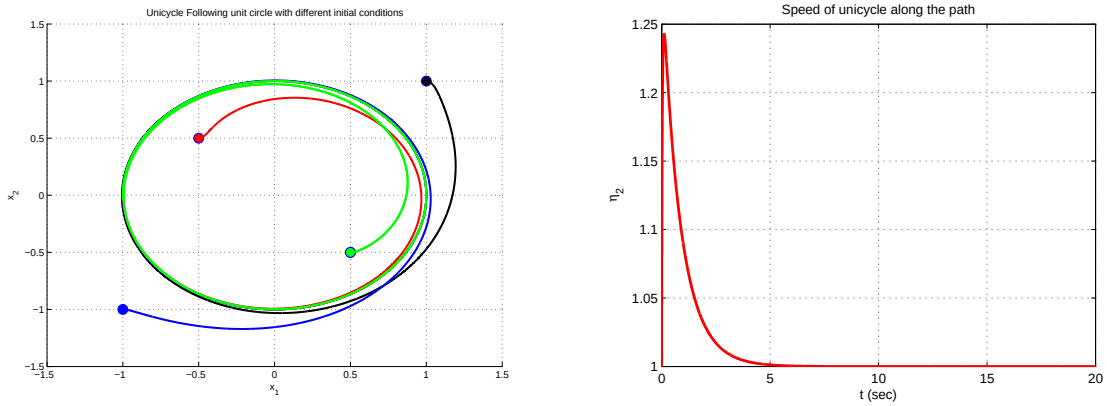
We refer to the control input $v^{\pitchfork}$ as the transversal input because it can be used to control the transversal subsystem. The transversal subsystem is LTI and controllable so there are many possible design techniques that can be used to stabilize $\xi = 0$. This simplest choice for the transversal controller is

$$v^{\pitchfork}(\xi) = k_1 \xi_1 + k_2 \xi_2, \tag{2.12}$$

with $k_i < 0$, $i \in \{1, 2\}$. This controller exponentially stabilizes the transversal states. Physically, since $\xi = 0$ is an equilibrium of the closed-loop transversal subsystem, if the robot is initialized on the path with the initial velocity tangent to the path, then it will remain on the path for all future time.

2.1.1 Simulation Results

In this section we present simulation results of the unicycle, (2.4), following both closed and non closed paths. We set $v = 1$ throughout. In Figure 2.4(a), the unicycle is following a circular path starting from different initial conditions. Each initial condition is represented by a solid dot. As seen above, the controller (2.10), (2.12), has the limitation that the speed of the robot is fixed. The speed of the unicycle while following circular path is represented in Figure 2.4(b). As expected, the dynamics of the unicycle on the path are given by (2.9) and this is illustrated in Figure 2.4(b).



(a) Unicycle starting from various initial conditions.

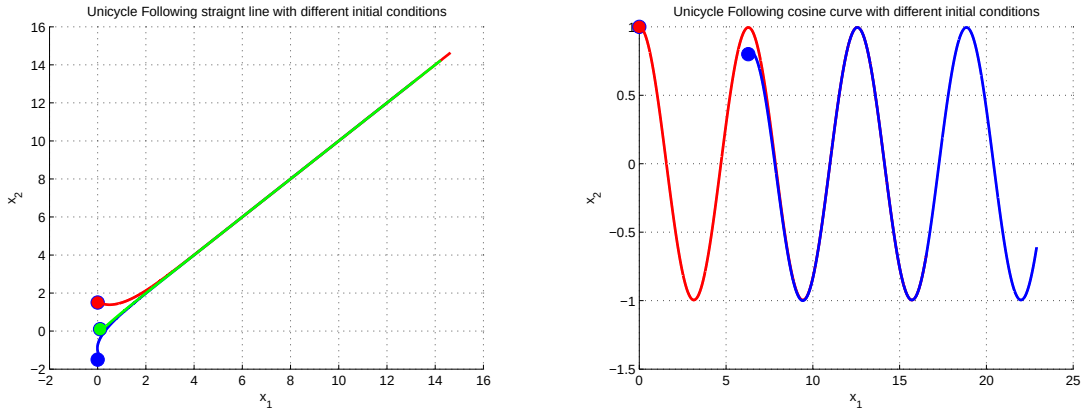
(b) Speed of unicycle along the curve

Figure 2.4: Unicycle (2.4) following a circular path.

Although we discussed the circle example in Section 2.1, the results in [43], [41] can also be applied to any path that satisfies Assumptions 1 and 2. This includes non closed curves. Here we present simulation results of the unicycle following some non closed curves. Figure 2.5(a) shows simulation results for the case when the unicycle is following a straight line. The experiment is repeated with different initial conditions as can be seen in the aforementioned figure. The next example of a non closed curve is that of a sinusoidal path. Figure 2.5(b) shows the reference path followed by the robot with various initial conditions.

2.2 Dynamic Transverse Feedback Linearization of Unicycle Following a Circular Path

In this thesis, one of our objective is to overcome the limitation of constant speed of the robot along the path. That means satisfying **PF3** defined in Section 1.3. As discussed in



(a) Unicycle following a straight line.

(b) Unicycle following a cosine curve

Figure 2.5: Unicycle (2.4) following non closed paths.

the last section, the solution proposed in [43] does not satisfy **PF3**. In particular, since v is fixed the motion on the path is fixed. A natural way to overcome this limitation is not to fix the translational velocity of the robot and use both inputs to design the controller. The main issue with this approach is that it affects the relative degree of the system and the analysis from the previous section is no longer applicable.

Consider the model of the unicycle robot (2.1) with both the inputs and the circular path (2.3). Following the procedure of the previous section, we take the derivative of the curve until the control input appears,

$$\begin{aligned}\dot{\alpha} &= 2x_1(\dot{x}_1) + 2x_2(\dot{x}_2) \\ &= 2x_1(v\cos x_3) + 2x_2(v\sin x_3).\end{aligned}$$

The control input appears in the first derivative and the equation

$$2x_1(v\cos x_3) + 2x_2(v\sin x_3) = 0$$

can easily be solved by setting $v = 0$. Hence the zero dynamics algorithm [29] terminates and the largest controlled invariant subset of Γ is given by Γ itself. Clearly the control $v = 0$ is not very useful for a mobile robot.

Observe that only one control input, translational speed, of the unicycle appears in the first derivative. The steering input has not yet appeared. Intuitively, without the steering input we cannot control the steering angle of the robot. In order to control both the speed and direction of the robot we would like to have both inputs appear in the derivatives of α. In other words, both inputs should appear at the same derivative of the path $\alpha(x)$. One way to force both inputs appear at the same derivative is to "delay" the appearance of one

of the input v. We do this by controlling the derivative of v rather than v itself. It is clear that the second derivative of $\alpha(x)$ contains the term $\dot{x}_3 = \omega$. Let $v = \mathrm{v} + \overline{u}$, where $\mathrm{v} > 0$ is constant and $\overline{u}$ will be treated as as a new state. Instead of controlling v we will instead control the derivative of $\overline{u}$. This procedure is referred to as dynamic extension [29]. As it is easy to see, in this case, for both the inputs to appear at the same time we need to add only one state. Let $x_4 := \overline{u}$ and choose $u_1 := \dot{x}_4$ and $u_2 := \omega$. With these definitions, the system (2.1) takes the form

$$\begin{aligned} \dot{x} &= f(x) + g_1(x)u_1 + g_2(x)u_2 \\ &= \begin{bmatrix} (\mathrm{v} + x_4)\cos x_3 \\ (\mathrm{v} + x_4)\sin x_3 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} u_2. \end{aligned} \tag{2.13}$$

Now our objective is to design the control law $u = (u_1, u_2)$ to solve the the path following problem defined in Section 1.3. Similar to Section 2.1, the *lift* of γ to $\mathbb{R}^4$

$$\Gamma := (s \circ h)^{-1}(0) = \left\{ x \in \mathbb{R}^4 : s(h(x)) = \alpha(x) = 0 \right\}.$$

As discussed above, following the procedure from Section 2.1, both inputs appear in the second derivative of α. In this case the path following manifold is given by

$$\Gamma^\star = \left\{ x \in \mathbb{R}^4 : \alpha(x) = \dot{\alpha}(x) = 0 \right\}.$$

We define a "virtual" output function [25].

$$\hat{y} = \begin{bmatrix} \pi(x) \\ \alpha(x) \end{bmatrix} \tag{2.14}$$

where $\pi(x)$ is defined in (2.5).

We now show that as long as the unicycle does not have zero translational speed, then this output yields a well defined relative degree on the path.

Lemma 2.2.1. *The dynamic extension of the unicycle robot* (2.13) *with output* (2.14) *yields a well defined vector relative degree of* $\{2, 2\}$ *at each point on* $\Gamma^\star$ *where* $x_4 \neq -\mathrm{v}$.

Proof. Let $x^\star \in \Gamma$ be arbitrary. By definition of Γ the output $h(x^\star)$ is on the path γ. Let $\lambda^\star \in \mathbb{S}$ be such that $h(x^\star) = \sigma(\lambda^\star)$. By the definition of vector relative degree we must show that

$$L_{g_1}\pi(x) = L_{g_2}\pi(x) = L_{g_1}\alpha(x) = L_{g_2}\alpha(x) = 0$$

in a neighbourhood of $x^\star$ and that the decoupling matrix

$$D(x^\star) = \begin{bmatrix} L_{g_1}L_f\pi(x^\star) & L_{g_2}L_f\pi(x^\star) \\ L_{g_1}L_f\alpha(x^\star) & L_{g_2}L_f\alpha(x^\star) \end{bmatrix} \tag{2.15}$$

is nonsingular. Since

$$\frac{\partial \pi(x)}{\partial x_i} = \frac{\partial \alpha(x)}{\partial x_i} \equiv 0$$

for $i \in \{3, 4\}$, it is easy to check that $L_{g_1}\pi(x) = L_{g_2}\pi(x) = L_{g_1}\alpha(x) = L_{g_2}\alpha(x) = 0$.

To show that the decoupling matrix is full rank, it suffices to show that the determinant of $D(x^\star)$ is not zero. Direct calculations yield

$$\begin{aligned} L_{g_1}L_f\alpha &= 2(\mathrm{v} + x_4)(x_2\cos x_3 - x_1 \sin x_3) \\ L_{g_2}L_f\alpha &= 2x_1\cos x_3 + 2x_2 \sin x_3 \\ L_{g_1}L_f\pi &= (\mathrm{v} + x_4)(x_1\cos x_3 + x_2 \sin x_3) \\ L_{g_2}L_f^2\pi &= -(x_2\cos x_3 - x_1 \sin x_3) \end{aligned} \tag{2.16}$$

Hence

$$\det\left(D(x)\right) = 2(\mathrm{v} + x_4). \tag{2.17}$$

The only way for this determinant to vanish is if $\mathrm{v} = -x_4$. Thus $D(x)$ is nonsingular at each $x^\star \in \Gamma^\star$ where $x_4 \neq 0$. Since $\Gamma^\star \subset \Gamma$, the lemma is proved. □

An immediate consequence of Lemma 2.2.1 is that it allows us to define a local diffeomorphism using the function $\pi(x)$ and $\alpha(x)$ and their iterated Lie derivatives along the vector field $f(x)$.

Corollary 2.2.2. *Let $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^4 : x_4 + \mathrm{v} = 0\}$. There exists a neighbourhood $U \subset \mathbb{R}^4$ containing $x^\star$ such that the mapping $T : U \subset \mathbb{R}^4 \to T(U) \subset \mathbb{R}^4$, defined by*

$$\begin{bmatrix} \eta_1 \\ \eta_2 \\ \xi_1 \\ \xi_2 \end{bmatrix} = T(x) = \begin{bmatrix} \pi(x) \\ L_f\pi(x) \\ \alpha(x) \\ L_f\alpha(x) \end{bmatrix} \tag{2.18}$$

is a diffeomorphism.

Proof. Let $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^4 : x_4 + \mathrm{v} = 0\}$. By Lemma 2.2.1 system (2.13) with output (2.14) yields a well defined vector relative degree of $\{2, 2\}$ at $x^\star$.By direct calculations the jacobian, $\frac{\partial T}{\partial x} := dT_{x^\star}$ is given by,

$$dT_{x^\star} = \begin{bmatrix} \frac{\partial \eta_1}{\partial x_1} & \frac{\partial \eta_1}{\partial x_2} & \frac{\partial \eta_1}{\partial x_3} & \frac{\partial \eta_1}{\partial x_4} \\ \frac{\partial \eta_2}{\partial x_1} & \frac{\partial \eta_2}{\partial x_2} & \frac{\partial \eta_2}{\partial x_3} & \frac{\partial \eta_2}{\partial x_4} \\ \frac{\partial \xi_1}{\partial x_1} & \frac{\partial \xi_2}{\partial x_2} & \frac{\partial \xi_1}{\partial x_3} & \frac{\partial \xi_1}{\partial x_4} \\ \frac{\partial \xi_2}{\partial x_1} & \frac{\partial \xi_2}{\partial x_2} & \frac{\partial \xi_2}{\partial x_3} & \frac{\partial \xi_2}{\partial x_4} \end{bmatrix} = \begin{bmatrix} \frac{\partial \eta_1}{\partial x_1} & \frac{\partial \eta_1}{\partial x_2} & 0 & 0 \\ \frac{\partial \eta_2}{\partial x_1} & \frac{\partial \eta_2}{\partial x_2} & \frac{\partial \eta_2}{\partial x_3} & \frac{\partial \eta_2}{\partial x_4} \\ 2x_1 & 2x_2 & 0 & 0 \\ \frac{\partial \xi_2}{\partial x_1} & \frac{\partial \xi_2}{\partial x_2} & \frac{\partial \xi_2}{\partial x_3} & \frac{\partial \xi_2}{\partial x_4} \end{bmatrix}. \tag{2.19}$$

where

$$\frac{\partial \eta_1}{\partial x_1} = -\frac{x_2}{x_1^2 + x_2^2},$$

$$\frac{\partial \eta_1}{\partial x_2} = \frac{x_1}{x_1^2 + x_2^2},$$

$$\frac{\partial \eta_2}{\partial x_1} = \frac{(v + x_4)(-x_1^2 \sin x_3 + 2x_1 x_2 \cos x_3 + x_2^2 \sin x_3)}{(x_1^2 + x_2^2)^2},$$

$$\frac{\partial \eta_2}{\partial x_2} = -\frac{(v + x_4)(x_1^2 \cos x_3 + 2x_1 x_2 \sin x_3 - x_2^2 \cos x_3)}{(x_1^2 + x_2^2)^2},$$

$$\frac{\partial \eta_2}{\partial x_3} = \frac{(v + x_4)(x_1 \cos x_3 + x_2 \sin x_3)}{x_1^2 + x_2^2},$$

$$\frac{\partial \eta_2}{\partial x_4} = -\frac{x_2 \cos x_3 - x_1 \sin x_3}{x_1^2 + x_2^2},$$

$$\frac{\partial \xi_2}{\partial x_1} = 2(v + x_4) \cos x_3,$$

$$\frac{\partial \xi_2}{\partial x_2} = 2(v + x_4) \sin x_3,$$

$$\frac{\partial \xi_2}{\partial x_3} = 2(v + x_4)(x_2 \cos x_3 - x_1 \sin x_3),$$

$$\frac{\partial \xi_2}{\partial x_4} = 2(x_1 \cos x_3 + x_2 \sin x_3).$$

The determinant of (2.19) is given by

$$\det (D(x) = 4(v + x_4). \tag{2.20}$$

The determinant goes to zero only if $x_4 = -\mathrm{v}$ for all $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^4 : x_4 + \mathrm{v} = 0\}$. By Theorem 2.1.2 T is diffeomorphism onto its image. □

Using the coordinate transformation T from Corollary 2.2.2, in a neighbourhood of any point $x^\star \in \Gamma$ the system (2.13) in (η, ξ) coordinates reads

$$\begin{aligned}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= L_f^2 \pi + L_{g_1} L_f \pi u_1 + L_{g_2} L_f \pi u_2 \big|_{x = T^{-1}(\eta,\xi)} \\
\dot{\xi}_1 &= \xi_2 \\
\dot{\xi}_2 &= L_f^2 \alpha + L_{g_1} L_f \alpha u_1 + L_{g_2} L_f \alpha u_2 \big|_{x = T^{-1}(\eta,\xi)} .
\end{aligned} \tag{2.21}$$

Similar to Section 2.1 ξ-subsystem is called transversal subsystem and the states ξ the transversal states. We call the η-subsystem the tangential subsystem and states η the tangential states.

Consider the regular feedback transformation

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} := D^{-1}(x) \left(\begin{bmatrix} -L_f^2 \pi \\ -L_f^2 \alpha \end{bmatrix} + \begin{bmatrix} v^{\parallel} \\ v^{\pitchfork} \end{bmatrix} \right), \tag{2.22}$$

where $(v^{\parallel}, v^{\pitchfork}) \in \mathbb{R}^2$ are auxiliary control inputs. By Lemma 2.2.1 this controller is well defined in a neighbourhood of every $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^4 : x_4 + \mathrm{v} = 0\}$. Thus in a neighbourhood of $x^\star$, the closed-loop system becomes

$$\begin{aligned}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= v^{\parallel} \\
\dot{\xi}_1 &= \xi_2 \\
\dot{\xi}_2 &= v^{\pitchfork}.
\end{aligned} \tag{2.23}$$

We have effectively feedback linearized the extended system using an output that is physically meaningful for path following. The specifications **PF1**, **PF2**, and **PF3** can be easily expressed in terms of the state variables (η, ξ). Meeting these specifications is also simplified because both the transversal and tangential subsystems are LTI and controllable. The virtual output (2.14) is an example of a differentially flat output. Note that for general system there is no guarantee that this type of dynamic extension will yield a fully feedback linearizable system.

Transversal and Tangential Controller Design

To exponentially stabilize $\xi = 0$ the controller (2.12) is used as presented in Section 2.1. This controller makes the closed-loop system meet specifications **PF1** and **PF2**. We have

not made a specific choice for **PF3**, but for the purposes of illustration, suppose we want to track a velocity profile along the curve. A simple proportional controller is used in order to achieve **PF3**

$$v^{\parallel}(\eta) = k_3(\eta_2 - \eta_2^{ref}), \tag{2.24}$$

$k_3 < 0$. The parameter η_2^{ref} is a desired reference velocity profile. The overall control scheme is presented in Figure 2.6.

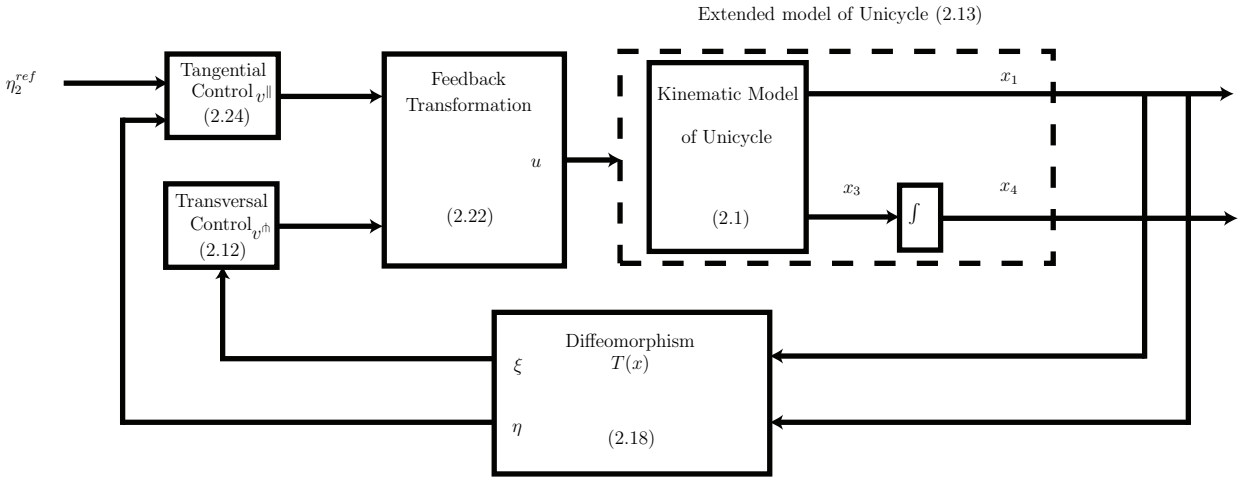


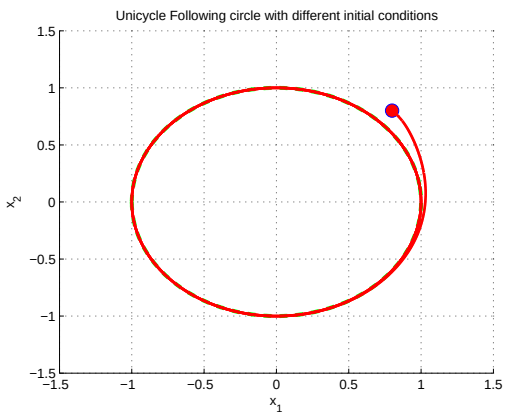
Figure 2.6: Dynamic Transverse Feedback Linearization of Unicycle robot (2.1) with equation references.

Simulation

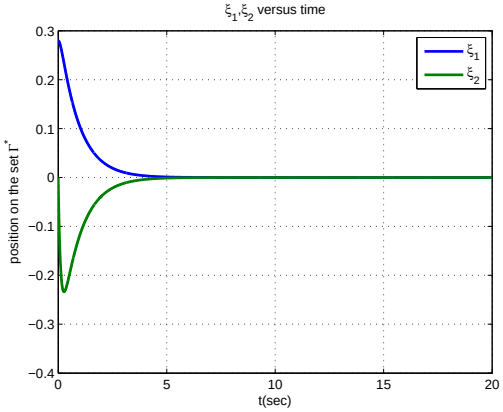
Simulation results of the unicycle (2.13) with the feedback law (2.22), (2.12), (2.24) is presented in Figure 2.7. For these simulations v = 1. In Figure 2.7(a) the unicycle follows the desired circular path. The unicycle is initialized at $x(0) = (0.8, 0.8, \frac{\pi}{4}, 0) \in \mathbb{R}^4$. The robot converges to the path and then traverses the path with the desired speed and in the desired direction. Figure 2.7(b) shows the ξ-states converging to zero. While the unicycle traverses the path we make the unicycle follow the velocity profile given by

$$\eta_2^{ref} = \begin{cases} -0.5 & 0 \leq t < 10s \\ -1.5 & t \geq 10s. \end{cases}$$

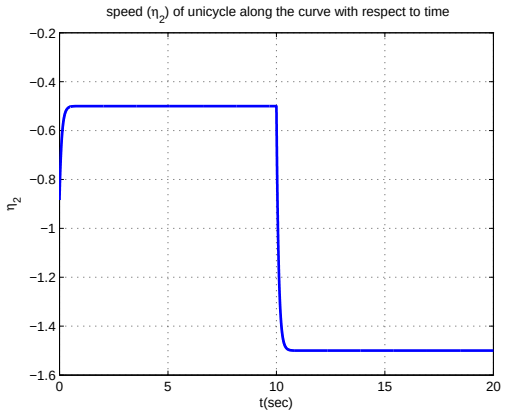
Figure 2.7(c) shows the unicycle following the desired velocity profile. The path parameter η_1 is shown in Figure 2.7(d). As the curve is closed the path parameter is bounded.



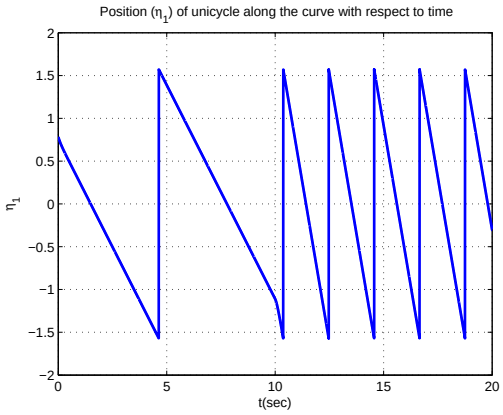
(a) Unicycle following a circular path.



(b) ξ-states converging to 0.



(c) Unicycle following a desired velocity profile.



(d) Path parameter.

Figure 2.7: Unicycle robot (2.13) following a circular path.

It is important to note that in this example we have assumed that both parametric and implicit forms of the curves are available. In Chapter 5, we provide a procedure to implicitize a given parametric curve. In this example it is also assumed that the given curve is closed and parameterized by its arc length. In Chapter 3, we give a procedure to deal with non closed and non unit-speed parameterized curves. The approach for solving the path following problem can be summarized in the following steps,

Step 1 Fix the translational velocity of the mobile robot and use the steering input to find the path following manifold $\Gamma^\star$ using the lift of the path $\alpha(x)$.

Step 2 Perform dynamic extension of the given system such that the steering and velocity inputs appear in the same derivative of α.

Step 3 If possible, feedback linearize the extended system. It is important to once again point out that in general the η-subsystem will be nonlinear. In the example of the unicycle, the system is differentially flat[1] and the virtual output we used is a flat output. A block diagram of the dynamically feedback linearized system is shown in Figure 2.8.

Step 4 Design a transversal feedback controller $v^{\pitchfork}(\xi)$ stabilizing the origin of the transversal subsystem. Stabilizing the transversal subsystem is equivalent to stabilizing the path following manifold $\Gamma^{\star}$. This corresponds to forcing the output of the system to the path.

Step 5 Design a tangential feedback controller $v^{\parallel}(\eta, \xi)$ such that, when $\xi = 0$, the tangential subsystem meets the goal **PF3**.

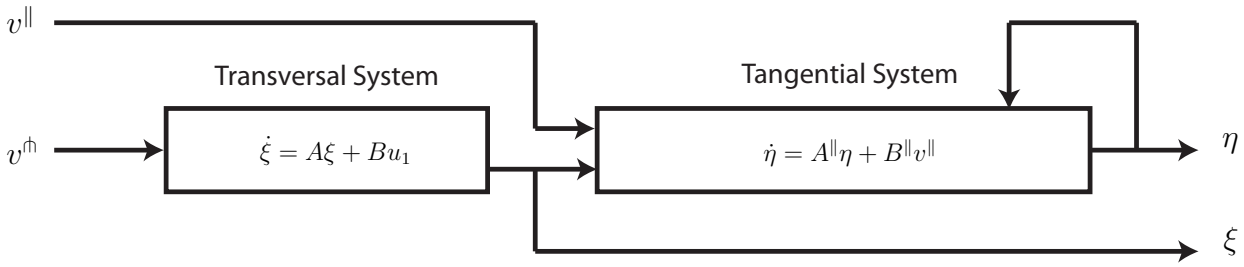


Figure 2.8: Block diagram of Feedback Linearization

2.3 Comparison to Other Control Techniques

In this section we compare the path following technique described earlier in this chapter to a few other existing trajectory tracking and path following techniques.

2.3.1 Trajectory Tracking Controller

One of the common techniques used in the literate to execute a desired motion is to follow a trajectory defined in the output space of the mobile robot. We present an example based on the work presented by the authors in [7]. Consider the kinematic model of the unicycle

[1]Flat systems are discussed further in subsequent chapters. For further details readers are referred to [21], [22].

type robot (2.1) represented using slightly different notation just for ease of explaining the tracking procedure

$$\begin{aligned} \dot{x}(t) &= v(t)\cos\theta(t), \\ \dot{y}(t) &= v(t)\sin\theta(t), \\ \dot{\theta} &= \omega(t). \end{aligned} \tag{2.25}$$

Here x and y denote the robot's position and θ represents the orientation of the unicycle. The translational and rotational velocities of the unicycle are represented by v and ω respectively. The control objective in this tracking example is to track a time-varying reference trajectory specified by $[x_r(t) \;\; y_r(t) \;\; \theta_r(t)]^\top$. The reference position $(x_r(t), y_r(t))$ satisfies the dynamics,

$$\begin{aligned} \dot{x}_r(t) &= v_r(t)\cos\theta_r(t), \\ \dot{y}_r(t) &= v_r(t)\sin\theta_r(t). \end{aligned} \tag{2.26}$$

The reference orientation $\theta_r(t)$, translational velocity $v_r(t)$,and rotational velocity $\omega_r(t)$ are defined in terms of the velocities $\dot{x}_r(t)$, $\dot{y}_r(t)$ and accelerations $\ddot{x}_r(t)$, $\ddot{y}_r(t)$ as follows [7]

$$\begin{aligned} \theta_r(t) &= \tan^{-1}\left(\frac{\dot{y}_r(t)}{\dot{x}_r(t)}\right), \\ v_r(t) &= \sqrt{\dot{x}_r^2(t) + \dot{y}_r^2(t)}, \\ \omega_r(t) &= \frac{\dot{x}_r(t)\ddot{y}_r(t) - \ddot{x}_r(t)\dot{y}_r(t)}{\dot{x}_r^2(t) + \dot{y}_r^2(t)}. \end{aligned} \tag{2.27}$$

A common way to solve a trajectory tracking problem is to first define error coordinates. The rotated version of the tracking error as defined in [7] is given by

$$\begin{bmatrix} x_e(t) \\ y_e(t) \\ \theta_e(t) \end{bmatrix} = \begin{bmatrix} \cos\theta(t) & \sin\theta(t) & 0 \\ -\sin\theta(t) & \cos\theta(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r(t) - x(t) \\ y_r(t) - y(t) \\ \theta_r(t) - \theta(t) \end{bmatrix}. \tag{2.28}$$

The tracking error dynamics can be derived using (2.25), (2.26) and (2.27),

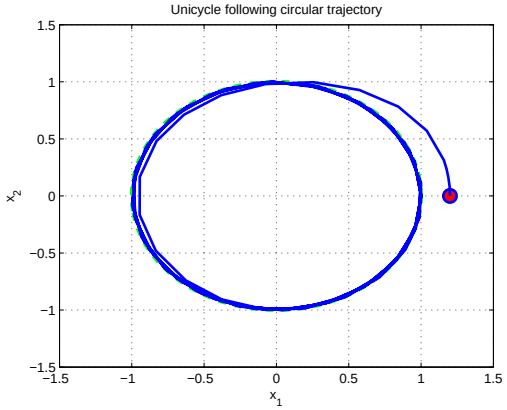
$$\begin{aligned} \dot{x}_e(t) &= \omega(t)y_e(t) + v_r(t)\cos\theta_e(t) - v(t), \\ \dot{y}_e(t) &= -\omega(t)x_e(t) + v_r(t)\sin\theta_e(t), \\ \dot{\theta}_e(t) &= \omega_r(t) - \omega(t). \end{aligned} \tag{2.29}$$

The trajectory tracking controller proposed in [30] is given by

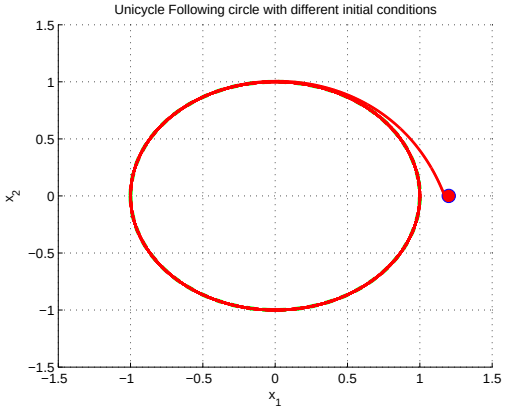
$$\begin{aligned} \dot{v}(t) &= v_r(t) + c_2 x_e(t) - c_3\omega_r(t)y_e(t), \\ \dot{\omega}(t) &= \omega_r(t) + c_1\sin\theta_e(t). \end{aligned} \tag{2.30}$$

Interested readers are referred to [30] for further details and the proof of exponential stability of the origin of the error system (2.29).

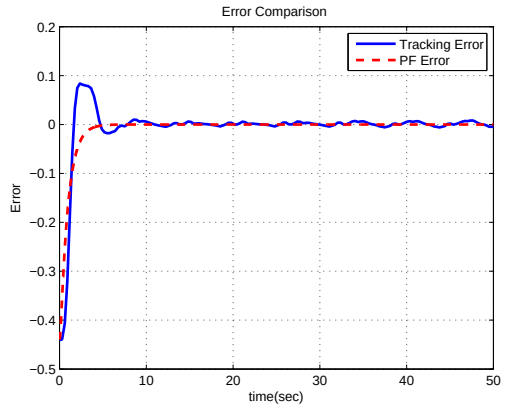
We now compare the results of the above mentioned trajectory tracking controller (2.30) with our path following controller (2.22), (2.12) and (2.24). See Figure 2.9.



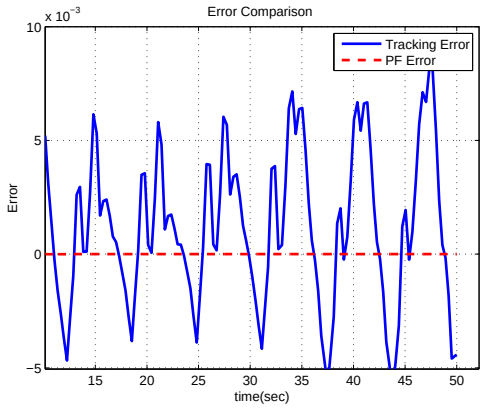
(a) Unicycle tracking a circular trajectory.



(b) Unicycle Following a circular path.



(c) Comparison between path following error and tracking error.



(d) Zoomed in comparison between path following error and tracking error.

Figure 2.9: Comparison between path following and trajectory tracking.

Figure 2.9(a) shows the unicycle tracking a circular trajectory $(x_r(t), y_r(t)) = (\cos(t), \sin(t))$. The unicycle is initialized at $x(0) = 0$, $y(0) = 1.2$. As shown in Figure 2.9(a), the unicycle converges to the trajectory and tries to track the trajectory. Note that while tracking the trajectory there is some error. This is because the unicycle tries to follow a virtual target that moves along the trajectory. The virtual target $(x_r(t), y_r(t))$ is parameterized by time and the controller forces the unicycle to reach to the virtual target. If the target moves too fast the unicycle may leave the trajectory.

In Figure 2.9(b) we apply the path follwing controller (2.22), (2.12) and (2.24) to the unicycle and the circular path $\gamma = \{(x_1, x_2) : x_1^2 - x_2^2 - 1 = 0\}$. The path is not parameterized by time. This means there is no requirement to achieve a desired position along that path at a desired time. The primary objective is to follow the desired path. Moreover, our proposed path following controller guarantees the invariance of path. That means once the robot is on the path it will remain on the path for all future time. In Figure 2.9(c) a comparison between path following error and tracking error is shown. These errors are point to set distance. The tracking error is shown with the solid line, while the path following error is shown with the dotted line. Clearly path following error is less than the trajectory tracking error. It is interesting to observe in Figure 2.9(d), which is the zoomed in view of the graph shown at the left, that the path following controller allows the robot to traverse through the path with zero steady state error, this is not the case for the trajectory tracking case. Note that the tracking error could likely be improved by adding integral action to the tracking control law.

2.3.2 Path Following Using Sliding Mode Control Theory

In this example we compare our path following controller to another path following controller proposed by the authors [17] using sliding mode control theory. Consider again the kinematic model of unicycle (2.1). The state x_3 represents the orientation θ of the unicycle. In [17] a simple sliding mode controller is designed by identifying a sliding manifold. Since $\dot{\theta} = \omega$ a simple sliding mode controller without first or higher order dynamics can be designed, as a result the following sliding manifold is designed in order to make the orientation angle of the mobile robot converge to the desired orientation angle

$$\mathcal{S} = \{(x_1, x_2, \theta) \in \mathbb{R}^3 : s_\theta = \theta - \theta_{ref} = 0\}, \tag{2.31}$$

where, $\theta_{ref} = \theta_{ref}(x_1, x_2)$.

If $s_\theta \dot{s}_\theta < 0$, then the set (2.31) becomes attractive and the desired dynamics are achieved.

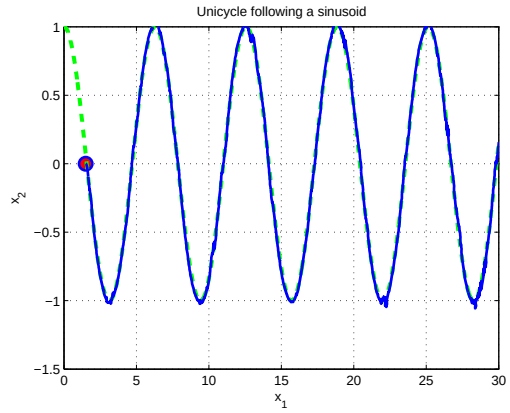
$$\dot{s}_\theta = \dot{\theta} - \dot{\theta}_{ref}.$$

Since the steering control input ω appears in $\dot{\theta}$, we can assign the s_θ dynamics however we want and in particular we can stabilize the sliding surface $s_\theta = 0$. As is customary in sliding mode, we stabilize $s_\theta = 0$ in finite-time using the control law

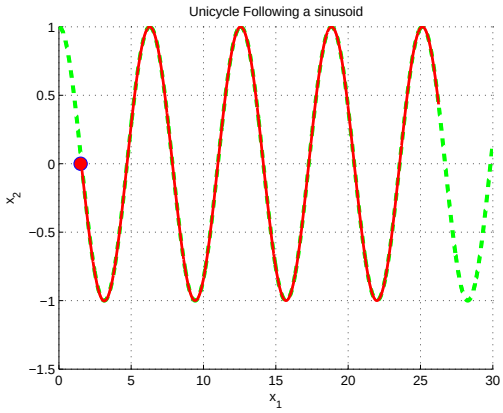
$$\omega = -Msgn(s_\theta).$$

With this controller we can stabilize the sliding surface $\mathcal{S}$

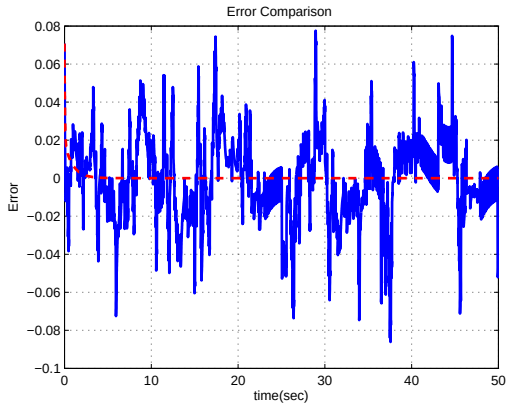
$$\dot{s}_\theta = -Msgn(s_\theta) - \dot{\theta}_{ref} \tag{2.32}$$



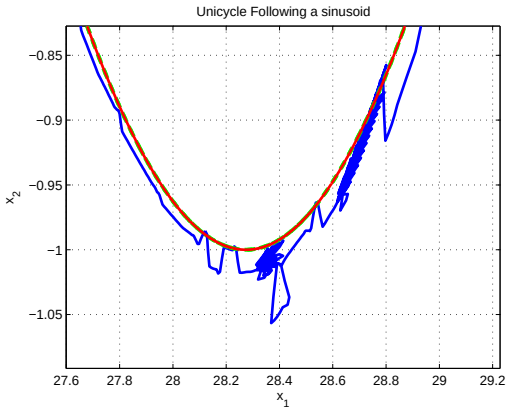
(a) Path following by sliding mode controller.



(b) Path following by transverse Feedback linearization.



(c) Error comparison.



(d) Zoomed in comparison of path following errors.

Figure 2.10: Comparison of path following controller designed using sliding mode control theory and the one proposed in this thesis.

and if M dominates $\dot{\theta}_{ref}$, i.e., $M > |\dot{\theta}_{ref}|$, then $s_\theta \dot{s}_\theta < 0$ is satisfied and s_θ reaches zero in a finite amount of time.

In Figure 2.10 we present the results of path following of a unicycle robot following a sinusoidal curve. The sliding mode controller is implemented for comparison purposes based on the work presented by the authors in [17]. In Figure 2.10(a) unicycle is following a sinusoidal curve. The initial position of the unicycle is represented as a solid dot in the figure. The desired curve is represented as a dotted line while the traversed path is represented by a solid line in Figure 2.10(a) and Figure 2.10(b) for sliding mode controller and the controller proposed in Section 2.1. In both cases unicycle is initialized at the same point. In Figure 2.10(c) path following error is compared. The solid line represent path

following error of sliding mode control, while the dotted line represent path following error of our proposed controller. Clearly, the proposed controller has better error performance, this is because the proposed controller makes the desired path invariant and results in perfect path following. However, using sliding mode controller perfect path following is not possible. Figure 2.10(d) gives a zoomed in view of a part of the curve followed using sliding mode controller and the controller designed earlier in this chapter. One of the reasons for the comparatively large path following error is that the sliding mode controller is discontinues. This results in chattering that may result in comparatively larger path following errors.

Chapter 3

Path Following Controllers: Car-like Robot

In Chapter 2, a path following controller was reviewed based on the work done by authors [42] for a single input kinematic model of unicycle. In Section 2.2, the kinematic model of the two input unicycle was considered and a dynamic controller was designed for a circular path to make the unicycle follow the unit circle with the desired speed. In the last chapter, the procedure given was not for general curves. Moreover, the discussion on dynamic extension of the unicycle was intuitive, because the model was relatively simple. In this chapter, we provide a procedure to deal with a large class of general curves and we work with a more complicated model of a mobile robot, the car-like robot. Specifically, in this chapter a path following controller for the two input kinematic model of a car-like robot is presented. A dynamic feedback control law is designed to make the position of the car to follow a large class of curves with the desired speed in the desired direction. The controller is designed by characterizing the path following manifold when one of the inputs is fixed. Once the path following manifold is found we apply dynamic extension to increase its dimension. We refer to this process as tangential dynamic extension. We then find a physically meaningful differentially flat output for the extended system which allows us to easily solve the path following problem.

3.1 Model of a Car-like Robot

Consider the kinematic model of a car-like robot, Figure 3.1,

$$\dot{x} = \begin{bmatrix} \cos x_3 & 0 \\ \sin x_3 & 0 \\ \frac{1}{\ell}\tan x_4 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{3.1}$$

where $x \in \mathbb{R}^4$ is the state, the input $v \in \mathbb{R}$ is the translational speed and $\omega \in \mathbb{R}$ is the angular velocity of the steering angle. We take the car's position in the plane as the output of (3.1)

$$y = h(x) = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\top. \tag{3.2}$$

Suppose we are given a regular parameterized path (1.5) satisfying Assumptions 1 and 2.

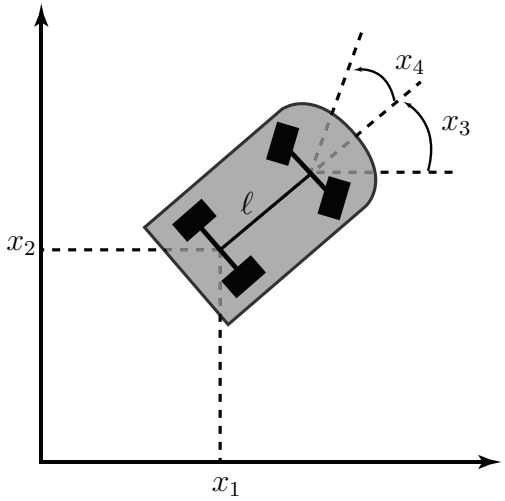


Figure 3.1: The kinematic model of the car-like robot.

As discussed in Chapter 1, Assumption 2 means that the entire path can be represented as the zero level set of the function s in the output space of system (3.1). As Assumption 1 holds this is always possible, at least locally. Since the output (3.2) satisfies $\operatorname{rank}(dh_x) = 2$ for all $x \in \mathbb{R}^4$, the map $h : \mathbb{R}^4 \to \mathbb{R}^2$ is transversal [23] to γ and therefore, if Assumption 1 holds, the *lift* of γ to $\mathbb{R}^4$

$$\Gamma := (s \circ h)^{-1}(0) = \left\{ x \in \mathbb{R}^4 : s(h(x)) = 0 \right\}$$

is a three dimensional submanifold. Define $\alpha(x) := s \circ h(x)$.

Given a curve $\sigma(\mathbb{D})$ satisfying Assumptions 1 and 2, we seek a smooth control law of the form

$$\begin{aligned} \dot{\zeta} &= a(x,\zeta) + b(x,\zeta)u \\ \begin{bmatrix} v \\ \omega \end{bmatrix} &= c(x,\zeta) + d(x,\zeta)u. \end{aligned} \tag{3.3}$$

with $\zeta \in \mathbb{R}^q$ and $u = (u_1, u_2) \in \mathbb{R}^2$ and an open subset of initial conditions $U \times V \subset \mathbb{R}^4 \times \mathbb{R}^q$ such that $\gamma \subset h(U)$ and such that the closed-loop system satisfies **PF1**, **PF2** and **PF3**. The dimension q of the controller state ζ is not fixed *a priori*. It will be determined based on analysis of the path following manifold which we discuss in the next section.

3.2 Dynamic Extension

The path following manifold, denoted $\Gamma^\star$, associated with the curve γ is the maximal controlled invariant subset of the lift Γ. Physically it consists of all those motions of the car-like robot (3.1) for which the output signal (3.2) can be made to remain on the curve γ by suitable choice of control signal [42]. The path following manifold is the key object that allows one to treat the path following problem as a set stabilization problem. If the path following manifold can be made attractive and controlled invariant, then **PF1** and **PF2** will be satisfied.

When we apply the above definition to the car-like robot, or more generally to any drift-less system, it is immediate that $\Gamma^\star = \Gamma$. This is because one can trivially make the entire set Γ controlled invariant by setting the translational speed v to zero. Specifically, in the case of the car, the equation

$$(\partial_{x_1}\alpha \cos x_3 + \partial_{x_2}\alpha \sin x_3)\, v = 0$$

can always be solved by choosing $v = 0$. Where, $\partial_{x_1}\alpha$ and $\partial_{x_2}\alpha$ represent partial derivatives of α with respect to x_1 and x_2 respectively. From the point of view of mobile robots, this is not a useful characterization because such a controller causes the system to stop and hence not traverse the curve.

On the other hand, when $v \neq 0$ is fixed, the path following manifold can be characterized [43] using the steering input ω. In fact, in [43], it was shown that for the system (3.1) with v fixed, the function $\alpha = s \circ h$ yields a well defined relative degree of 3 at each point on the path. This fact was used to apply transverse feedback linearization to stabilize the path following manifold and thereby solve the path following problem. As discussed in Chapter 2, the main deficiency with the solution presented in [43] is that **PF3** cannot be satisfied. In particular, since v is fixed, the motion on the path is fixed. Similar to Section 2.2, we provide a solution that removes this deficiency for the car-like robot.

The reason that the solution presented in Section 2.1 exhibits the above deficiency is because the path following manifold $\Gamma^\star$ is one dimensional, i.e., a curve in the state space of (3.1). Since motion on $\Gamma^\star$ corresponds to motion along the path in the output space, when $\Gamma^\star$ is one dimensional, and v is fixed, there are no degrees of freedom to alter the motion along the path. Here we use dynamic extension to increase the dimension of the path following manifold in order to control the motion along the path.

Consider once again the model of a car like robot (3.1). The control objective is to make the output y approach and traverse the curve γ. Making $y \to \gamma$ is equivalent to making the state x of (3.1) approach the set Γ. Let $v = \mathrm{v} > 0$ be fixed. In [43] it was shown that for system (3.1), a path satisfying Assumptions 1 and 2, the path following manifold is given by

$$\Gamma^\star = \left\{ x \in \mathbb{R}^4 : x = \left(\sigma(\lambda), \varphi(\lambda), \arctan\left(\frac{\ell}{\mathrm{v}} \dot{\varphi}(\lambda) \right) \right), \lambda \in \mathbb{D} \right\} \tag{3.4}$$

where $\varphi(\lambda) = \arg\left(\sigma_1' + j\sigma_2'\right)$ is the angle $\sigma'(\lambda)$ makes with the y_1 axis. Another way to characterize the largest controlled invariant set $\Gamma^\star \subset \Gamma$ is to take the function $\alpha = s \circ h$ as the output of system (3.1) and check that this function yields a well defined relative degree of three.

Let $n^\star := \dim(\Gamma^\star)$. In this case $n^\star = 1$ and its co-dimension is $n - n^\star = 3$. Let $r^\star = 1$ denote the derivative of α at which the control input v appears. We use dynamic extension to generate a controller of the form (3.3) and thereby increase the dimension of the closed-loop system so that the dimension and co-dimension of $\Gamma^\star$ are equal. In other words, we delay the appearance of the input v in the derivatives of α so that ω and the delayed version of v appear in the same derivative. This effectively increases the dimension of the path following manifold; we call this approach tangential dynamic extension. This goal can be achieved if we increase the dimension of $\Gamma^\star$ by two which suggest we pick $q = n - n^\star - r^\star = 2$ in (3.3) so that the control law has two states $\zeta = (\zeta_1, \zeta_2)$.

Let $v = \mathrm{v} + \zeta_1$, where ζ_1 is the first state of our dynamics controller. In general [29] we are free to choose any dynamics for $\dot{\zeta}_1$ but we take the simplest possible structure for the control law (3.3) and let $\dot{\zeta}_1 = \zeta_2$. In order to finish defining the control law we let $\dot{\zeta}_2 = u_1$ where u_1 is a new, auxiliary input that we will use to indirectly change the translational velocity v. The structure of the control law so far is

$$\begin{aligned} \dot{\zeta}_1 &= \zeta_2 \\ \dot{\zeta}_2 &= u_1 \\ v &= \mathrm{v} + \zeta_1 \\ \omega &= u_2. \end{aligned} \tag{3.5}$$

For the extended system the path following manifold is given by

$$\Gamma^\star = \left\{ (x,\zeta) \in \mathbb{R}^4 \times \mathbb{R}^2 : x = \left(\sigma(\lambda), \varphi(\lambda), \arctan\left(\frac{\ell}{\mathrm{v}} \dot{\varphi}(\lambda) \right) \right), \lambda \in \mathbb{D} \right\}$$

To simplify notation we will no longer distinguish between states of the system (x_1, x_2, x_3, x_4) and states of the controller (ζ_1, ζ_2). Let $x_5 := \zeta_1$, $x_6 := \zeta_2$. Therefore the system we study is given by

$$\begin{aligned} \dot{x} &= f(x) + g_1(x)u_1 + g_2(x)u_2 \\ &= \begin{bmatrix} (\mathrm{v} + x_5)\cos x_3 \\ (\mathrm{v} + x_5)\sin x_3 \\ \frac{(\mathrm{v}+x_5)}{\ell}\tan x_4 \\ 0 \\ x_6 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} u_2 \end{aligned} \tag{3.6}$$

Our objective is to now design the control law $u = (u_1, u_2)$ to solve the the path following problem. Stabilizing the path following manifold in extended coordinates remains the key way to accomplish **PF1** and **PF2**. As observed in Section 2.2 that since v is not fixed and because the path following manifold has dimension three, we expect to able to control the motion along the path in order to satisfy **PF3**.

3.3 Path Following Control Design

We treat path following problem as a set stabilization problem and we follow the general approach of [42] for designing path following controllers, see also [26]. In order to satisfy **PF1** and **PF2**, we first stabilize the path following manifold $\Gamma^\star$. Once the path following manifold has been stabilized we use any remaining freedom in the control law to impose desired dynamics on the path.

We find a particular "virtual" output function for the system (3.6) and show that it yields a well defined relative degree. The benefit of using this physically meaningful output is that it facilitates control design. In this case the output yields a well defined relative degree of $\{3, 3\}$ and hence system (3.6) is feedback linearizable.

Before implementing the above program we must introduce a projection operator in the output space of the car. This operator associates to each point y in the output space of (3.1) sufficiently close to the path γ a number in $\mathbb{D}$. Let $\gamma_\epsilon \subset \mathbb{R}^2$ denote a tubular neighbourhood of the curve γ. The tubular neighbourhood has the property that if $y \in \gamma_\epsilon$

then there exists a $y^\star \in \gamma$ that is closest to y. The tubular neighbourhood allows us to define the function

$$\begin{aligned} \varpi :&\gamma_\epsilon \to \mathbb{D} \\ &y \mapsto \arg\inf_{\lambda\in\mathbb{D}} \|y - \sigma(\lambda)\|. \end{aligned} \tag{3.7}$$

It can be shown in Figure 3.2. This function is as smooth as σ which we assume to be at

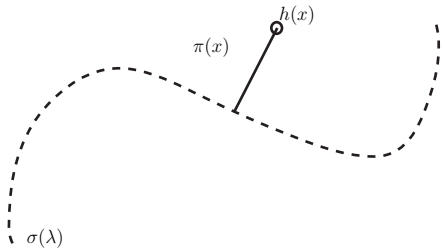


Figure 3.2: Argument that minimizes the distance from the curve.

least C^3. Using the above map we define the "virtual" output function

$$\hat{y} = \begin{bmatrix} \pi(x) \\ \alpha(x) \end{bmatrix} = \begin{bmatrix} \varpi \circ h(x) \\ s \circ h(x) \end{bmatrix}. \tag{3.8}$$

We now show that as long as the car does not have zero translational speed, then this output yields a well defined relative degree along the path.

Lemma 3.3.1. *The dynamic extension of the car-like robot* (3.6) *with output* (3.8) *yields a well defined vector relative degree of* $\{3, 3\}$ *at each point on* $\Gamma^\star$ *where* $x_5 = \zeta_1 \neq -\mathrm{v}$.

Proof. Let $x^\star \in \Gamma$ be arbitrary. By definition of Γ the output $h(x^\star)$ is on the path γ. Let $\lambda^\star \in \mathbb{D}$ be such that $h(x^\star) = \sigma(\lambda^\star)$. By the definition of vector relative degree we must show that

$$L_{g_1} L_f^i \pi(x) = L_{g_2} L_f^i \pi(x) = L_{g_1} L_f^i \alpha(x) = L_{g_2} L_f^i \alpha(x) = 0$$

for $i \in \{0, 1\}$ in a neighbourhood of $x^\star$ and that the decoupling matrix

$$D(x^\star) = \begin{bmatrix} L_{g_1} L_f^2 \pi(x^\star) & L_{g_2} L_f^2 \pi(x^\star) \\ L_{g_1} L_f^2 \alpha(x^\star) & L_{g_2} L_f^2 \alpha(x^\star) \end{bmatrix} \tag{3.9}$$

is nonsingular. Since

$$\frac{\partial \pi(x)}{\partial x_i} = \frac{\partial \alpha(x)}{\partial x_i} \equiv 0$$

for $i \in \{3, 4, 5, 6\}$, it is easy to check that $L_{g_j} L_f^i \pi(x) = L_{g_j} L_f^i \alpha(x) = 0$ for $i \in \{0, 1\}$, $j \in \{1, 2\}$.

To show that the decoupling matrix is full rank, it suffices to show that the determinant of $D(x^\star)$ is not zero. Direct calculations yield

$$\begin{aligned} L_{g_1} L_f^2 \alpha &= \tfrac{(\mathrm{v}+x_5)^2}{\ell}(\partial_{x_2}\alpha \cos x_3 - \partial_{x_1}\alpha \sin x_3)\sec^2 x_4 \\ L_{g_2} L_f^2 \alpha &= \partial_{x_1}\alpha \cos x_3 + \partial_{x_2}\alpha \sin x_3 \\ L_{g_1} L_f^2 \pi &= \frac{(\mathrm{v}+x_5)^2}{\ell}(1+\tan^2 x_4)(\sigma_2' \cos x_3 - \sigma_1' \sin x_3) \\ L_{g_2} L_f^2 \pi &= \sigma_1' \cos x_3 + \sigma_2' \sin x_3 \end{aligned} \tag{3.10}$$

where $\sigma_i' = \frac{\partial \sigma_i}{\partial \lambda}\big|_{\lambda=\lambda^\star}$, $i \in \{1,2\}$. Hence

$$\det\left(D(x)\right) = \frac{(\mathrm{v}+x_5)^2}{\ell \cos^2 x_4}\left[\sigma_1' \partial_{x_2}\alpha - \sigma_2' \partial_{x_1}\alpha\right]. \tag{3.11}$$

The only way for this determinant to vanish is if either (i) $\mathrm{v} = -x_5$ or (ii) $\sigma_1' \partial_{x_2}\alpha - \sigma_2' \partial_{x_1}\alpha = 0$. We argue that condition (ii) never occurs on the path because the vectors $\mathrm{col}(\partial_{x_1}\alpha, \partial_{x_2}\alpha)$ and σ' are orthogonal.

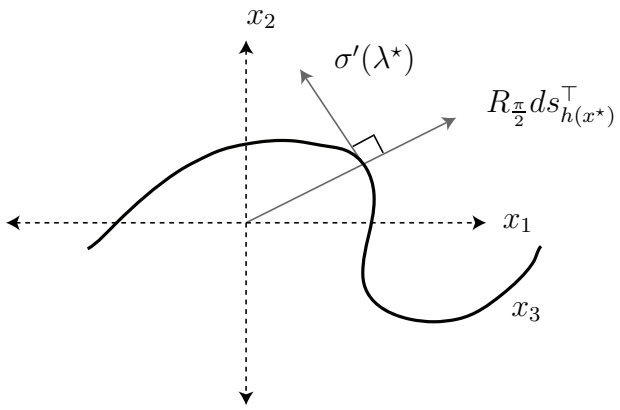


Figure 3.3: Representation of vector $\sigma'(\lambda^\star)$ orthogonal to $R_{\frac{\pi}{2}} ds^\top_{h(x^\star)}$.

By the chain rule and the form of the output map (3.2)

$$\begin{bmatrix} \partial_{x_1}\alpha \\ \partial_{x_2}\alpha \end{bmatrix}_{x=x^\star} = \begin{bmatrix} \partial_{y_1} s \\ \partial_{y_2} s \end{bmatrix}_{y=h(x^\star)} = ds^\top_{h(x^\star)}.$$

By Assumption 2 the differential ds_y is non zero when $y \in \gamma$. Thus the vector $ds^\top_{h(x^\star)}$ is a non zero gradient vector and is orthogonal to the path at $h(x^\star)$. On the other hand the vector $\sigma'(\lambda^\star)$ is non zero because σ is regular and also tangent to the curve. Hence

$ds_{h(x^\star)}\sigma'(\lambda^\star) = 0$. If we rotate the vector $ds^\top_{h(x^\star)}$ by $\pi/2$ radians then the rotated vector and σ' will be linearly dependent. Let $R_{\frac{\pi}{2}}$ be a rotation by $\pi/2$. Then

$$R_{\frac{\pi}{2}} ds^\top_{h(x^\star)} = k(\sigma(\lambda^\star))\sigma'(\lambda^\star)$$

for some smooth, scalar valued, non zero function $k : \mathbb{R}^2 \to \mathbb{R}$. The function k is never equal to zero because the vector $ds^\top_{h(x^\star)}$ is never zero.

Returning to the expression for $\det(D(x))$, we have that

$$\begin{aligned}
\sigma_1'\partial_{x_2}\alpha - \sigma_2'\partial_{x_1}\alpha &= \left(R_{\frac{\pi}{2}} ds^\top_{h(x^\star)}\right)^\top \sigma'(\lambda^\star) \\
&= \left(k(\sigma(\lambda^\star))\sigma'(\lambda^\star)\right)^\top \sigma'(\lambda^\star) \\
&= k(\sigma(\lambda^\star))\|\sigma'(\lambda^\star)\|^2 \\
&= k(\sigma(\lambda^\star).
\end{aligned}$$

We have shown for any $x^\star \in \Gamma$ with $x_5 \neq -\mathrm{v}$ that $\det(D(x^\star)) \neq 0$. Since $\Gamma^\star \subset \Gamma$, the lemma is proved. □

An immediate consequence of Lemma 3.3.1 is that it allows us to define a local diffeomorphism using the function $\pi(x)$ and $\alpha(x)$ and their iterated Lie derivatives along the vector field $f(x)$.

Corollary 3.3.2. *Let $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^6 : x_5 + \mathrm{v} = 0\}$. There exists a neighbourhood $U \subset \mathbb{R}^6$ containing $x^\star$ such that the mapping $T : U \subset \mathbb{R}^6 \to T(U) \subset \mathbb{R}^6$, defined by*

$$\begin{bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \\ \xi_1 \\ \xi_2 \\ \xi_3 \end{bmatrix} = T(x) = \begin{bmatrix} \pi(x) \\ L_f\pi(x) \\ L_f^2\pi(x) \\ \alpha(x) \\ L_f\alpha(x) \\ L_f^2\alpha(x) \end{bmatrix} \tag{3.12}$$

is a diffeomorphism onto its image.

Proof. Let $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^6 : x_5 + \mathrm{v} = 0\}$. By Lemma 3.3.1 system (3.6) with output (3.8) yields a well defined vector relative degree of $\{3, 3\}$ at $x^\star$. By[29, Lemma 5.2.1] the row vectors

$$\begin{aligned}
&d\alpha(x^\star), dL_f\alpha(x^\star), dL_f^2\alpha(x^\star) \\
&d\pi(x^\star), dL_f\pi(x^\star), dL_f^2\pi(x^\star)
\end{aligned} \tag{3.13}$$

are linearly independent. These are the rows of the Jacobian matrix $dT_{x^\star}$ which implies that $dT_{x^\star}$ is nonsingular. By the inverse function theorem 2.1.2, T is a diffeomorphism onto its image. □

Using the coordinate transformation T from Corollary 3.3.2, in a neighbourhood of any point $x^\star \in \Gamma$, the system (3.6) in (η, ξ) coordinates reads

$$\begin{aligned}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= \eta_3 \\
\dot{\eta}_3 &= L_f^3\pi + L_{g_1}L_f^2\pi u_1 + L_{g_2}L_f^2\pi u_2\big|_{x=T^{-1}(\eta,\xi)} \\
\dot{\xi}_1 &= \xi_2 \\
\dot{\xi}_2 &= \xi_3 \\
\dot{\xi}_3 &= L_f^3\alpha + L_{g_1}L_f^2\alpha u_1 + L_{g_2}L_f^2\alpha u_2\big|_{x=T^{-1}(\eta,\xi)}
\end{aligned} \tag{3.14}$$

The structure of the system in the new coordinates is similar to the structure of unicycle in the transformed coordinates formulated in Section 2.2. Similarly, stabilizing $\xi = 0$ cause $\alpha(x)$, $\dot{\alpha}(x)$ and $\ddot{\alpha}(x)$ to converge to zero. This is equivalent to getting the car on the desired path with heading velocity tangent to the path. On the path following manifold the motion of the car-like robot on the path is governed by the η-dynamics. When the robot is on the path following manifold, i.e., $\xi = 0$ then η_1 determines the position of the robot on the path, η_2 represent velocity of the robot along the path and η_3 represent acceleration of the robot along the path.

Consider the regular feedback transformation similar to 2.22,

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} := D^{-1}(x)\left(\begin{bmatrix} -L_f^3\pi \\ -L_f^3\alpha \end{bmatrix} + \begin{bmatrix} v^{\parallel} \\ v^{\pitchfork} \end{bmatrix},\right) \tag{3.15}$$

where $(v^{\parallel}, v^{\pitchfork}) \in \mathbb{R}^2$ are auxiliary control inputs. By Lemma 3.3.1, this controller is well defined in a neighbourhood of every $x^\star \in \Gamma \backslash \{x \in \mathbb{R}^6 : x_5 + \mathrm{v} = 0\}$. Thus in a neighbourhood of $x^\star$, the closed-loop system becomes

$$\begin{aligned}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= \eta_3 \\
\dot{\eta}_3 &= v^{\parallel} \\
\dot{\xi}_1 &= \xi_2 \\
\dot{\xi}_2 &= \xi_3 \\
\dot{\xi}_3 &= v^{\pitchfork}
\end{aligned} \tag{3.16}$$

Where $v^{\pitchfork}$ and $v^{\parallel}$ are the transversal and tangential input. The control law (3.15) has decoupled the transversal and tangential subsystems which makes designing $(v^{\parallel}, v^{\pitchfork})$ to solve the path following problem particularly easy. In summary, dynamic extension and

transverse feedback linearization allows us to represent the system as a linear time-invariant system (LTI) and use LTI controller design techniques to design the controller for system (3.16). Another way to state this is to say that the output (3.8) is a differentially flat output for the car-like robot (3.1) [59, 37]. However, in our case we choose a geometrically meaningful output function. This gives us the physical intuition of the system and the path. In Chapter 4 we show that this procedure cannot always be applied in a straight forward manner to every differentially flat mobile robot.

3.3.1 Transversal and Tangential Controller Design

For the transversal subsystem, we use the controller similar to the one we used for the case of unicycle in chapter 2,

$$v^{\pitchfork}(\xi) = k_1\xi_1 + k_2\xi_2 + k_3\xi_3, \tag{3.17}$$

with $k_i < 0$, $i \in \{1, 2, 3\}$. It is easy to observe that the controller (3.17) exponentially stabilizes the LTI transversal subsystem and forces all the ξ-states go to zero. Physically, since $\xi = 0$ is an equilibrium of the closed-loop transversal subsystem, if the robot is initialized on the path with the initial velocity tangent to the path, then it will remain on the path for all future time. Hence, the property of path invariance is achieved.

In order to achieve the goal of controlling the speed of the robot on the curve, we use a controller similar to (2.24) used in the last chapter,

$$v^{\parallel}(\eta) = k_4(\eta_1 - \eta_1^{ref}) + k_5(\eta_2 - \eta_2^{ref}) + k_6\eta_3, \tag{3.18}$$

where $k_i < 0$, $i \in \{4, 5, 6\}$. The parameter η_1^{ref} is the desired reference position on the path and η_2^{ref} is a desired reference velocity profile. Note however that whenever $x_5 = -\mathrm{v}$, the robot has no translational velocity. In that case, the decoupling matrix loses rank and the control law (3.15) is not well defined. Hence, we cannot stabilize a particular point on the curve using this control law and henceforth we set $k_4 = 0$. The overall control scheme is presented in Figure 3.4.

3.4 Implementation Issues

In order to implement the controller described in Section 3.3, we must compute the coordinate transformation $(x_1, x_2, x_3, x_4, x_5, x_6) \mapsto (\xi_1, \xi_2, \xi_3, \eta_1, \eta_2, \eta_3)$ defined in (3.12), the feedback (3.15) with $D(x)$ defined in (3.9) and the transversal and tangential controllers (3.17), (3.18).

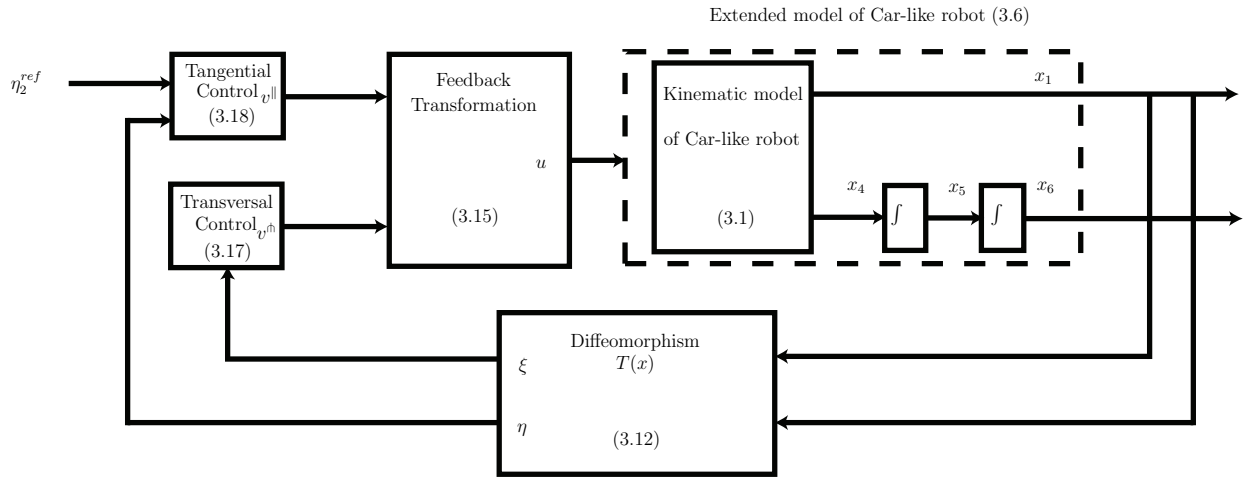


Figure 3.4: Feedback control system of car-like robot with equation references.

3.4.1 Computation of Transversal States

We assume[1] that we have a zero level set representation of the curve γ. Hence we know the function $\alpha(x) = s \circ h(x)$ and therefore ξ_1, ξ_2 and ξ_3 can be computed symbolically. Similarly the terms $L_f^3\alpha$, $L_{g_1}L_f^2\alpha$ and $L_{g_2}L_f^2\alpha$ can be computed easily to at least partially define the feedback (3.15).

3.4.2 Computation of Tangential States

The states η_1, η_2 and η_3 are slightly more complicated to compute. We have assumed, without loss of generality, that the given parameterized curve is unit-speed. The utility of the tangential controller is very much related to the unit-speed parameterization of the curve σ. For if σ is not a unit-speed curve then it is difficult to give physical intuition to the parameter η_2^{ref} in (3.18). In particular, if σ is not unit-speed, then setting η_2^{ref} to a constant value will not lead to uniform speed along the path and may cause the velocity of the vehicle to grow and become unbounded.

Example 3.4.1. *Suppose a regular curve is given in parametric form* $\sigma : \mathbb{R} \to \mathbb{R}^2$, $\lambda \mapsto \operatorname{col}(\lambda^4 - \lambda + 1, \lambda^3 + \lambda + 1)$. *This curve has implicit representation* $\gamma = s^{-1}(0)$ *with*

$$s(y) = -y_2^4 + y_1^3 + 7y_2^3 + 4y_1y_2^2 - 5y_1^2 - 22y_2^2 - 13y_1y_2 + 18y_1 + 34y_2 - 23.$$

It can be shown that Lemma 3.3.1 and Corollary 3.3.2 hold even though σ *is not unit-speed. Now suppose that we put the system into the form* (3.16) *and apply the control*

[1] In Chapter 5 we give a procedure to obtain a zero level representation of a given parameterized curve.

laws (3.17), (3.18). *If we set* $\eta_2^{ref} = k$ *for some constant* $k \neq 0$*, then the velocity of the car becomes unbounded as the vehicle moves along the path. This is because, for points on the curve far away from* $\sigma(0) = (1, 1)$*, small changes in the parameter* λ *result in very large changes in position along the path. A controller that maintains a constant* η_2 *will cause* η_1*, the path parameter, to track a ramp and will require the translational velocity* $x_5 + \mathrm{v}$ *to become unbounded. Thus, while theoretically a non unit-speed parameterized curve poses no obstacle, a unit-speed curve greatly simplifies tangential controller design.*

In general a regular parameterized curve is not given with unit-speed parameterization and finding a closed form expression for the unit-speed parameterization may be difficult or even impossible. Let $\widetilde{\sigma} : \mathbb{R} \to \mathbb{R}^2$ be the given C^r, $r \geq 3$ curve that satisfies assumptions 1 and 2. We do not assume that $\widetilde{\sigma}$ has unit-speed parameterization. If $\widetilde{\sigma}$ models a closed curve then for some $T > 0$ it is true that for all $\lambda \in \mathbb{R}$ $\widetilde{\sigma}(\lambda + T) = \widetilde{\sigma}(\lambda)$, i.e., the curve is T periodic.

Let $V \subseteq \mathbb{R}^2$ be a neighbourhood of $\mathbb{R}^2$ such that $V \cap \gamma$ contains a single connected component of γ. Since γ is a one dimensional manifold, such a V exists. If γ is a closed curve then we can take V such that $\gamma \subset V$.

If γ is a non closed curve let $\mathcal{I} := (\lambda_s, \lambda_e) \subset \mathbb{R}$ be an interval of the real line such that $\gamma \cap V = \widetilde{\sigma}(\mathcal{I})$. Since we can always reparameterize $\widetilde{\sigma}$, without loss of generality we let $\lambda_s = 0$. If γ is closed we take $\mathcal{I} = [0, T)$. Let L denote the length of the portion of the curve in V. Now introduce a projection operator,

$$\lambda^\star = \widetilde{\varpi}(y) = \arg \inf_{\lambda \in \mathcal{I}} \|y - \widetilde{\sigma}(\lambda)\| \tag{3.19}$$

defined in γ_ϵ. To calculate the first tangential state we must find the unit length parameter so we let

$$\eta_1 = g(\lambda^\star) := \int_0^{\lambda^\star} \left\| \frac{d\sigma}{d\lambda} \right\| du \tag{3.20}$$

so that $\eta_1 = g \circ \widetilde{\varpi} \circ h(x)$. To calculate η_2 we note

$$\begin{aligned} \eta_2 &= \frac{\partial (g \circ \widetilde{\varpi} \circ h)}{\partial x} \frac{dx}{dt} \\ &= \left(\frac{\partial g}{\partial \lambda} \right) \Big|_{\lambda = \lambda^\star} \left(\frac{\partial \widetilde{\varpi}}{\partial y} \right) \Big|_{y = h(x)} \begin{bmatrix} (v + x_5) \cos(x_3) \\ (v + x_5) \sin(x_3) \end{bmatrix}. \end{aligned}$$

Simple geometric arguments, similar to those used in the proof of Lemma 3.3.1, show that $\frac{\partial \widetilde{\varpi}}{\partial y} \Big|_y$ is given by

$$\frac{\partial \widetilde{\varpi}}{\partial y} = \frac{(\sigma'(\lambda^\star))^\top}{\|\sigma'(\lambda^\star)\|^2}.$$

Differentiating (3.20) one obtains

$$\left.\frac{\partial g}{\partial \lambda}\right|_{\lambda=\lambda^\star} = \|\sigma'(\lambda^\star)\|$$

and so

$$\eta_2 = \frac{(\sigma'(\lambda^\star))^\top}{\|\sigma'(\lambda^\star)\|} \begin{bmatrix} (v+x_5)\cos(x_3) \\ (v+x_5)\sin(x_3) \end{bmatrix}. \tag{3.21}$$

To simplify notation let

$$\Delta(x) := \frac{(\sigma'(\lambda^\star))^\top}{\|\sigma'(\lambda^\star)\|},\ \Omega := \begin{bmatrix} (v+x_5)\cos(x_3) \\ (v+x_5)\sin(x_3) \end{bmatrix}.$$

To find η_3 we differentiate (3.21), $\dot{\eta}_2 = \dot{\Delta}\Omega + \Delta\dot{\Omega}$. The term $\dot{\Omega}$ is easy to compute using the system dynamics (3.6).

$$\dot{\Omega} = \frac{\partial \Omega}{\partial x}\dot{x} = L_f\Omega = \begin{bmatrix} -\frac{(v+x_5)^2}{\ell}\sin(x_3)\tan(x_4) \\ \frac{(v+x_5)^2}{\ell}\cos(x_3)\tan(x_4) \end{bmatrix} \tag{3.22}$$

The term $\dot{\Delta} = \Delta'\dot{\lambda}$ can be found by noting that

$$\Delta' := \frac{\partial \Delta}{\partial \lambda} = \frac{(\sigma'')^\top \|\sigma'\|^2 - (\sigma')^\top \sum_{i=1}^2 \sigma_i'\sigma_i}{\|\sigma'\|^3} \tag{3.23}$$

and, using (3.20) and the chain rule,

$$\dot{\lambda} = \frac{1}{\|\sigma'\|^2}\eta_2. \tag{3.24}$$

This shows that the tangential state η_3 can be computed effectively using (3.4.2), (3.23), (3.24), Ω and $\dot{\Omega}$. Finally, in order to implement the feedback transformation (3.15) we must find expressions for $L_f^3\pi$ and the first row of the decoupling matrix (3.9). The entries of the decoupling matrix are given by (3.10). Taking the time derivatives of η_3, tedious, yet easy, calculations give

$$\dot{\eta}_3 = \Delta'\dot{\Omega}\frac{\eta_2}{\|\sigma'\|} + \frac{\dot{\eta}_2}{\|\sigma'\|}\Delta'\Omega + \frac{\eta_2}{\|\sigma'\|}\frac{d\Delta'}{dt}\Omega + \eta_2\Delta'\Omega\frac{\sum_{i=1}^2 \sigma_i'\sigma_i''}{\|\sigma'\|^3}\dot{\lambda} + \dot{\Delta}\dot{\Omega} + \Delta\ddot{\Omega},$$

where

$$\frac{d\Delta'}{dt} = \Delta''\dot{\lambda} = \Delta''\frac{\eta_2}{\|\sigma'\|},$$

and

$$\Delta'' = \frac{1}{\|\sigma'\|^6}\left[\|\sigma'\|^3\left\{(\sigma''')^\top\|\sigma'\| + (\sigma'')^\top\left(\sum_{i=1}^{2}\sigma_i'\sigma_i''\right) - (\sigma')^\top\sum_{i=1}^{2}\left(\sigma_i''^2 + \sigma_i'\sigma_i'''\right)\right\}\right]$$
$$-\frac{3}{\|\sigma'\|^6}\left[\|\sigma'\|^2\left(\sum_{i=1}^{2}\sigma_i'\sigma_i''\right)\left\{(\sigma'')^\top\|\sigma'\| - (\sigma')^\top\left(\sum_{i=1}^{2}\sigma_i'\sigma_i''\right)\right\}\right],$$

$\ddot{\Omega}$ is given by

$$\ddot{\Omega} := \Omega_1 + \Omega_2$$

where

$$\Omega_1 := \begin{bmatrix} -\frac{1}{l}\sin x_3(1+\tan^2 x_4)(v+x_5)^2u_1 + \cos x_3 u_2 \\ \frac{1}{l}\cos x_3(1+\tan^2 x_4)(v+x_5)^2u_1 + \sin x_3 u_2 \end{bmatrix}$$

and,

$$\Omega_2 := \begin{bmatrix} -\dot{x}_3\frac{1}{l}\cos x_3\tan x_4(v+x_5)^2 - \dot{x}_3x_6\sin x_3 \\ -\dot{x}_3\frac{1}{l}\sin x_3\tan x_4(v+x_5)^2 - \dot{x}_3x_6\cos x_3 \end{bmatrix} +$$
$$\begin{bmatrix} -2\dot{x}_5\frac{1}{l}\sin x_3\tan x_4(v+x_5) \\ 2\dot{x}_5\frac{1}{l}\cos x_3\tan x_4(v+x_5) \end{bmatrix}.$$

$$L_f^3\pi = \frac{\eta_2}{\|\sigma'\|}\left(\Delta'\dot{\Omega} + \Delta'\Omega + \frac{d\Delta'}{dt}\Omega + \frac{\eta_2\sum_{i=1}^{2}\sigma_i'\sigma_i}{\|\sigma'\|^3}\Delta'\Omega\right) \tag{3.25}$$
$$+ \dot{\Delta}\dot{\Omega} + \Delta\Omega_2.$$

Implementation of controller and the regular feedback (3.15) is summarized by Algorithm 1.

3.4.3 Experimental Implementation

In this thesis, the proposed controller was not tested on a real robot. In Section 3.5, we show that the controller works in simulation. In order to implement this controller on an experimental setup, full state feedback is needed. In the case of an indoor car-like robot, the position (x_1, x_2) of the robot can be determined using a camera placed on the ceiling. A marker can be placed on the top of the robot and the orientation x_3 of the robot can be determined with the help of the image. The camera will then feed the image data to a computer that will send the localization data to the mobile robot. The steering angle x_4 can be determined by adding a potentiometer or optical encoder at the steering wheel. Theses sensors directly give the angle of the steering wheel. Once we determine position, orientation and steering angle we have all the states of the system. Due to unmodeled

input : $\sigma(\lambda) : \mathbb{R} \to \mathbb{R}^2$
$\sigma(\mathbb{D}) = \{y \in \mathbb{R} : s(y) = 0\}$
System model (3.6)
Current state $x \in \mathbb{R}^6$
output: (u_1, u_2)

for *each* **do**
Using (3.19) numerically calculate $\lambda^\star$.
Calculate $\sigma'(\lambda^\star)$, $\sigma''(\lambda^\star)$, $\sigma'''(\lambda^\star)$, $\|\sigma'(\lambda^\star)\|$.
Numerically calculate η_1 using (3.20).
Calculate η_2 using (3.21).
Calculate η_3 using (3.4.2), (3.23), (3.24), Ω and $\dot{\Omega}$. Calculate $L_f^3\pi$ using expression (3.25).
Calculate ξ_1, ξ_2, ξ_3, $L_f^3\alpha$.
Compute decoupling matrix $D(x)$ using (3.10) Compute $(u_1 u_2)$ given by (3.15), (3.17), (3.18) using the above expressions.
end

Algorithm 1: Control Algorithm

parameters like friction of tires and error in accurate estimation of robot position and orientation we may expect some discrepancies between experimental and simulated results. It is important to note that since high gains were used to stabilize the ξ-states, in practice it could saturate motor voltages. That means, practically we cannot make the convergence to the desired path too fast. Placing the camera on the ceiling is quite restrictive, this approach cannot be used for outdoor path following purposes. Moreover, for indoor path following the path cannot be followed beyond the field of view of the robot. To estimate the position of the car-like robot, a GPS can be used. However, there are limitation on the accuracy of the GPS. There are methods available in the literature where GPS is used in conjunction with inertial navigation system (consisting of gyroscopes and accelerometers) to get better estimation of position. Another approach could be to install the camera at the top of the robot. In this case, the estimation of robot position and orientation would be difficult. There are techniques available in the literature for indoor and outdoor navigation, like simultaneous localization and mapping.

3.5 Simulation Results

In this section, simulation results of a car-like robot following curves are presented.

3.5.1 Simulation I

We simulate the car-like robot (3.1) with dynamic controller (3.5) and feedback law (3.15), (3.17), (3.18) where (η,ξ) are defined in (3.12). The controller in these simulations is implemented using Algorithm 1. Consider the curve $\sigma : \mathbb{R} \to \mathbb{R}^2,\ \lambda \mapsto \mathrm{col}(\lambda, \cos(\lambda))$ with implicit representation $\gamma = \{y \in \mathbb{R}^2 : s(y) = y_2 - \cos(y_1) = 0\}$. It is important to note that along with **PF1** and **PF2**, **PF3** is also satisfied in these path following simulations. We desire to track the velocity profile along the curve given by

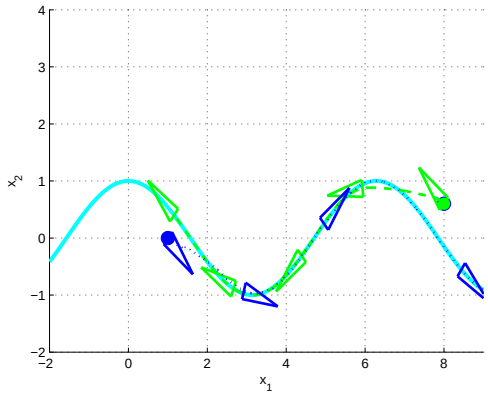
$$\eta_2^{ref} = \begin{cases} -0.5 & 0 \leq t < 10s \\ -1 & t \geq 10s. \end{cases} \tag{3.26}$$

Simulation results of the car-like robot tracking velocity profile (3.26) are shown in Figure 3.5. Position and orientation of the closed-loop system versus time is shown in Figure 3.5(a). The position of the robot is represented by a triangle while the initial position is represented with a dot at the base of the triangle. The solid curve represents the desired path, dashed line represents the output trajectory of the closed-loop system. By choosing the transversal gains $\{k_1, k_2, k_3\}$ much larger than the tangential gains $\{k_5, k_6\}$ we ensure that the exponential convergence of ξ to zero is much faster than the convergence of η_2 along the desired velocity profile. The convergence of all the ξ-states ensures that the robot is on the path. Figure 3.5(b) shows all the ξ-states converging to zero. It can be observed from Figure 3.5(c) that the robot follows the desired speed profile hence **PF3** is satisfied. Figure 3.5(d) shows the evolution of path parameter with respect to time. As we are dealing with a sinusoidal curve, the path parameter is unbounded.

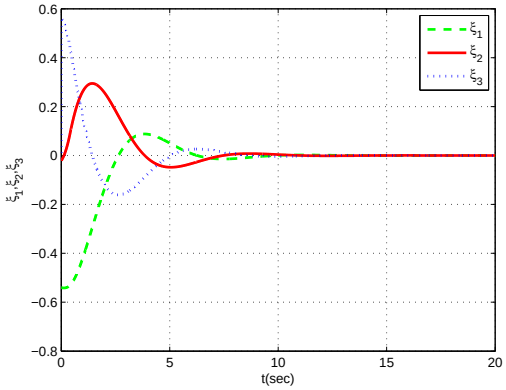
The control design procedure is divided into two separate steps, transversal controller design and tangential controller design. It is important to note that the choice of the velocity profile does not affect the transversal controller design. If we want follow a different velocity profile the transversal controller remains the same. We just need to redesign the tangential controller. Figure 3.6 shows the car-like robot following the same sinusoidal curve but tracks a velocity profile given by,

$$\eta_2^{ref} = -\frac{1}{5}\left(e^{-0.1t} \times 0.6\sin(t - \frac{\pi}{3}) + 2\right). \tag{3.27}$$

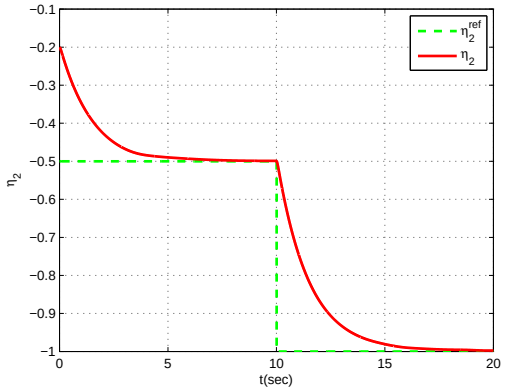
Figure 3.6(b) shows that the robot follows the desired speed profile. Since the curve is not a closed curve, the path parameter η_1 is unbounded. Figure 3.6(c) represents the variation of η_1 with respect to time. It is interesting to observe that a trajectory similar to (3.27) can be defined in terms of path parameter η_1 instead of time. As we are dealing with path following problem and we want underscore the distinction from tracking problem. It is possible to assign a speed profile according to the complexity of the path because we can track a velocity profile that is a parameter of path not a parameter of time. That means,



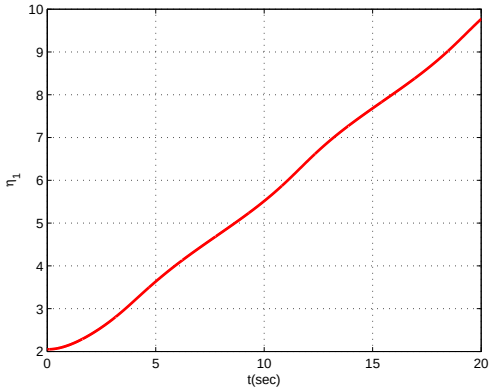
(a) Car-like robot following the sinusoidal curve.



(b) Exponential stabilization of the transversal states ξ_1, ξ_2, ξ_3.



(c) Velocity (η_2) along the path.



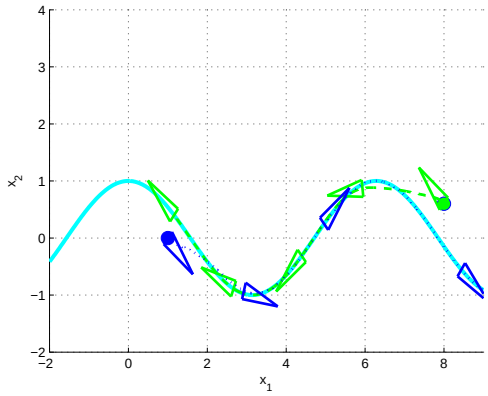
(d) Path parameter, η_1.

Figure 3.5: Car-like robot (3.6) following the sinusoidal curve $\sigma : \mathbb{R} \to \mathbb{R}^2$, $\lambda \mapsto \text{col}(\lambda, \cos(\lambda))$ while tracking velocity profile (3.26).

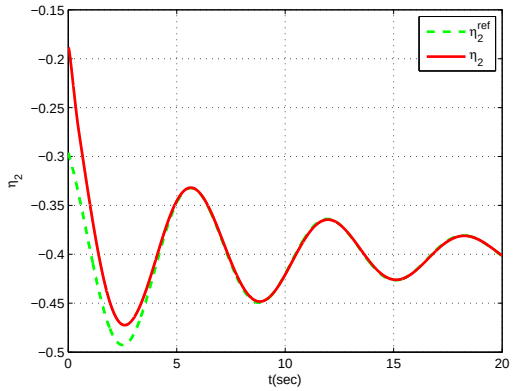
around sharp corners of the path the speed of the robot can be decreased and for straight portion of the path the speed can be increased. Since the speed profile (3.27) is defined in terms of time but it is still interesting to observe the change of η_2 with respect to η_1 because, in this case, η_1 is a monotone function just like time. The result is shown in Figure 3.6(d). Consider a trajectory defined in terms of path parameter η_1.

$$\eta_2^{ref} = \frac{1}{5}\sin\eta_1 + 2. \tag{3.28}$$

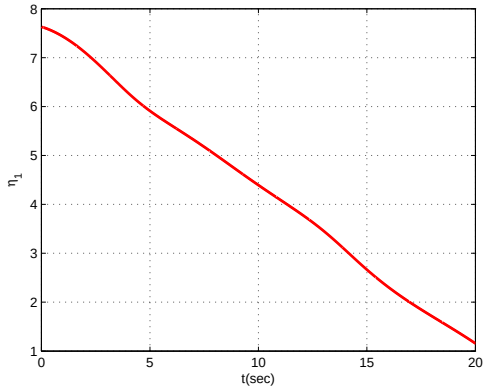
In this case, we show that the car-like robot (3.6) is following the sinusoidal path and tracking the desired speed profile (3.28) shown in Figure 3.7. The speed profile (3.28) is not a function of time but a function of path parameter.



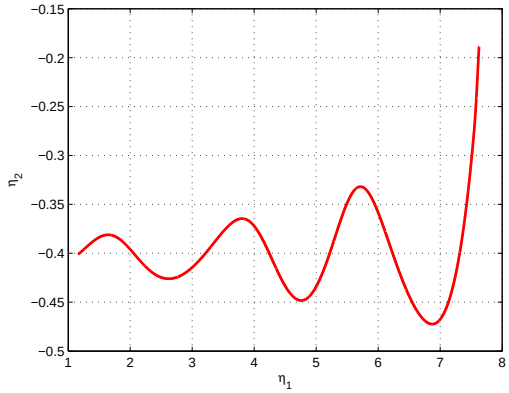
(a) Car-like robot following the sinusoidal curve.



(b) Velocity, η_2, along the path.



(c) Path parameter, η_1.

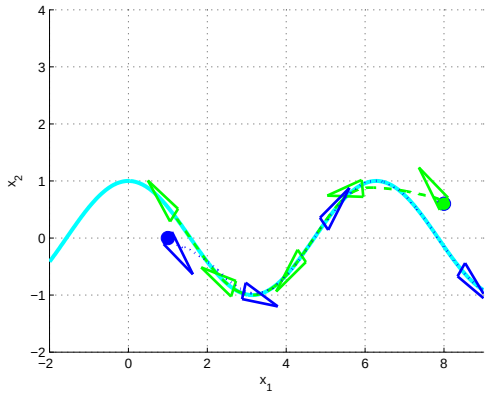


(d) Velocity (η_2) along the path Vs path parameter η_1.

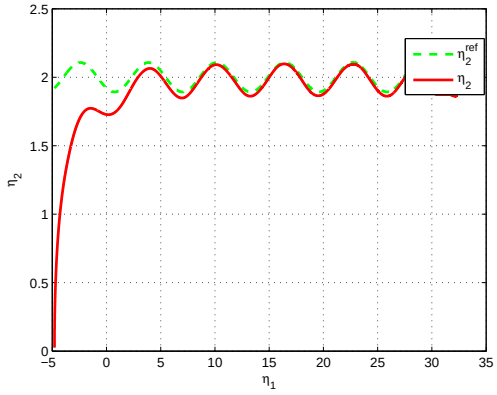
Figure 3.6: Car-like robot (3.6) following the sinusoidal curve $\sigma : \mathbb{R} \to \mathbb{R}^2$, $\lambda \mapsto \text{col}(\lambda, \cos(\lambda))$ while tracking velocity profile (3.27).

3.5.2 Simulation II

Finally we show the path following results for the curve in Example 3.4.1. The curve given in Example 3.4.1 is an example of a non closed curve, whose parametric and implicit representation is known and the curve is not a unit-speed curve. Using the procedure outlined in Algorithm 1 a controller is tested in simulation for the path given in (3.4.1)The desired speed along the curve is $\eta_2^{ref} = 0.5$. Figure 3.8(a) shows the resulting motion in the output space. The robot is initialized off the path, but the exponentially stabilized ξ-states force the vehicle to converge to the path. The speed of the car-like robot is shown in Figure 3.8(b). It is observed that the robot follows the desired speed very closely. The



(a) Car-like robot following the sinusoidal curve.



(b) Velocity, η_2, along the path.

Figure 3.7: Car-like robot (3.6) following the sinusoidal curve $\sigma : \mathbb{R} \to \mathbb{R}^2$, $\lambda \mapsto \text{col}(\lambda, \cos(\lambda))$ while tracking velocity profile (3.28).

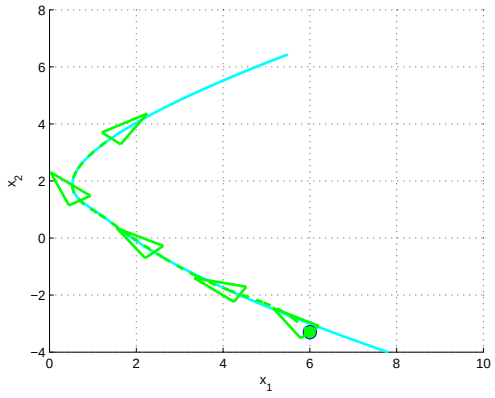
speed of the car like robot is following the desired velocity profile, $\eta_2^{ref} = 0.5$. Figure 3.8(c) represents the change of path parameter with respect to time. Since the path is not closed the path parameter η_1 is unbounded. The velocity of car-like robot is plotted against the path parameter η_1 in Figure 3.8(d). As discussed in the last simulation a desired speed profile can be defined in terms of path parameter and made to follow in the similar way.

3.6 Robustness of the Proposed Controller

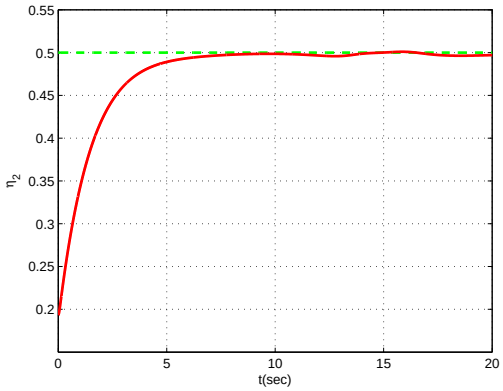
In the last section it was shown by simulations that the proposed controller works reasonably well for a large class of curves. The controller relies on cancelation of nonlinear terms and a natural question that comes to mind is how well the controller performs in the presence of modeling uncertainty. In practical applications there can be modeling inaccuracies or errors in state estimations. We assume that we know all the sates of car-like robot reasonably well. However, we analyze the performance of the proposed controller when there are modeling errors. Consider the model of the car-like robot,

$$\dot{x} = \begin{bmatrix} v\cos x_3 \\ v\sin x_3 \\ \frac{1}{\ell}\tan x_4 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u. \tag{3.29}$$

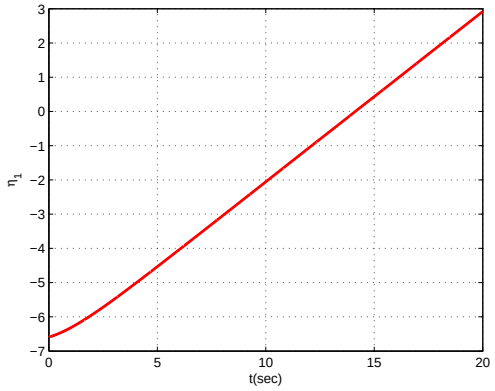
For the ease of exposition we fix the translational velocity of the robot. We call this model the nominal model. We will focus on the transverse feedback linearization of the nominal



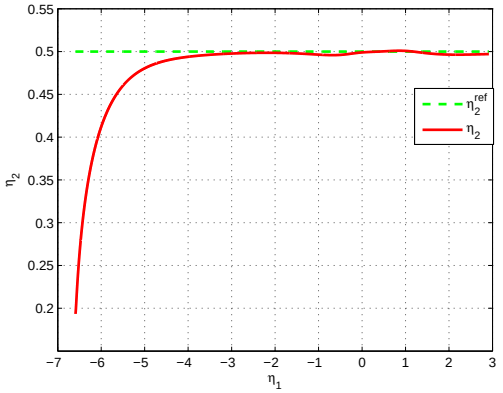
(a) Car-like robot following the curve (3.4.1).



(b) Velocity, η_2, along the path.



(c) Path parameter, η_1.



(d) Velocity (η_2) along the path Vs path parameter η_1.

Figure 3.8: Car-like robot 3.6 following the curve (3.4.1) starting from $x(0) = (6, -3, \frac{3\pi}{4}, 0, 0, 0)$ while tracking velocity profile $\eta_2^{ref} = 0.5$.

model but note that similar comments apply for the full dynamic controller derived in this chapter. Suppose, due to modeling errors, the length of the robot ℓ is not accurate. The actual length of the car-like robot is $\ell + \delta$ where δ is the modeling error. The actual model of the car-like robot becomes

$$\dot{x} = \begin{bmatrix} v\cos x_3 \\ v\sin x_3 \\ \frac{1}{\ell+\delta}\tan x_4 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u. \tag{3.30}$$

Following the procedure outlined in Section 2.1, we design a controller for the nominal system. In that case the transversal subsystem is

$$\begin{aligned}\dot{\xi}_1 &= \xi_2 \\ \dot{\xi}_2 &= \xi_3 \\ \dot{\xi}_3 &= L_f^3\alpha + L_g L_f^2 \alpha u\big|_{x=T^{-1}(\eta,\xi)}.\end{aligned} \tag{3.31}$$

and we can use,

$$u = \frac{1}{L_g L_f^2 \alpha}(-L_f\alpha^3 - k_1\xi_1 - k_2\xi_2 - k_3\xi_3), \tag{3.32}$$

where $k_i > 0$ for $i = 1, 2, 3$ to exponentially stabilize $\xi = 0$. Using the feedback control law (3.31) becomes,

$$\begin{aligned}\dot{\xi}_1 &= \xi_2 \\ \dot{\xi}_2 &= \xi_3 \\ \dot{\xi}_3 &= -k_1\xi_1 - k_2\xi_2 - k_3\xi_3.\end{aligned} \tag{3.33}$$

which is clearly exponentially stable. On the other hand, if we apply the control law (3.32) designed based on the nominal model (3.29) to the actual mode (3.30), the closed-loop transversal dynamics are given by

$$\begin{aligned}\dot{\xi}_1 &= \xi_2 \\ \dot{\xi}_2 &= \xi_3 + \psi(x) \\ \dot{\xi}_3 &= -k_1\xi_1 - k_2\xi_2 - k_3\xi_3 + \varsigma(x)\end{aligned} \tag{3.34}$$

where the smooth scalar functions $\psi(x)$, $\varsigma(x)$ arise due to the modeling error. We can write this system as a nominal system $\dot{\xi} = A\xi$ with a perturbation $p(x) = \begin{bmatrix} 0 & \psi(x) & \varsigma(x) \end{bmatrix}^\top$ term

$$\dot{\xi} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -k_1 & -k_2 & -k_3 \end{bmatrix} \begin{bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{bmatrix} + \begin{bmatrix} 0 \\ \psi(x) \\ \varsigma(x) \end{bmatrix}. \tag{3.35}$$

In this case, it can be shown that the functions $\psi(x)$ and $\varsigma(x)$ are bounded near the path following manifold, i.e, near $\xi = 0$. It can also be shown that these are nonvanishing perturbations in the sense that $\psi(x)|_{\xi=0} \neq 0$, $\varsigma(x)|_{\xi=0} \neq 0$. Consider the result [2] drawn from [31],

[2] The Lemma presented here uses slightly different symbols.

Lemma 3.6.1 ([31]). *Let $\xi = 0$ be an exponential stable equilibrium point of the nominal system (3.34). Let $V(\xi)$, be a Lyapunov function of the nominal system that satisfies the following equations,*

$$\begin{aligned} c_1\|\xi\|^2 \leq V(\xi) \leq c_2\|\xi\|^2, \\ \frac{\partial V}{\partial \xi} A\xi \leq -c_3\|x\|^2, \\ \left\| \frac{\partial V}{\partial \xi} \right\| \leq c_4\|\xi\|, \end{aligned} \tag{3.36}$$

for all $\xi \in D$, for some positive constants c_1, c_2, c_3 and c_4, where $D = \{\xi \in \mathbb{R}^n : \|\xi\| < r\}$. Suppose the perturbation term $p(\xi)$ satisfies,

$$\|p(\xi)\| \leq \widetilde{\delta} < \frac{c_3}{c_4}\sqrt{\frac{c_1}{c_2}}\widetilde{\theta} r, \tag{3.37}$$

for all $t \geq 0$, all $\xi \in D$, some positive constant $\theta < 1$ and $\widetilde{\delta} > 0$. Then , for all $\|\xi(0)\| < \sqrt{c_1/c_2}r$, the solution $\xi(t)$ of the perturbed system (3.35) satisfies,

$$\|\xi\| \leq k \exp\left[-\widetilde{\gamma}(t)\right] \|\xi(0)\|, \quad \forall\, 0 \leq t < T \tag{3.38}$$

and

$$\|\xi(t)\| \leq b, \quad \forall t > t_0 + T \tag{3.39}$$

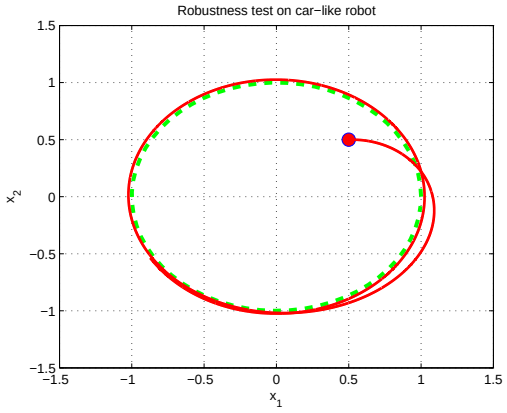
for some finite T, where

$$k = \sqrt{\frac{c_2}{c_2}}, \quad \widetilde{\gamma} = \frac{(1-\widetilde{\theta})c_3}{2c_2}, \quad b = \frac{c_4}{c_3}\sqrt{\frac{c_2}{c_1}}\frac{\widetilde{\delta}}{\widetilde{\theta}}. \tag{3.40}$$

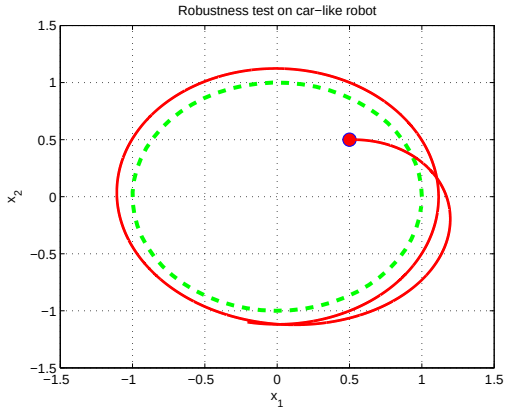
We now discuss how the above result can be applied to our problem without going in the details of the formal proof. The system (3.35), can be viewed as a perturbation of a nominal system $\dot{\xi} = A\xi$. A Lyapunov function $V(\xi) = \xi^T P \xi$ can be found where P is calculated by solving the Lyapunov equation $PA + A^T P = -I$, where I is 3×3 identity matrix. Since, $\xi = 0$ is an exponentially stable equilibrium point of the nominal system (3.34) and it can be shown that the chosen Lyapunov function satisfies the set of equations (3.36). Moreover, it can also be shown that the perturbed term $p(x)$ is bounded and satisfies (3.37). Therefore by Lemma 3.6.1, the perturbed system will stay close to the path because ξ will stay close to zero for all time.

To show the robustness of the proposed controlled we show some simulation results, see Figure 3.9. In these simulations the length of the car-like robot is assumed to be $L = 1$. We assume that we have a modeling error δ, so that the measured length is $L + \delta$. We use the word measured length to represent the real model for the car-like robot. However the word actual length is used to represent the nominal model of car-like robot.

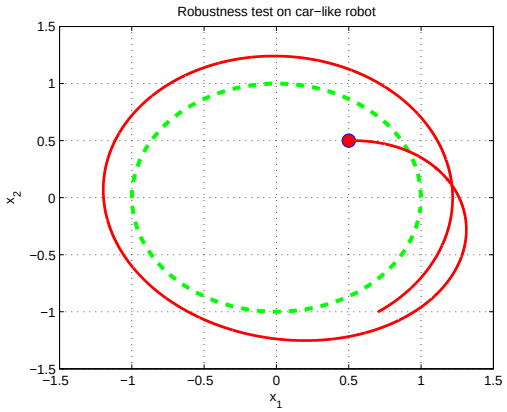
In Figure 3.9(a) we assume that there a small modeling error $\delta = 0.1$. This error is reasonably small compared to the actual length of the car-like robot which is $L = 1$. We expect our controller to overcome such small modeling error and as the results show in Figure 3.9(a) the robot follows the path with very small path following error. In Figure 3.9(b) we increase the modeling error to $\delta = 0.5$. This is a large modeling error as the measured length is half more than the length of the actual robot. However, the simulation results show that the robot follows the path but a larger path following error is observed compared to the previous case. In Figure 3.9(c) we increase the modeling error to $\delta = 1$. This is a very large modeling error as it means that the measured length is twice as much as the length of the actual robot. The simulation results show that in this case the path following error is very large but still the state of the system remains bounded. Finally we suppose an unrealistic modeling error $\delta = 50$. Figure 3.9(d) shows that the phase curves of the system are no longer bounded. It is interesting to note that, since the perturbation terms do not vanish on the path following manifold, the point $\xi = 0$ is no longer an equilibrium of the transversal dynamics. This fact means that the path following manifold is no longer controlled invariant and as a result we cannot guarantee path invariance. Nevertheless, these simulation results are encouraging and suggest that, at least for errors in the parameter ℓ, the proposed controller is fairly robust.



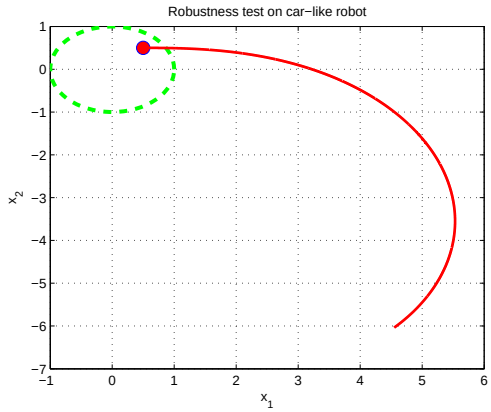
(a) Very small modeling error $\delta = 0.1$.



(b) Small modeling error $\delta = 0.5$.



(c) Large modeling error $\delta = 1$



(d) Unrealistically large modeling error $\delta = 50$.

Figure 3.9: Simulation results showing the robustness of the controller.

Chapter 4

Approximate Feedback Linearization

In this chapter, we investigate systems that fail to have a well-defined relative degree. Such systems are called non-regular systems. In [24] the authors give a procedure to deal with non-regular SISO system. In this chapter we apply the procedure to a SISO standard 1-trailer system. Later on, we apply the result to MIMO standard 1-trailer system. We fix the translational velocity of the front vehicle to some non-constant $v \neq 0$ and characterize the path following manifold $\Gamma^\star$. We try to solve the path following problem, satisfying **PF1** and **PF2** using solely the steering control input. We show that transverse feedback linearization cannot be applied to the 1-trailer system because the system does not have a well-defined relative degree. Moreover, we perform dynamic extension of the system and show that by considering both the control inputs, dynamic transverse feedback linearization cannot be applied to the 1-trailer system to solve the path following problem because the system does not have a well defined vector relative degree. We give a procedure to deal with such systems in Section 4.3.

There has been a great deal of interest in solving the path following and trajectory tracking problem of systems consisting of wheeled mobile robots pulling trailers. Examples of the works dealing with articulated vehicles or trailer systems include, [4], [5], [6], [8], [10], [21], [33], [37], [40], [47]. The trailer system has now become a testbed for analyzing several path following and trajectory generation problems in nonlinear control. In the literature two types of trailer systems are found, the standard trailer system and the general trailer system [4]. In the standard n-trailer system, each axle is hitched to the preceding trailer by means of a rigid bar. That means there is no *off axle hitching*. It was shown by Fliess et al. [21] that a standard n-trailer system is differentially flat and that the flat output is the central axle point of the last trailer. In the general trailer system each trailer is not attached directly at the middle of the preceding axle but at a positive distance from the central axle. This means the general trailer system has *off axle hitching*. The authors in [21] showed that the general 1-trailer system is also differentially flat. It was further

shown that the general n-trailer system is not differentially flat.

4.1 Model of the Standard 1-trailer System

In this chapter we consider the standard 1-trailer system modeled with a car-like robot as the lead vehicle. The kinematic model of a standard one trailer system driven by a car-like robot, shown in Figure 4.1, is

$$\dot{x} = \begin{bmatrix} \cos x_3 & 0 \\ \sin x_3 & 0 \\ \frac{1}{\ell}\tan x_4 & 0 \\ 0 & 1 \\ \frac{1}{d_1}\sin(x_3 - x_5) & 0 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{4.1}$$

where $x \in \mathbb{R}^5$ is the state, the input $v \in \mathbb{R}$ is the translational speed and $\omega \in \mathbb{R}$ is the angular velocity of the steering angle. We take the position of the last trailer in the plane as the output of (4.1),

$$y = h(x) = \begin{bmatrix} x_1 - d_1 \cos x_5 & x_2 - d_1 \sin x_5 \end{bmatrix}^\top. \tag{4.2}$$

Suppose we are given a path to follow in the output space $\mathbb{R}^2$ of (4.1). In this chapter we use a similar framework we used in Chapter 2, that is, first we design a controller to satisfy **PF1** and **PF2**, defined in Chapter 1, using transverse feedback linearization. Then we design a controller that satisfies **PF1**, **PF2** and **PF3**, defined in Chapter 1, using dynamic transverse feedback linearization. However we cannot apply the exact same procedure we applied in Chapter 2 and 3 because this system does not have a well defined relative degree. In [24] the authors outline a procedure of approximate feedback linearization to deal with such systems. Before proceeding to the case of standard 1-trailer system, we give a brief literature review of approximate feedback linearization.

4.2 Review of Approximate Feedback Linearization

In [24] the authors discuss the application of the theory of feedback linearization to the ball and beam system. For some nonlinear systems, like the ball and beam system, the conditions for feedback linearization fail but do so "slightly". The regular procedure of controller design via feedback linearization cannot be applicable. Consider the familiar

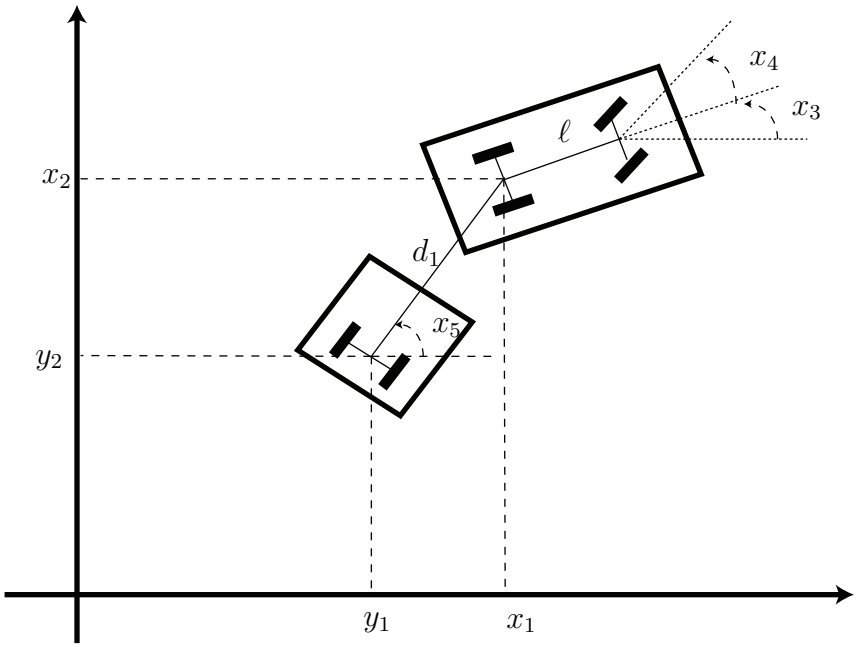


Figure 4.1: The kinematic model of standard 1-trailer system.

model of a ball and beam experiment found in many undergraduate control laboratories. The model, taken from [24], is

$$\dot{x} = \begin{bmatrix} x_2 \\ B(x_1 x_4^2 - G\sin x_3) \\ x_4 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u \tag{4.3}$$

where, $B := M/(J_b/R^2 + M)$, M and J_b are the mass and moment of inertia of the ball respectively, R is the radius of the ball and G is the acceleration due to gravity. The state variables are $x = (x_1, x_2, x_3, x_4)^\top := (r, \dot{r}, \theta, \dot{\theta})^\top$ where θ is the beam angle and r is the ball position [24]. Take the beam angle as the output $y = h(x) = x_3$ and define

$$f(x) := \begin{bmatrix} x_2 \\ B(x_1 x_4^2 - G\sin x_3) \\ x_4 \\ 0 \end{bmatrix}, g(x) := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

so that the ball and the beam system (4.3) can be written as

$$\begin{aligned}\dot{x} &= f(x) + g(x)u \\ y &= h(x).\end{aligned}$$

Following the procedure from Chapter 2, but, instead of differentiation a curve defined in the output space until the input appears, we differentiation the output $y(x)$ directly until the input appears [51],

$$\begin{aligned}y &= h(x) = x_1 \\ \dot{y} &= L_f h(x) = x_2 \\ \ddot{y} &= L_f^2 h(x) = Bx_1 x_4^2 - BG \sin x_3 \\ y^{(3)} &= L_f^3 h(x) + L_g L_f^2 h(x) u = \left[Bx_2 x_4^2 - BGx_4 \cos x_3\right] + \left[2Bx_1 x_4\right] u.\end{aligned}$$

If the term $L_g L_f^2 h(x)$ is non-zero at a point of interest, a control law of the form $u = (-L_f^3 h(x) + v)/L_g L_f^2 h(x)$ can be used as in Chapter 2. Unfortunately, the control term $L_g L_f^2 h(x)$ is zero whenever the beam angular velocity or ball position is zero. Therefore the relative degree of the system is not well defined near $x = 0$. Thus input-output linearization is not applicable to this problem at $x = 0$. The authors in [24] propose an approximation procedure that is different from the Jacobian linearization. The system fails to have a well defined relative degree because of the term $L_g L_f^2 h(x) = 2Bx_1 x_4$. An approximation is made by assuming that the term $2Bx_1 x_4$ is identically equal to zero. By neglecting the term $2Bx_1 x_4$ an approximate system is obtained with a well defined relative degree. The simulation results in [24] show that controllers designed using the approximate system provide a good tracking on the original system. In [24, Theorem 4.4] it is shown that the tracking error will remain bounded.

A single inverted pendulum is another example of a control system which fails to have a well-defined relative degree at points in the state space which are of interest, the two equilibrium points [3]. A car-like robot connected with a trailer is another example of a system that fails to have a well-defined relative degree. We investigate the trailer system in the next section. A mathematical argument can be made based on Definition 4.2, 4.3 and Theorem 4.4 in [24] that explains why the control scheme works for the case of a trailer system but due to the large expressions involved we do not perform it in this thesis.

4.3 Approximate Transverse Feedback Linearization

In [24] the discussion was restricted to approximate feedback linearization. We would like to apply this idea to partially feedback linearizable systems. We apply the approximate

transverse feedback linearization and approximate dynamic transverse feedback linearization to a 1-trailer system. We give a simulation based argument. We first design a path following controller based on these approximations and then simulate the closed-loop system using the original, non-approximated, system. Using these simulations, we argue that our approximation appears to be valid.

Consider (4.1) and a circular path (2.3). The lift of γ to $\mathbb{R}^5$ is given by

$$\Gamma := (s \circ h)^{-1}(0) = \left\{ x \in \mathbb{R}^5 : s(h(x)) = 0 \right\},$$

is a two dimensional submanifold. Define

$$\alpha(x) := s \circ h(x) = (x_1 - d_1 \cos x_5)^2 + (x_2 - d_1 \sin x_5)^2 - 1. \tag{4.4}$$

We know from Chapter 2 that making $x \to \Gamma$ is equivalent to making $y \to \gamma$. Since we also know from Chapter 2 and 3 that **PF1** and **PF2** can be satisfied by fixing the speed of the robot to some non zero constant. Fix the speed of the trailer system $v = \mathrm{v} \neq 0$. Let $u := \omega$. Under these conventions the system (4.1) can be written as

$$\dot{x} = \begin{bmatrix} \mathrm{v} \cos x_3 \\ \mathrm{v} \sin x_3 \\ \frac{1}{\ell} \tan x_4 \\ 0 \\ \frac{\mathrm{v}}{d_1} \sin(x_3 - x_5) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} u. \tag{4.5}$$

Define

$$f(x) := \begin{bmatrix} \mathrm{v} \cos x_3 \\ \mathrm{v} \sin x_3 \\ \frac{1}{\ell} \tan x_4 \\ 0 \\ \frac{\mathrm{v}}{d_1} \sin(x_3 - x_5) \end{bmatrix}, g(x) := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix},$$

so (4.5) can be written as compactly as $\dot{x} = f(x) + g(x)u$. Following the usual procedure we differentiate the function $\alpha(x)$ until the input u appears

$$\begin{aligned} \dot{\alpha} &= \frac{\partial \alpha(x)}{\partial x} \dot{x} \\ &= \frac{\partial \alpha(x)}{\partial x} f(x) + \frac{\partial \alpha(x)}{\partial x} g(x) u \\ &= L_f \alpha(x) + L_g \alpha(x) u, \end{aligned}$$

where,

$$L_f\alpha(x) = 2\mathrm{v}\left(\frac{x_1\cos x_3}{2} + \frac{x_2\sin x_3}{2} - d_1\cos(x_3 - x_5) + \frac{x_1\cos(x_3 - 2x_5)}{2} + \frac{x_2\cos(x_3 - 2x_5)}{2}\right),$$
$$L_g\alpha(x) = 0.$$

Since no control input appears, we take the derivative again,

$$\ddot{\alpha} = L_f^2\alpha(x) + L_gL_f\alpha(x)u,$$

where,

$$L_gL_f\alpha(x) = 0.$$

and $L_f^2\alpha(x)$ can be computed easily using any Maple or Matlab using symbolic math toolbox.[1] We take the derivative again,

$$\alpha^{(3)} = L_f^3\alpha(x) + L_gL_f^2\alpha(x)u, \tag{4.6}$$

where,

$$L_gL_f^2\alpha(x) = \frac{1}{\ell}\left(\mathrm{v}\cos x_3(2x_2 - 2d_1\sin x_5 - \mathrm{v}\sin x_3(2x_1 - 2d_1\cos x_5))\right) + \left(\frac{\mathrm{v}\cos(x_3 - x_5)(2d_1\sin x_5)(x_1 - d_1\cos x_5) - 2d_1\cos x_5(x_2 - d_1\sin x_5)}{\ell d_1}\right).$$

and $L_f^3\alpha(x)$ can be easily computed using a computer program. If the coefficient multiplying the input u, $L_gL_f^2\alpha(x)$, is nonzero at a point on the set

$$\left\{x \in \mathbb{R}^5 : \alpha(x) = L_f\alpha(x) = L_f^2\alpha(x) = 0\right\} \tag{4.7}$$

then $\alpha(x)$ yields a well-defined relative degree. In that case we can use a control law of the from,

$$u = \frac{1}{L_gL_f^2\alpha}(-L_f^3\alpha + v^{\text{th}}), \tag{4.8}$$

to stabilize the set (4.7) and make the path attractive and invariant. Unfortunately, the coefficient $L_gL_f^2\alpha(x)$ is zero whenever $x_3 = x_5 = 0$. From the model of the standard 1-trailer system (4.1), we know that x_3 represents the orientation of the car in the 1-trailer system and x_5 represents the position of the trailer in the 1-trailer system shown in

[1] A Matlab program is included in Appendix B that computes these expression.

Figure 4.1. The 1-trailer system can have any orientation of car and the attached trailer on the path. That means the orientation $x_3 = x_5 = 0$ can occur on the given path. Therefore the relative degree of the system is not well defined. We turn to the approximate feedback linearization approach to control the system. Following [24] we assume that $L_g L_f^2 \alpha(x)$ is identically equal to zero and continue taking derivatives of (4.6)

$$\alpha^{(4)} \approx L_f^4 \alpha(x) + L_g L_f^3 \alpha(x) u,$$

where $L_f^4 \alpha(x) \neq 0$ and $L_g L_f^3 \alpha(x) \neq 0$ can be computed using Matlab. With this approximation the path following manifold $\Gamma^\star$ for the trailer system following a circular path is approximately equal to

$$\Gamma^\star \approx \left\{ x \in \mathbb{R}^5 : \alpha(x) = L_f \alpha(x) = L_f^2 \alpha(x) = L_f^3 \alpha(x) = 0 \right\}.$$

Note that, since,

$$\left\{ x \in \mathbb{R}^5 : \alpha(x) = L_f \alpha(x) = L_f^2 \alpha(x) = L_f^3 \alpha(x) = 0 \right\} \subset \Gamma^\star$$

if we stabilize the approximation of $\Gamma^\star$ we will cause the system output to approach the desired path.

The four functions that approximate $\Gamma^\star$ partially define a local coordinate transformation. As in Chapter 2 we use the angle of the output with respect to the origin to complete the coordinate transformation

$$\pi(x) := \tan^{-1}\left(\frac{x_2 - d_1 \sin x_5}{x_1 - d_1 \cos x_5} \right). \tag{4.9}$$

Now consider the coordinate transformation

$$\begin{bmatrix} \eta_1 \\ \xi_1 \\ \xi_2 \\ \xi_3 \\ \xi_4 \end{bmatrix} = T(x) = \begin{bmatrix} \pi(x) \\ \alpha(x) \\ L_f \alpha(x) \\ L_f^2 \alpha(x) \\ L_f^3 \alpha(x) \end{bmatrix}. \tag{4.10}$$

In (η, ξ)-coordinates the system is modelled as

$$\begin{aligned} \dot{\eta}_1 &= f(\eta, \xi, u) \\ \dot{\xi}_1 &= \xi_2 \\ \dot{\xi}_2 &= \xi_3 \\ \dot{\xi}_3 &= \xi_4 + \phi(\eta, \xi) u \\ \dot{\xi}_4 &= L_f^4 \alpha + L_g L_f^3 \alpha u \big|_{x = T^{-1}(\eta, \xi)} \end{aligned} \tag{4.11}$$

where we neglect the term $\phi(\eta,\xi) = L_g L_f^2 \alpha(x)\big|_{x=T^{-1}(\eta,\xi)}$. It in interesting to note the in [24] and [3] an exact approximate feedback linearization is performed, however, using the similar procedure we performed partial approximate feedback linearization.

When $\xi = 0$ the system output converges to the path. Just like previous chapters we call the ξ-subsystem the transversal subsystem and the states ξ the transversal states. Consider the regular feedback transformation

$$u = \frac{1}{L_g L_f^3 \alpha}(-L_f^4 \alpha + v^{\pitchfork}), \tag{4.12}$$

where $v^{\pitchfork}$ is auxiliary control inputs. Since $L_g L_f^3 \alpha(x^\star) \neq 0$ therefore this controller is well-defined in a neighbourhood of $x^\star$. Thus in a neighbourhood of $x^\star$ the closed-loop system becomes

$$\begin{aligned}
\dot{\eta}_1 &= f(\eta,\xi,u)\\
\dot{\xi}_1 &= \xi_2\\
\dot{\xi}_2 &= \xi_3\\
\dot{\xi}_3 &= \xi_4 + \phi(\eta,\xi)\ \frac{1}{L_g L_f^3 \alpha}(-L_f^4 \alpha + v^{\pitchfork})\Big|_{x=T^{-1}(\eta,\xi)}\\
\dot{\xi}_4 &= v^{\pitchfork}.
\end{aligned} \tag{4.13}$$

We design a control law similar to the previous chapters to stabilize the transversal subsystem.

$$v^{\pitchfork}(\xi) = k_1\xi_1 + k_2\xi_2 + k_3\xi_3 + k_4\xi_4, \tag{4.14}$$

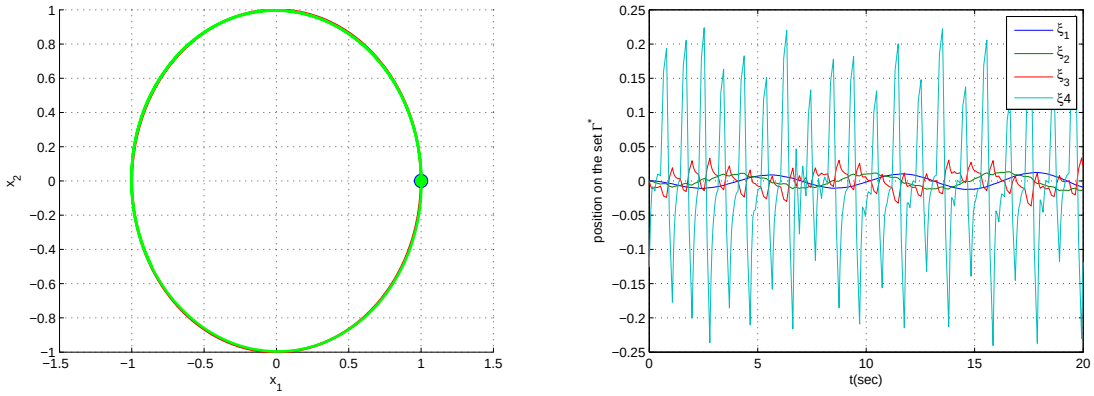
with $k_i < 0$, $i \in \{1,2,3,4\}$. We now show by way of simulation that this approach works.

4.3.1 Simulation Results

In the first simulation, we initialize the trailer system (4.5) at

$$x_{01} = (1.00, 0.10, 1.67, 0.10, 1.57),$$

i.e., sufficiently close to the curve see Figure 4.2. It can be seen in Figure 4.2(a) that the 1-trailer system follows the curve. However, it is interesting to analyze the behavior of the ξ-states. It can be seen from Figure 4.2(b) that although all the ξ-states are not converging to zero, unlike the case of unicycle and car-like robot shown in previous chapter, the ξ-states are not blowing up. Figure 4.2 show that our approximation is a valid approximation and we get bounded path following error if we initialize sufficiently closer to the curve.



(a) Trailer system (4.5) following circular curve.

(b) ξ-states

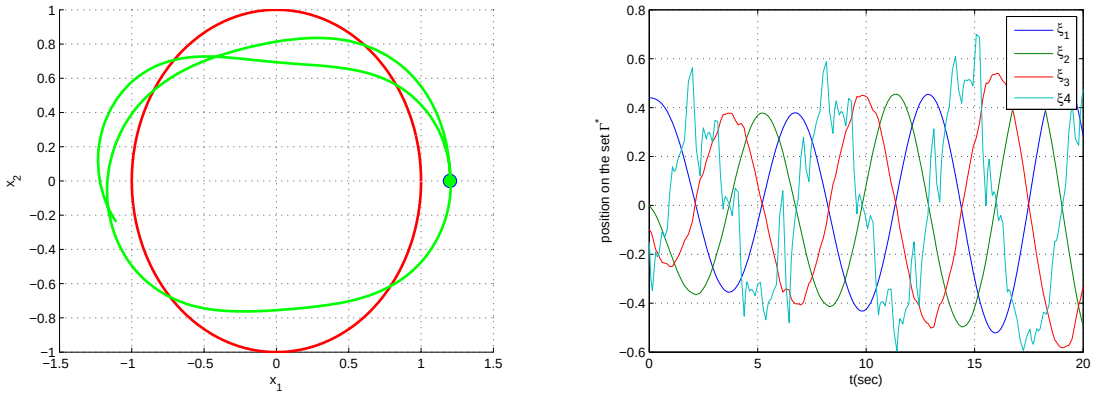
Figure 4.2: 1-trailer system initialized at x_{01}.

In the second simulation Figure 4.3, we initialize the trailer system (4.5) at

$$x_{02} = (1.20, 0.10, 1.67, 0.10, 1.57),$$

i.e., slightly away from the curve. Figure 4.3(a) shows that the path following error is quite large. This is because the neglected term $L_g L_f^2 \alpha(x)$ is zero when the system is on the curve. The term is non-zero in a neighborhood of the curve and since we have neglected the term, the error is large. Also the term is nonlinear so the error increases in nonlinear fashion, i.e., a small changes in the initial conditions result in huge path following error. It can be seen from Figure 4.3(b) that error in the ξ-state are significantly larger compared to the previous simulation result. Since all the ξ-states are bounded, therefore, we managed to follow the path but large error in ξ-states are the cause of large path following error.

It is shown by authors in [57], for non-regular system, better performance can be achieved by switching trough singularities. In the case of fire truck system, following the approach proposed in [57] two control laws can be designed; approximate control law and exact control law. Approximate control law can be used in the neighborhood of the desired path because the system loses relative degree. When the trailer system is away from the path the system has a well defined relative degree therefore the exact control law can be used. In [57] the authors propose a switching law that allows to switch between the approximate and exact controller. Switching through singularities is not studied in this thesis, however, it can be an interesting area to explore in future for such systems.



(a) Trailer system (4.5) following circular curve.

(b) ξ-states

Figure 4.3: 1-trailer system initialized at x_{02}.

4.4 Approximate Dynamic Transverse Feedback Linearization

From the discussion of previous chapters, we know that in order to satisfy **PF3** we need to consider both the inputs of the system. Moreover, we need to increase the dimension of path following manifold. To achieve that we perform dynamic extension of the 1-trailer sytem (4.1) by adopting a similar procedure to that discussed in Chapter 2 and Chapter 3. The system after dynamic extension takes the form,

$$\begin{aligned}\dot{x} &= f(x) + g_1(x)u_1 + g_2(x)u_2 \\ &= \begin{bmatrix} (\mathrm{v} + x_6)\cos x_3 \\ (\mathrm{v} + x_6)\sin x_3 \\ \frac{(\mathrm{v}+x_6)}{\ell}\tan x_4 \\ 0 \\ \frac{\mathrm{v}+x_6}{d_1}\sin(x_3 - x_5) \\ x_7 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} u_2 \end{aligned} \tag{4.15}$$

Our objective is to now design the control law $u = (u_1, u_2)$ to solve the the path following problem. The controller design procedure for this system is different from the car-like robot and unicycle because this system is a non-regular control system. We apply the procedure outlined in [24] to our extended MIMO system. Again due to huge and complicated expression of the decoupling matrix we argue on the basis of numerical graphs of the

entries of the decoupling matrix. Consider the virtual output function (4.16) defined in a similar way in Chapters 2, 3.

$$\hat{y} = \begin{bmatrix} \pi(x) \\ \alpha(x) \end{bmatrix}. \tag{4.16}$$

where, $\pi(x)$ and $\alpha(x)$ is defined in (4.9) and (4.4) respectively. Following the procedure used in the previous chapters we differentiate the virtual output until the inputs appears. A Matlab program is written to symbolically differentiate the output. The program is included in the appendix. By the definition of vector relative degree we must show that

$$L_{g_1} L_f^i \pi(x) = L_{g_2} L_f^i \pi(x) = L_{g_1} L_f^i \alpha(x) = L_{g_2} L_f^i \alpha(x) = 0 \tag{4.17}$$

for $i \in \{0, 1\}$ in a neighborhood of the path and that the decoupling matrix

$$D(x^\star) = \begin{bmatrix} L_{g_1} L_f^2 \pi(x) & L_{g_2} L_f^2 \pi(x) \\ L_{g_1} L_f^2 \alpha(x) & L_{g_2} L_f^2 \alpha(x) \end{bmatrix} \tag{4.18}$$

is non-singular. It can be seen from the Matlab program that the terms in (4.17) are all zero. The graphs of the entries of the decoupling matrix are shown in Figure 4.4. As shown in Figure 4.4(a) and 4.4(b), $L_{g_1} L_f^2 \pi(x)$,$L_{g_2} L_f^2 \pi(x)$ never goes to zero in a neighborhood of the curve. However it in interesting to note from Figure 4.4(c) and 4.4(d) that $L_{g_1} L_f^2 \alpha(x)$ and $L_{g_2} L_f^2 \alpha(x)$ are zero on the curve. Therefore the decoupling matrix (4.18) loses rank on the curve. Hence the system does not have a well-defined vector relative degree.

To control the system we need to approximate the system vector field. Similar to the non-regular single input case we neglect the terms, $L_{g_1} L_f^2 \alpha(x)$ and $L_{g_2} L_f^2 \alpha(x)$. Since by neglecting the terms $L_{g_1} L_f^2 \alpha(x)$ and $L_{g_2} L_f^2 \alpha(x)$ the control inputs vanish, we need to take derivative again. The decoupling matrix now becomes,

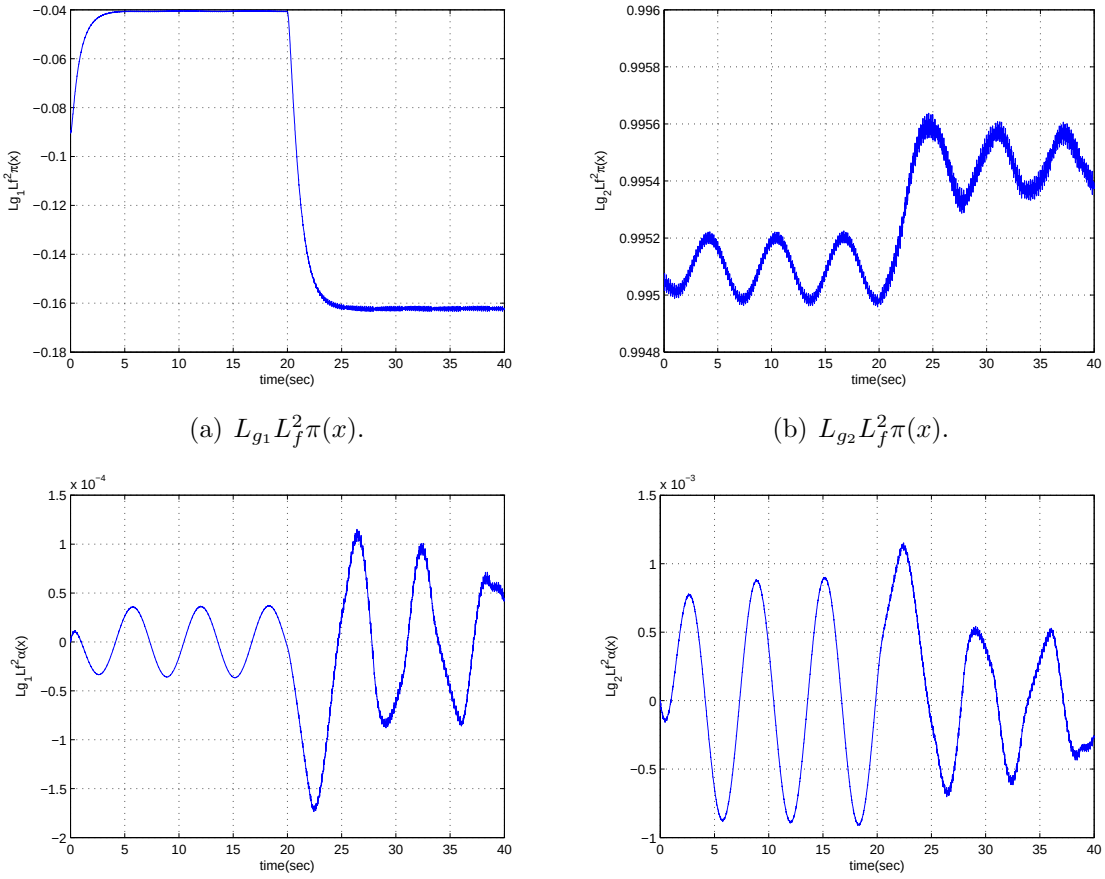
$$D(x^\star) = \begin{bmatrix} L_{g_1} L_f^2 \pi(x) & L_{g_2} L_f^2 \pi(x) \\ L_{g_1} L_f^3 \alpha(x) & L_{g_2} L_f^3 \alpha(x) \end{bmatrix} \tag{4.19}$$

The terms $L_{g_1} L_f^3 \alpha(x)$ and $L_{g_2} L_f^3 \alpha(x)$ are not equal to zero at the point of interest, so we expect the decoupling matrix to be full rank. Figure 4.5 shows that the determinant of the decoupling matrix is bounded away from zero. With this approximation we can design transversal and tangential controllers as in Chapter 3.

4.4.1 Simulation Results

In this section we want to demonstrate that by approximate partial feedback linearization we can satisfy **PF1, PF2** and **PF3**. We desire to following the velocity profile

$$\eta_2^{ref} = \begin{cases} -0.2 & 0 \leq t < 20s \\ -0.4 & t \geq 20s. \end{cases}$$



(a) $L_{g_1}L_f^2\pi(x)$. (b) $L_{g_2}L_f^2\pi(x)$.

(c) $L_{g_1}L_f^2\alpha(x)$. (d) $L_{g_2}L_f^2\alpha(x)$.

Figure 4.4: Entries of decoupling matrix (4.18).

In the first simulation, Figure 4.6, we initialize the trailer system (4.15) at

$$x_{01} = (1.000, 0.1000, 1.6705, 0.1042, 1.5708, 0.1000, 0.0000).$$

As we initialize the system very close to the curve the approximation gives very small path following error as shown in Figure 4.6(a). It is also interesting to note that the system closely tracks the desired velocity profile as shown in figure 4.6(b).

Now we initialize the system slightly away from the path,

$$x_{02} = (1.100, 0.1000, 1.6705, 0.1042, 1.5708, 0.1000, 0.0000),$$

and track the same velocity profile. In this case, sine we initialize the system slightly away from the curve, the approximation gives relatively large path following error as shown in

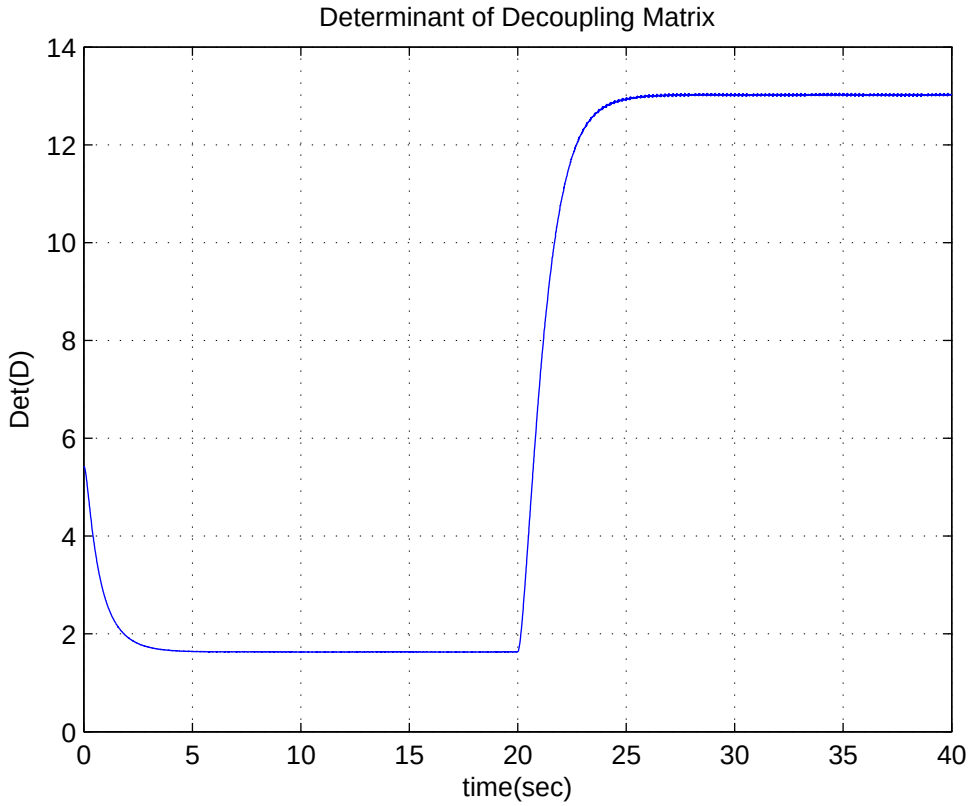
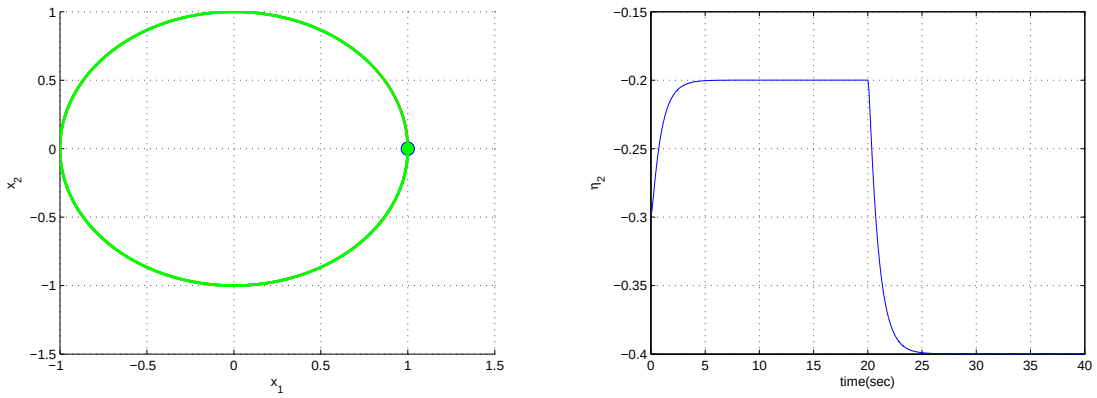


Figure 4.5: Determinant of decoupling matrix (4.18).

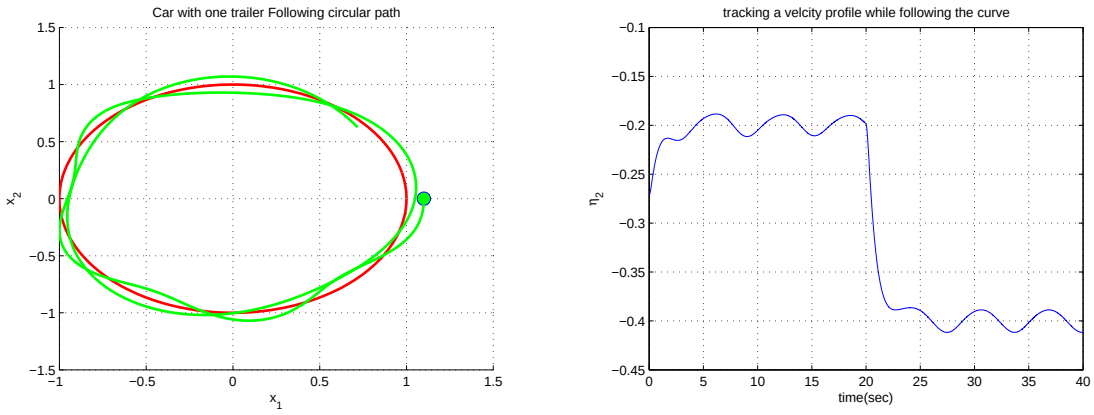


(a) Trailer system (4.15) following circular curve.

(b) velocity of trailer system

Figure 4.6: 1-trailer system (4.15) initialized at x_{01}.

Figure 4.7(a). However, it is important to note that the system still follows the path. The approximation also effects the results of the velocity tracking error. As shown in Figure 4.7(b), the velocity error is significantly higher compared to the case when the trailer system is initialized close to the curve.



(a) Trailer system (4.15) following circular curve.

(b) velocity of trailer system

Figure 4.7: 1-trailer system initialized at x_{02}.

4.5 Conclusion

In this chapter an application of the theory of feedback linearization to non-regular systems was presented. Non-regular systems are systems that fail to have a well defined relative degree. It was shown that the theory developed in Chapter 3 and Chapter 2 sometimes need tweaks in order to apply to non-regular systems. It was shown using an example of the standard 1-trailer system that for systems that fail to have a well defined relative degree an approximate relative degree is achieved by approximating the system's vector field. Due to complicated expression involved in the analysis it was hard to give a mathematical argument that the theory works, however, it was shown through simulations that the approximation technique can be used where sufficiently large path following errors are permissible.

Chapter 5

Curve Approximation

The control design technique discussed in this thesis relies on having both a parametric and an implicit representation of the path to be followed. Given a smooth parametric representation of a curve in $\mathbb{R}^2$, in this chapter we provide a procedure for finding an implicit representation of the image of the curve. The proposed solution involves two steps. First, we approximate the given parametric curve as a rational parametric curve using the Weierstrass approximation theorem. Second, relying on elimination theory, we represent the image of the rational approximation of the curve as an implicit function.

5.1 Introduction

There are two basic representations for planar curves; parametric and implicit. In a parametric representation, points on the curve or surface are given as functions of a parameters. In an implicit representation, also called a zero level set representation, the curve or surface is defined to be the set of points that satisfy one or more equations. Implicitization is the process of converting a parametrically defined curve into implicit form. The implicitization of parametric curves relies on the theory of elimination. The general theory of elimination is a somewhat lost art and very few references are available. The interested readers are referred to [48],[49]. It is widely accepted that the parametric representation is best suited for generating points along the curve while a zero level set representation is helpful for determining whether a point lies on the curve or not [52].

Most general way of representing a curve is to express it in parametric form. In the case of a unit circle both parametric, $(\cos(\lambda), \sin(\lambda))$, and implicit form, $x_1^2 + x_2^2 - 1 = 0$, are well known. However for a general parametric curves, a solution to the implicitization problem is not known. From Chapters 2, 3 and 4 we know that in order to solve the path following problem, an implicit or zero level set representation of the parametric curve is

required. According the procedure we followed in Chapter 2, 3, 4 the first transversal state is defined as the implicit form of the curve, i.e., $\xi_1 = \alpha(x)$, where $\alpha(x)$, represents the lift of the implicit form of the curve. For the case of the unicycle and the circle the implicit representation is $\alpha(x) = x_1^2 + x_2^2 - 1 = 0$. We have assumed in previous chapters that we know the implicit representation, $\alpha(x)$, of the given parametric curve. However, the lack of a general procedure to convert parametric curves to implicit form prevented their use in many practical applications. We present a method to convert a general parameterized curve to a zero level set. The method is based on the Weierstrass approximation theorem and implicitization by Sylvester matrix.

5.2 General Problem

Suppose that a curve is given as a continuously differentiable parameterized path

$$\begin{aligned} \sigma : \mathbb{D} &\to \mathbb{R}^2 \\ \lambda &\mapsto \begin{bmatrix} \sigma_1(\lambda) \\ \sigma_2(\lambda) \end{bmatrix} \end{aligned} \tag{5.1}$$

where $\mathbb{D}$ equals either $\mathbb{S}$ if the curve is closed or $\mathbb{R}$ if the curve is non-closed.

We want to find an implicit representation of the image of σ. Thus we are interested in solving the following problem.

Problem 1. *Given a curve* (5.1) *find, if possible, a continuously differentiable function* $s : \mathbb{R}^2 \to \mathbb{R}$ *such that*

$$\sigma(\mathbb{D}) = \{y \in \mathbb{R}^p : s(y) = 0\}.$$

5.3 Special Cases of the General Problem

Consider the special case when the given parameterized curve (5.1) has domain $\mathbb{R}$ and $\sigma_2(\lambda) = \lambda$, i.e.,

$$\begin{aligned} \sigma : \mathbb{R} &\to \mathbb{R}^2 \\ \lambda &\mapsto \begin{bmatrix} \sigma_1(\lambda) \\ \lambda \end{bmatrix}. \end{aligned} \tag{5.2}$$

Alternatively, suppose that $\sigma_1(\lambda) = \lambda$, i.e.,

$$\begin{aligned} \sigma : \mathbb{R} &\to \mathbb{R}^2 \\ \lambda &\mapsto \begin{bmatrix} \lambda \\ \sigma_2(\lambda) \end{bmatrix}. \end{aligned} \tag{5.3}$$

We first find implicit representations for these two special cases because the analysis will suggest a method to apply to the general case. For these special cases it easy to see that the implicitization is given by

$$s(y) = y_2 - \sigma_2(y_1) = 0$$

or

$$s(y) = y_1 - \sigma_1(y_2) = 0.$$

However, this approach does not generalize to arbitrary planar curves and so we present an alternative approximation approach suitable for generalization. The special cases (5.2) and (5.3) are similar and solving one immediately provides a solution to the other. Thus we will focus on special case (5.2) from now on.

5.3.1 Proposed Solution

In order to find an implicit representation of the curve (5.2), we first approximate $\sigma_1(\lambda)$ as a polynomial using Theorem A.3.5. In the first instance of this problem suppose that we are given a *fixed* partition of an interval of the real line.

Problem 2. *Let the parameterized curve* (5.2)*, a real number* $\epsilon > 0$ *and a finite set of real numbers* $k_0 < k_1 < \ldots, k_q < k_{q+1}$ *be given. Let* $I_i := [k_{i-1}, k_i]$*,* $i \in \{1, \ldots, q+1\}$*. Find, if possible,* $q+1$ *polynomials* $p_i(\lambda) : I_i \longrightarrow \mathbb{R}$*,* $i \in \{1, \ldots, q+1\}$ *such that*

$$\max_{\lambda \in I_i} \|\sigma_1(\lambda) - p_i(\lambda)\| < \epsilon. \tag{5.4}$$

Our solution to Problem 2 is based on the constructive proof of Theorem A.3.5 presented in [18]. In that proof, given a continuous function $f : [0,1] \longrightarrow \mathbb{R}$, a Bernstein polynomial B_n^f defined in (A.11) is shown to converge uniformly to f as the order of B_n^f gets sufficiently large. Specifically, for any $\epsilon > 0$, there exists an integer $N > 0$ such that

$$(\forall n \geq N) \max_{\lambda \in [0,1]} \|f(\lambda) - B_n^f(\lambda)\| < \epsilon. \tag{5.5}$$

The order N of B_n^f for which (5.5) holds is lower bounded by

$$N \geq \frac{M}{\delta^2 \epsilon} \tag{5.6}$$

where $M = \|f\|_\infty$ and δ comes from the definition of uniform continuity, Definition A.3.1. Note that by Theorem A.3.2, the function f is uniformly continuous on $[0,1]$. Practically, finding δ for arbitrary functions is not easy. This makes determining an *a priori* estimate

for the integer N difficult. Thus, although we do not know *a priori* the order of B_n^f for which we get a good approximation, Theorem A.3.5 tells us that if we increase the order of B_n^f we will eventually get a good estimate.

In the case of Problem 2, take the interval I_i, $i \in \{1, \ldots, q+1\}$. On this interval, define the function $\tau_i : [0,1] \longrightarrow I_i$ as

$$\tau_i(\lambda) = k_{i-1} + \lambda\left(k_i - k_{i-1}\right).$$

This function is a homeomorphism between $[0,1]$ and I_i. We use it to define

$$f_i(\lambda) := \sigma_1 \circ \tau_i(\lambda). \tag{5.7}$$

Using the Weierstrass Approximation Theorem and Bernstein polynomials for the function (5.7), we have that for any $\epsilon > 0$, there exists an integer $N > 0$ such that (5.5) holds.

Lemma 5.3.1. *There exists a positive finite integer N such that Problem 2 is solved by any*

$$p_i(\lambda) = B_n^{f_i}(\tau_i^{-1}(\lambda)), \;\; n \geq N, i = 1, \ldots, q+1.$$

Proof. Fix $i \in \{1, \ldots, q+1\}$. In (5.6) we know only ϵ, which is the given, finite, error tolerance. We need to know M and δ in order to find N. We will find N_i on each interval I_i and then take N as the largest of the N_i.

Define $M_i = \sup_{\lambda \in [k_i, k_{i+1}]} \parallel \sigma_1(\lambda) \parallel - \inf_{\lambda \in [k_i, k_{i+1}]} \parallel \sigma_1(\lambda) \parallel$. Clearly this $M_i \geq \|f_i\|_\infty$. Since the interval $[k_i, k_{i+1}]$ is compact and $\sigma_1(\lambda)$ is continuous on I_i then by Theorem (A.3.2), $\sigma_1(\lambda)$ is uniformly continuous.

By the definition of uniform continuity, (A.3.1), there exists some $\delta > 0$. Thus the qualities M_i, ϵ and δ_i are all finite. Let N_i be

$$N_i = \frac{M_i}{\delta_i^2 \epsilon}.$$

Repeat this construction for each $i \in \{1, \ldots, q+1\}$ and let

$$N = \max\{\{N_1, \ldots, N_{q+1}\}\}.$$

□

The above proof relies on finding the numbers δ_i which is difficult in practice. Instead of directly finding these δ_i we have proposed an algorithm that is independent of them. The algorithm keeps on increasing the order of the polynomial and iteratively checking the error. Hence Lemma (5.3.1) and Algorithm2, given below, solves the given problem.

input : $\sigma : \mathbb{R} \to \mathbb{R}^2$
$\epsilon > 0$
$N = 1$; (start with the smallest possible order)
$I_i = [k_{i-1}, k_i]$
output: $P_i(\lambda)$

for *each* I_i **do**
  while $error > \epsilon$ **do**
    for $k = 0 : N$ **do**
      compute: $c_k := \sigma_1(k/N)$
      compute: $p_i(\lambda) = c_k \binom{N}{k} \lambda^k (1-\lambda)^{N-k}$
    end
    calculate error: $\max_{\lambda \in I_i} \|\sigma_1(\lambda) - p_i(\lambda)\|$
    N=N+1
  end
end

Algorithm 2: Curve Approximation

Implicitization

The special structure of the curve (5.2) suggests a rather easy way of implicitization. Using the results of Lemma (5.3.1), curve (5.2) is represented in implicit form,

$$\gamma_i = \{(y_1, y_2) \in \mathbb{R}^2 \ : \ y_1 - p_i(y_2) = 0\}, \ \ i = 1, \ldots, q+1.$$

Hence the curve (5.2) is first approximated with polynomials and then converted to implicit form. Each implicit curve γ_i covers a piece of the path.

Simulation Results

In this section we apply results of Lemma 5.3.1 and Algorithm 2 to few curves.

Simulation I In this simulation we consider a sinusoidal curve. The parametric representation of the curve is given by,

$$\sigma : \lambda \mapsto \begin{bmatrix} \sin(\lambda) \\ \lambda \end{bmatrix}. \tag{5.8}$$

The curve (5.8) is defined on the interval $[0, 4\pi]$. Furthermore, the interval $[0, 4\pi]$ and is divided into 2 equal partitions of length 2π. Given an error tolerance ϵ, we want to approximate the curve (5.8) with a sequence of polynomials $p_i(\lambda)$, such that $\max_{\lambda \in I_i} \|\sigma_1(\lambda) -$

$p_i(\lambda)\| < \epsilon$. In this case the given error tolerance is $\epsilon = 0.25$. Figure 5.1 represents the parameterized curve (5.8) approximated with a sequence of polynomials $p_i(\lambda)$. The original curve is represented by a dashed line while the approximated curve is represented by a solid line in Figure 5.1. Solid dots in the Figure 5.1 represent the partition points. The simulation results are represented in Table 5.1.

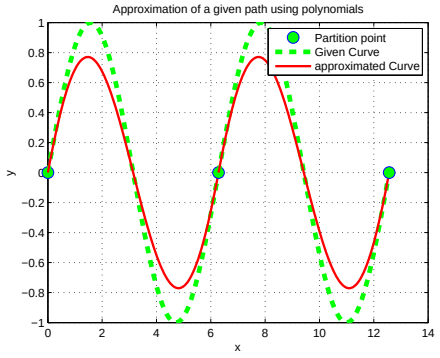


Figure 5.1: Approximation of curve (5.8) with $\epsilon = 0.25$.

i	Partition	$\| \sigma_1(\lambda) - p_i(\lambda) \|_\infty$	N
1	$[0, 2\pi]$	0.2416	14
2	$[2\pi, 4\pi]$	0.2416	14

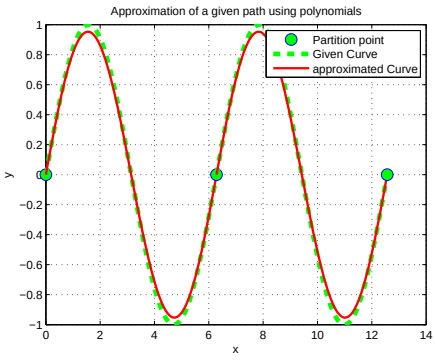
Table 5.1: Approximation results of curve (5.8) with $\epsilon = 0.25$.

Simulation II As it can be observed from the last simulation result that the approximation is not a good one because the the error $\| \sigma_1(\lambda) - p_i(\lambda) \|_\infty$ is quite large. In this simulation we desire to reduce the error so we decrease the error tolerance. Given the curve (5.8) we want to approximate it such that the $\| \sigma_1(\lambda) - p_i(\lambda) \|_\infty < 0.05$. Similar to the previous simulation the curve is defined on the interval $[0, 4\pi]$ and the interval $[0, 4\pi]$ is divided into 2 equal partitions of length 2π. As seen from Figure 5.2 the approximation is better compared to the previous case. The reason for the better approximation is that a strict error tolerance $\epsilon = 0.05$ in this case. However, it is interesting to observe from Table 5.2 that the order of the polynomial has increased significantly compared to the previous simulation. A conclusion can be drawn that the higher the order of the polynomial the better would be the approximation.

Simulation III In this simulation we show how non-smooth curves can be approximated using algorithm 2. Consider a non-smooth curve. The parametric representation of the curve is given by,

$$\sigma : \lambda \mapsto \begin{bmatrix} |\lambda| \\ \lambda \end{bmatrix} \tag{5.9}$$

The curve (5.9) is defined on the interval $[0, 4\pi]$. Furthermore, the interval $[0, 4\pi]$ is divided into 2 equal partitions of length 2π. Given an error tolerance ϵ, we want to approximate

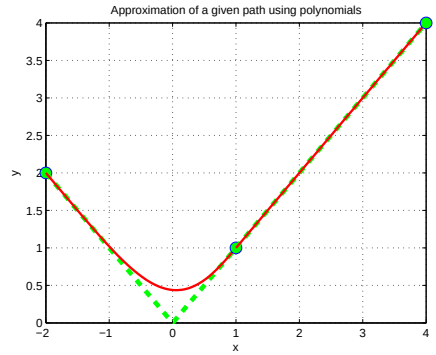


i	Partition	$\| \sigma_1(\lambda) - p_i(\lambda) \|_\infty$	N
1	$[0, 2\pi]$	0.0497	77
2	$[2\pi, 4\pi]$	0.0497	77

Table 5.2: Approximation results of curve (5.8) with $\epsilon = 0.05$.

Figure 5.2: Approximation of curve (5.8) with $\epsilon = 0.05$.

the curve (5.9) with a sequence of polynomials $p_i(\lambda)$, such that $\max_{\lambda \in I_i} \|\sigma_1(\lambda) - p_i(\lambda)\| < \epsilon$. In this case the given error tolerance is $\epsilon = 0.45$. Figure 5.3 represents the parameterized curve (5.9) approximated with a sequence of polynomials $p_i(\lambda)$. The original curve is represented by a dashed line while the approximated curve is represented by a solid line in Figure 5.1. Solid dots in the Figure 5.3 represent the partition points. The simulation



i	Partition	$\| \sigma_1(\lambda) - p_i(\lambda) \|_\infty$	N
1	$[0, 2\pi]$	0.4282	6
2	$[2\pi, 4\pi]$	0.00	1

Table 5.3: Approximation results of curve (5.9) with $\epsilon = 0.45$.

Figure 5.3: Approximation of curve (5.9) with $\epsilon = 0.45$.

results are represented in Table 5.1. It is interesting to note that the non-smooth part of the curve belongs to the partition $[0, 2\pi]$ and is approximated by a polynomial of order six, as seen form Table 5.1. The error is within the specified limit. However the part of the curve is a straight line in the partition $[2\pi, 2\pi]$ and it is trivially approximated by a polynomial of degree 1, i.e., a straight line with the approximation error equal to zero.

Simulation IV As it can be observed from the last simulation that the error in the approximation $\| \sigma_1(\lambda) - p_i(\lambda) \|_\infty$ for the non-smooth partition is quite large. For certain applications better approximation may be required, so we desire to reduce the error. Given the curve (5.9), we want to approximate it such that the $\| \sigma_1(\lambda) - p_i(\lambda) \|_\infty < 0.15$. Similar to the previous simulation, the curve is defined on the interval $[0, 4\pi]$ and the interval $[0, 4\pi]$ is divided into 2 equal partitions of length 2π. As seen from Figure 5.4 the approximation is better compared to the previous case. The reason for the better approximation is that a more strict error tolerance $\epsilon = 0.15$ is used. However, it is interesting to observe from Table 5.4 that the order of the polynomial in the partition $[0, 2\pi]$ has increased significantly compared to the previous simulation while the order of the polynomial remains the same in the partition $[2\pi, 4\pi]$. This is because the curve to be approximated in the partition $[2\pi, 4\pi]$ is straight line and in is approximated with zero error with a straight line.

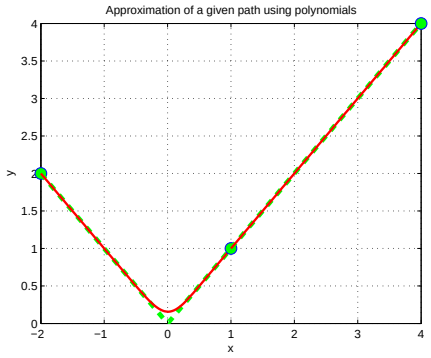


Figure 5.4: Approximation of curve (5.9) with $\epsilon = 0.15$.

i	Partition	$\| \sigma_1(\lambda) - p_i(\lambda) \|_\infty$	N
1	$[0, 2\pi]$	0.1470	51
2	$[2\pi, 4\pi]$	0.00	1

Table 5.4: Approximation results of curve (5.9) with $\epsilon = 0.15$.

5.4 Solution to the General Problem

The most general curves in $\mathbb{R}^2$ can be represented as (5.1). We solved our special problems in two parts; approximation and implicitization. The idea to solve the general problem is that we consider $\sigma_1(\lambda)$ and $\sigma_2(\lambda)$ as two separate curves. We take $\sigma_1(\lambda)$ and treat it as a special problem in order to approximate it with a polynomial. We then take $\sigma_2(\lambda)$ and treat it again as a special problem and approximate it with a polynomial.The approximated parameterized path can be represented as,

$$\sigma : \mathbb{R} \to \mathbb{R}^2 \tag{5.10}$$
$$\lambda \mapsto \begin{bmatrix} \sum_{i=0}^{n} a_i \lambda_{a,i}^n \\ \sum_{i=0}^{m} b_i \lambda_{b,i}^m. \end{bmatrix}$$

Although (5.10) is not a zero level representation, rather it is a parameterized representation. Its special structure allows us to convert it to a zero level set (implicit form).

5.4.1 Implicitization

Given a parametric curve of the form (5.10), we can represent (5.10) as a zero level set of a function, or implicit form

$$\sigma(\mathbb{R}) = \{y \in \mathbb{R}^p : s(y) = 0\}, \tag{5.11}$$

by the Sylvester matrix elimination method [52].

5.4.2 Sylvester Matrix Elimination Method

The Sylvester matrix elimination method is based on the concept of resultant of two polynomials, see Definition A.3.10. The resultant of two polynomials can easily be derived by thinking of polynomials as linear equations in powers of x. For ease of exposition, we consider the case where $p(x)$ is of second order polynomial and $q(x)$ is of third order polynomial. Once the basic idea has been grasped, it is straight forward to extend it to polynomials of any orders. Consider,

$$\begin{aligned} p(x) &= a_2x^2 + a_1x + a_0 = 0 \\ q(x) &= b_3x^3 + b_2x^2 + b_1x + b_0 = 0 \end{aligned} \tag{5.12}$$

The above equation can be written in matrix form as

$$\begin{bmatrix} 0 & a_2 & a_1 & a_0 \\ b_3 & b_2 & b_1 & b_0 \end{bmatrix} \begin{bmatrix} x^3 \\ x^2 \\ x \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{5.13}$$

It was shown by the authors in [52], Theorem 1, that the above system can be written as square matrix form by augmenting the system of equation $p(x)$ and $q(x)$ with new equations of form $x^{k-1}p(x) = 0$ or $x^{l-1}q(x) = 0$, where $k-1$ and $l-1$ are the order of

polynomials $q(x)$ and $p(x)$ respectively. Now consider the system,

$$\begin{aligned} p(x) &= 0 \\ xp(x) &= 0 \\ x^2p(x) &= 0 \\ q(x) &= 0 \\ xq(x) &= 0 \end{aligned} \tag{5.14}$$

so the above system of equation can be written in the matrix form as,

$$\begin{bmatrix} 0 & 0 & a_2 & a_1 & a_0 \\ 0 & a_2 & a_1 & a_0 & 0 \\ a_2 & a_1 & a_0 & 0 & 0 \\ 0 & b_3 & b_2 & b_1 & b_0 \\ b_3 & b_2 & b_1 & b_0 & 0 \end{bmatrix} \begin{bmatrix} x^4 \\ x^3 \\ x^2 \\ x \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{5.15}$$

One important property of homogeneous matrix equations involving square matrix in that they only have non-trivial solutions when the determinant of the matrix vanishes. The system $p(x) = 0, q(x) = 0$ only has a solution when p and q have a common root. Hence the determinant will vanish whenever p and q have a common root.

Now we can extend that method to any general integral polynomial. Consider, for $n, m \geq 1$

$$\begin{aligned} p(x) &= a_n x^n + \ldots + a_1 x + a_0 = 0 \\ q(x) &= b_m x^m + \ldots + b_1 x + b_0 = 0, \end{aligned} \tag{5.16}$$

the system can be written in the matrix form as

$$\begin{bmatrix} 0 & \cdots & \cdots & 0 & a_n & \cdots & a_1 & a_0 \\ 0 & \cdots & 0 & a_n & \cdots & a_1 & a_0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_n & \cdots & a_1 & a_0 & 0 & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & 0 & b_m & \cdots & b_1 & b_0 \\ 0 & \cdots & 0 & b_m & \cdots & b_1 & b_0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ b_m & \cdots & b_1 & b_0 & 0 & \cdots & \cdots & 0 \end{bmatrix} \begin{bmatrix} x^{n+m-1} \\ x^{n+m-2} \\ \vdots \\ x^n \\ \vdots \\ x^m \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{5.17}$$

In [63] the authors show that the resultant of the given two polynomials can be defined as a Sylvester matrix. Similarly, the Sylvester matrix for the given polynomials (5.16) can be written as,

$$S = \begin{bmatrix} 0 & \cdots & \cdots & 0 & a_n & \cdots & a_1 & a_0 - p \\ 0 & \cdots & 0 & a_n & \cdots & a_1 & a_0 - p & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_n & \cdots & a_1 & a_0 - p & 0 & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & 0 & b_m & \cdots & b_1 & b_0 - q \\ 0 & \cdots & 0 & b_m & \cdots & b_1 & b_0 - q & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ b_m & \cdots & b_1 & b_0 - q & 0 & \cdots & \cdots & 0 \end{bmatrix}. \tag{5.18}$$

The resultant of the Sylvester matrix (5.18) can be computed using Maple or Matlab's symbolic tool box. It is shown in [63], [52] that the resultant, which is an equation involving two variables x and y, represent the implicit form of the given rational parameterized curve defined by equations (5.16).

Implicitization of Parametric Curve

It is easy to implicitize a planar curve using elimination theory, and in fact the problem was addressed specifically in [52], [48], [49]. To illustrate the procedure consider the following example. Given a parameterized curve of form (5.10)

$$\begin{aligned} \sigma : \mathbb{R} &\to \mathbb{R}^2 \\ \lambda &\mapsto \begin{bmatrix} a_2\lambda^2 + a_1\lambda + a_0 \\ b_2\lambda^2 + b_1\lambda + b_0. \end{bmatrix} \end{aligned} \tag{5.19}$$

In particular (5.19) can be written as,

$$\begin{aligned} a_2\lambda^2 + a_1\lambda + (a_0 - x), \\ b_2\lambda^2 + b_1\lambda + (b_0 - y), \end{aligned} \tag{5.20}$$

the resultant of these two polynomials, which represent the implicit form of 5.19, is then defined by Sylvester matrix

$$S = \begin{bmatrix} a_2 & a_1 & a_0 - x & 0 \\ 0 & a_2 & a_1 & a_0 - x \\ b_2 & b_1 & b_0 - y & 0 \\ 0 & b_2 & b_1 & b_0 - y \end{bmatrix}. \tag{5.21}$$

Since the resultant expresses the relationship which must exist among the coefficients in order for there to exist a λ which simultaneously satisfies both equations, the resultant itself is the implicit form of the curve [52]. The determinant of (5.21) is given by,

$$\begin{aligned} |S| &= b_2^2x^2 - 2a_2b_2xy + a_2^2y^2 \\ &\quad + (2a_2b_2b_0 + a_1b_1b_2 - a_2b_1^2 - a_0b_2^2)x \\ &\quad + (a_2b_1a_1 + 2a_2b_2a_0 - 2a_2^2b_0 - a_1^2b_2)y \\ &\quad + (a_2b_0 - a_0b_2)^2 + (a_1b_2 - a_2b_1) \\ &\quad \times (a_1b_0 - a_0b_1) = 0. \end{aligned}$$

Using this method we can convert parametric equation (5.1)to implicit form which can be represented as a function of zero level set. Hence the general problem is solved. The general problem is solved in two steps i) A parameterized curve is converted to a rational parameterized form. ii) The rational parameterized curve is converted to implicit form relying relying on elimination theory. If the given curve is a rational parametric curve then the procedure discussed above implicitize the curve with zero error. Since, a general curve may not be a rational parametric curve, therefore, by using algorithm 2, we first approximate the given general parametric curve to a rational parametric curve and then we use the procedure discussed above to convert the approximated curve to implicit form. It is interesting to note that the implicitization error is equal to the approximation error because in the procedure of implicitization approximation occurs only at the first step. There is no approximation involve in step two discussed in Section 5.4.2.

Simulation Results

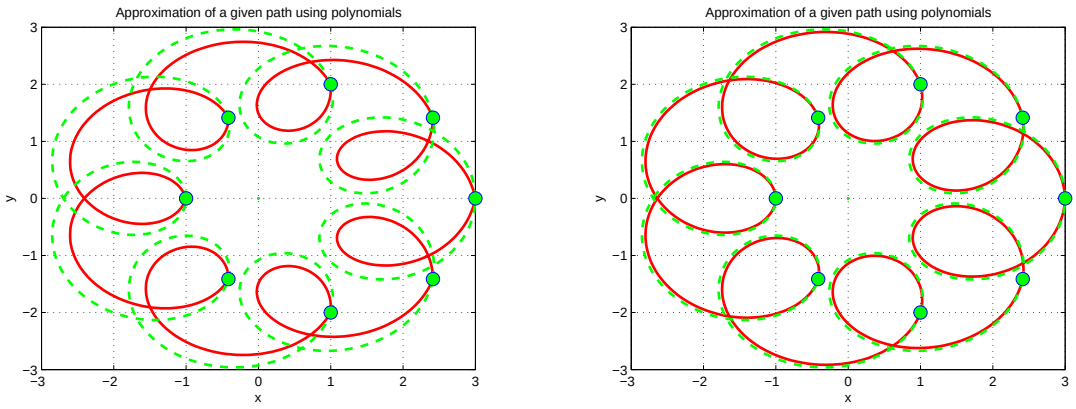
In this section, we apply the results of Section 5.4.2 and Algorithm 2 to few examples.

Simulation I and II In this simulation, we consider the following parametric curve,

$$\begin{aligned} \sigma &: \mathbb{R} \to \mathbb{R}^2 \\ \lambda &\mapsto \begin{bmatrix} 2\cos(\lambda) + \cos(8\lambda) \\ 2\sin(\lambda) + \sin(8\lambda) \end{bmatrix}. \end{aligned} \tag{5.22}$$

The curve is defined on the interval $[0, 2\pi]$. Furthermore, the interval $[0, 2\pi]$ is divided into 8 equal partitions. Given an error tolerance ϵ, we want to approximate the curve (5.22) with a sequence of polynomials $p_i(\lambda)$, such that $\max_{\lambda \in I_i} \|\sigma(\lambda) - p_i(\lambda)\| < \epsilon$. In the first case, the given error tolerance is $\epsilon = 0.25$. Figure 5.5(a) represents the parameterized curve (5.22) approximated with a sequence of polynomials $p_i(\lambda)$ for $\epsilon = 0.25$. In the second case, the same curve is approximated with a sequence of polynomials for $\epsilon = 0.05$ and the results

are shown in Figure 5.5(b). The original curve is represented by a dashed line while the approximated curve is represented by a solid line in Figure 5.5(a) and Figure 5.5(a). Solid dots in Figure 5.5(a) and Figure 5.5(a) represent the partition points. It is observed from the Figure 5.5 that the order of the polynomial increases as a small approximation error is when a smaller value of ϵ is used.



(a) $\epsilon = 0.25$. (b) $\epsilon = 0.05$

Figure 5.5: Approximation of curve (5.22).

The simulation parameter corresponding to Figure 5.5(a) are shown in Table 5.5 while the simulation parameters corresponding to Figure 5.5(a) are shown in Table 5.6. It in interesting to note from the Table 5.7 that the order of the approximated polynomial increases as we decrease the error tolerance. By approximating the curve (5.22) with polynomials means the curve (5.22) is represented by eight implicit rational parametric curves each defined on their corresponding intervals. By following the procedure of Sylvester matrix discussed in Section 5.4.2 the approximated rational parametric curve can be easily written as in the form of Sylvester matrix (5.18) and further converted to implicit by taking the determinant of the the sylvester matrix. The determinant can be easily computed by using Matlab or Maple symbolic tool box.

Simulation III and IV In this simulation we consider the following parametric curve,

$$\sigma : \mathbb{R} \to \mathbb{R}^2 \tag{5.23}$$
$$\lambda \mapsto \begin{bmatrix} \sqrt{\frac{M}{2}}\cos(\lambda) \\ \sqrt{\frac{M}{2}}\sin(\lambda), \end{bmatrix}.$$

The curve is called Cassinian Oval, where,

$$M = 2a^2\cos(2\lambda) + 2\sqrt{(-a^4 + b^4) + a^4(\cos(2\lambda))^2}.$$

i	$\| \sigma(\lambda) - p_i(\lambda) \|_\infty$	N
1	0.2199	3
2	0.2199	2
3	0.2199	2
4	0.2199	3
5	0.2199	3
6	0.2199	2
7	0.2199	2
8	0.2199	3

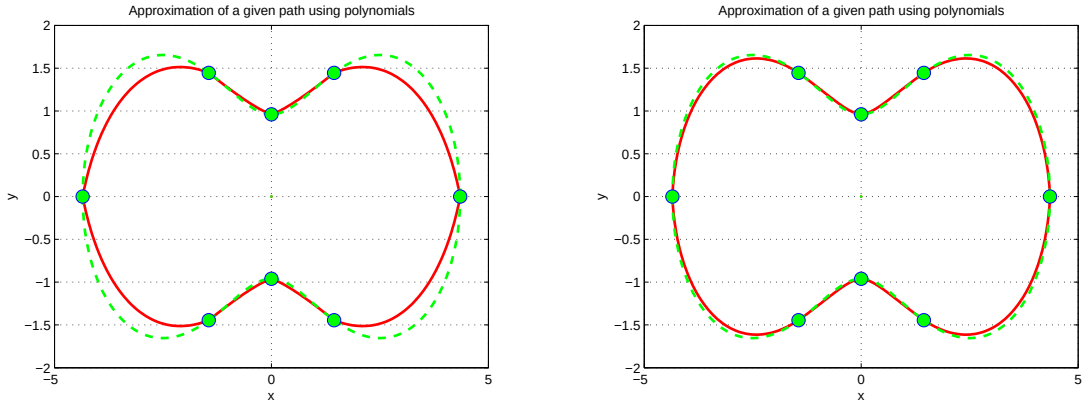
Table 5.5: $\epsilon = 0.25$

i	$\| \sigma(\lambda) - p_i(\lambda) \|_\infty$	N
1	0.0496	94
2	0.0496	96
3	0.0496	98
4	0.0496	100
5	0.0496	100
6	0.0496	98
7	0.0496	96
8	0.0496	94

Table 5.6: $\epsilon = 0.25$

Table 5.7: Approximation results of the curve (5.22)

Similar to the Simulation III and IV, the curve is defined on the interval $[0, 2\pi]$. Furthermore, the interval $[0, 2\pi]$ is divided into 8 equal partitions. For a given error tolerance ϵ, we want to approximate the curve (5.23) with a sequence of polynomials $p_i(\lambda)$, such that $\max_{\lambda \in I_i} \|\sigma(\lambda) - p_i(\lambda)\| < \epsilon$. In the first case the given error tolerance is $\epsilon = 0.25$. Figure 5.6(a) represents the parameterized curve (5.23) approximated with a sequence of polynomials $p_i(\lambda)$ for $\epsilon = 0.25$. In the second case the same curve is approximated with a sequence of polynomials for $\epsilon = 0.05$ and the results are shown in Figure 5.6(b). The original curve is represented by a dashed line while the approximated curve is represented by a solid line in Figure 5.5(a) and Figure 5.5(a). Solid dots in Figure 5.5(a) and Figure 5.5(a) represent the partition points. It is observed from the Figure 5.5 that the order of the polynomial increases as a small approximation error is when a smaller value of ϵ is used.



(a) $\epsilon = 0.25$.

(b) $\epsilon = 0.05$

Figure 5.6: Approximation of curve (5.23).

The simulation parameter corresponding to Figure 5.6(a) are shown in Table 5.8 while the simulation parameters corresponding to Figure 5.6(b) are shown in Table 5.9. It in interesting to note from the Table 5.10 that the order of the approximated polynomial increases as we decrease the error tolerance. Once the curve (5.23) is approximated by polynomials, i.e., the curve is converted to rational parameteric form by following the procedure of Sylvester matrix discussed in Section 5.4.2 the approximated rational parametric curve can be easily written as in the form of Sylvester matrix (5.18) and further converted to implicit form using Matlab or Maple symbolic tool box.

i	$\| \sigma(\lambda) - p_i(\lambda) \|_\infty$	N
1	0.2199	3
2	0.2199	2
3	0.2199	2
4	0.2199	3
5	0.2199	3
6	0.2199	2
7	0.2199	2
8	0.2199	3

Table 5.8: $\epsilon = 0.25$

i	$\| \sigma(\lambda) - p_i(\lambda) \|_\infty$	N
1	0.0480	15
2	0.0480	9
3	0.0480	9
4	0.0480	15
5	0.0480	15
6	0.0480	9
7	0.0480	9
8	0.0480	15

Table 5.9: $\epsilon = 0.25$

Table 5.10: Approximation results of the curve (5.22)

Chapter 6

Conclusion and Future Work

In the field of mobile robotics trajectory tracking and path following are the most fundamental tasks of practical interest. Path following is desirable in many situations because the invariance of path can be achieved by casting the problem of path following as a set stabilization problem.

In this thesis, the path following problem was investigated using feedback linearization for unicycle robot, car-like robot and the standard car-like robot attached with trailer systems. In Chapter 2, the problem is analyzed for the unicycle and it was proved that the path following controller works for a unit circle path. In Chapter 3, it was shown that the approach can be used for a large class of general curves. It was shown that, by dynamic extension, certain path following objectives can be achieved. The main drawback was that the procedure was not applicable for a large class of general nonlinear systems. However, the procedure is proposed for a large class of general curves. In this thesis mobile robots that fall in the category of differentially flat systems were considered. Instead of analyzing the problem from the point of view of differential flatness and using the differential flat outputs, physically meaningful outputs were used. It would be interesting in the future to use outputs or systems that are not differentially flat and design a controller using the proposed procedure.

In Chapter 4, a 1-trailer system attached with car-like robot was investigated. Although the system is differentially flat, the system is a non-regular system. Thus, the procedure designed in Chapter 2 and 3 cannot be used because the system does not have a well defined relative degree. By considering a circular curve, it was shown by simulation that the procedure can be used for such a system by approximating the vector fields of the original system. However, due to vastly complicated expressions a formal mathematical proof was not provided and a suggestion for future research is to prove results for the standard 1-trailer system. It is already proved that the general n-trailer system is not

differentially flat. Therefore, the application of the proposed technique to such systems would be of interest.

The procedure of dynamic transverse feedback linearization requires both zero level set or an implicit representation and a parametric representation of the curve. A lack of general procedure of implicitization restricts the generalization of the procedure for the general curves. In surveyed literature the path following or tracking problems are solved for specific systems and specific curves, and these methods lack the general procedure, one such example is the controller design by Lyapunov technique. One of our objectives in this thesis was to solve the path following problem for specific systems but also for a large class of general curves. This motivates representing general curves in a similar setting. In Chapter 5 a procedure is provided to approximate any given parametric curve with a rational parametric curve. It was shown based on elimination theory that the rational parametric curves can be converted to implicit form. One drawback of the proposed technique was that the order of the approximated rational parametric curve becomes very large as the approximation error becomes small. This results in the procedure being not well suited for applications where computation power is limited.

One of the future research directions in which the work can be extended include robustification of the controller and hybrid controller that switches through singularities. In this thesis one of the assumptions was that the curves are not self intersecting. It would be interesting in the future to prove results for self intersecting curves. A practical contribution could be to apply the results proven in this thesis to a practical robot. Another interesting future research work can be to prove result for a large class of general nonlinear system. The controller design approach used in this thesis was applied to few specific cases of mobile robots. The extension of this approach to other robotic systems in specific and other non-robotic systems in general would be quite interesting.

Appendix A

Basic Concepts

This Appendix reviews some of the basic concepts used in this thesis. We review very briefly some definitions from algebra, analysis and differential geometry. We discuss few definitions and important results of control system theory and feedback transformation that is used periodically in this book. Finally we give a brief overview of elimination theory and curve approximation. The purpose of this appendix is to give an informal and intuitive review of some of the basic tools used in this thesis. These concepts are taken from [29], [31], [51], [46], [60], [41], [25], [45].

A.1 Review of Algebra, Analysis and Differential Geometry

Informally a map is an operator taking elements from its domain, and generating elements in its co-domain. Let U and V be open subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively. A function f is sometimes called a mapping, and we say that f maps a domain element $a \in U$ to its codomain element $b \in V$, sometimes called the image of a. In symbols, we might write $f : U \to V$ and $f : a \mapsto b$. Surjective, injective and bijective maps are the basis properties of maps.

Definition A.1.1. *A map $f : U \subseteq \mathbb{R}^n \to V \subseteq \mathbb{R}^m$ is surjective or onto if for each $y \in V$ there exist at least one $x \in U$ such that $f(x) = y$.*

Definition A.1.2. *A map $f : U \subseteq \mathbb{R}^n \to V \subseteq \mathbb{R}^m$ is injective or one-to-one if, $x_1, x_2 \in U, f(x_1) = f(x_2)$ implies $x_1 = x_2$.*

Definition A.1.3. *A map $f : U \subseteq \mathbb{R}^n \to V \subseteq \mathbb{R}^m$ is bijective if it is both injective and surjective.*

Definition A.1.4. *A group G is a set with a binary operation $(.) : G \times G \mapsto G$, such that the following properties are satisfied:*

1. *associativity: $(a.b).c = a.(b.c)$ for all $a, b, c \in G$*
2. *$\exists$ an identity element e such that $e.a = a.e = a$ for all $a \in G$*
3. *$\forall a \in G$ there exists an inverse a^{-1} such that $a.a^{-1} = a^{-1}.a = e$*

Definition A.1.5. *A homomorphism between groups, $\phi : G \mapsto H$, is a map which preserves the group operation*

$$\phi(a.b) = \phi(a).\phi(b).$$

Definition A.1.6. *An isomomorphism is a homomorphism that is bijective.*

Smooth Manifold and Smooth Maps Roughly speaking, manifold are ,locally, vector spaces but are globally curved spaces. For example the surface of earth is "locally flat" but globally curved and globally the surface of earth is not a vector field. Although manifolds resemble Euclidean spaces near each point ("locally"), the global structure of a manifold may be more complicated. For example, any point on the usual two-dimensional surface of a sphere is surrounded by a circular region that can be flattened to a circular region of the plane, as in a geographical map. However, the sphere differs from the plane.

Let U and V be open subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively. A mapping $f : U \mapsto V$ is called smooth if f is differentiable and the derivative of the map $\partial f/\partial x$ is continuous. In this case the function f is of class C^1. If f is r^{th} order differentiable and $\partial^r f/\partial x$ is continuous then we say f is of class C^r. If f is smooth for all finite r then we say f is *smooth* or of class C^∞

Definition A.1.7. *A map $f : U \subset \mathbb{R}^n \to V \subset \mathbb{R}^m$ is diffeomorphism if f is a homeomorphism (i.e., a one-to-one or injective continuous map with continuous inverse) and if both f and f^{-1} are smooth.*

Definition A.1.8. *A subset $M \subset \mathbb{R}^k$ is called a smooth manifold of dimension m if for each $x \in M$ there is a neighborhood $W \cap M$, where $W \subset \mathbb{R}^k$, that is a diffeomorphic to an open subset $U \subset \mathbb{R}^m$*

A submanifold is simply a smaller manifold inside a larger manifold.

A.1.1 Vector fields and their Derivatives

A vector field is an assignment of a vector to each point in a subset of Euclidean space. A vector field in the plane for instance can be visualized as an arrow, with a given magnitude and direction, attached to each point in the plane. Vector fields are often used to model speed and direction of a moving objects throughout space, for example speed and direction of a mobile robot. The following demonstrates the notion of vector field,

$$f(x) = \begin{bmatrix} x_3^2 \\ x_2 \\ 1 + x_1^2 \end{bmatrix} = \frac{\partial x_3^2}{\partial x_1} + \frac{\partial x_2}{\partial x_2} + \frac{\partial (1 + x_1^2)}{\partial x_3}. \tag{A.1}$$

The Lie derivative also called the direction derivative evaluates the change of a vector field along the flow of another vector field. This change is coordinate invariant and therefore the Lie derivative is defined on any differentiable manifold.

Definition A.1.9. *Consider a vector field f and a real valued function,*

$$\lambda : U \subseteq \mathbb{R}^n \to \mathbb{R}, \tag{A.2}$$

the derivative of λ along f is a function $L_f\lambda : U \to \mathbb{R}$ defined as

$$L_f\lambda(x) := \langle d\lambda(x), f(x) \rangle = \frac{\partial \lambda}{\partial x} f(x) = \sum_{i=1}^{n} \frac{\partial \lambda}{\partial x_i} f_i(x), \tag{A.3}$$

which is also called the Lie derivative or directional derivative of λ along f, where $\langle ., . \rangle$ is the Euclidean inner product.

Repeated use of this operator is possible and the following notation can be used,

$$L_g L_f \lambda(x) := \frac{\partial L_f \lambda(x)}{\partial x} g(x) = \sum_{i=1}^{n} \frac{\partial L_f \lambda}{\partial x_i} g_i(x). \tag{A.4}$$

The operation can be recursively defined, such that taking the k derivatives of λ along f would be denoted by L_f^k where,

$$L_f^k \lambda(x) := \frac{\partial L_f^{k-1} \lambda(x)}{\partial x} f(x) = \sum_{i=1}^{n} \frac{\partial L_f^{k-1} \lambda}{\partial x_i} f_i(x). \tag{A.5}$$

A.2 Nonlinear Control Systems

Consider a time-invariant, finite-dimensional, deterministic control-affine system with m inputs, $u := [u_1 \cdots u_m]^\top \in \mathbb{R}^m$ and p outputs and $f : \mathbb{R}^n \to \mathbb{R}^n$, $g_i : \mathbb{R}^n \to \mathbb{R}^n$ and $h : \mathbb{R}^n \to \mathbb{R}^p$ are smooth C^r maps.

$$\dot{x} = f(x) + \sum_{i=1}^{m} g_i(x)u_i := f(x) + g(x)u, \tag{A.6}$$

$$\tag{A.7}$$

and consider a function,

$$y = h(x) = \begin{bmatrix} h_1(x) \\ \vdots \\ h_p(x) \end{bmatrix}, \forall y \in \mathbb{R}^p, \tag{A.8}$$

which is the output of the system. The relative degree is the key concept in solving feedback linearization problem.

Definition A.2.1. *Consider system* (A.6) *with* $u \in \mathbb{R}$ *and with output function* (A.8) *with* $m = p = 1$ *i.e.,* $y = h(x)$, $y \in \mathbb{R}$. *The system has a relative degree of* r *at a point* x_0 *if*

1. $L_g L_f^k h(x) = 0, \forall x \in$ *a neighborhood of* x_0 *and* $\forall k < r - 1$,
2. $L_g L_f^{r-1} h(x_0) \neq 0$.

The relative degree of a single input single output (SISO) system in the number of times we need to differentiate the output before the control input appears. A notion, called the vector relative degree can be defined for the multiple-input multiple-output (MIMO) systems.

Definition A.2.2. *Consider system A.6 with* $m = p$. *We define an* $m \times m$ *matrix,*

$$A(x) := \begin{bmatrix} L_{g_1} L_f^{r_1 - 1} h_1(x) & \cdots & L_{g_m} L_f^{r_1 - 1} h_1(x) \\ L_{g_1} L_f^{r_2 - 1} h_2(x) & \cdots & L_{g_m} L_f^{r_2 - 1} h_2(x) \\ \vdots & \ddots & \vdots \\ L_{g_1} L_f^{r_m - 1} h_m(x) & \cdots & L_{g_m} L_f^{r_m - 1} h_m(x) \end{bmatrix}. \tag{A.9}$$

The system has a vector relative degree of $\{r_1, \ldots r_m\}$ *at a point* x_0 *if*

1. $L_{g_j} L_f^k h_i(x) = 0, \forall 1 \leq j \leq m$ *for all* $k < r_i - 1$ *for all* $1 \leq i \leq m$ *and for all* x *in a neighborhood of* x_0.

2. *The matrix* $A(x)$ *is nonsingular at* $x = x_0$,

A.2.1 Feedback Linearization

Feedback linearization is a way of transforming the original nonlinear system model into equivalent model of simpler form. The central idea of feedback linearization is to algebraically transform nonlinear systems dynamics into (fully or partly) linear ones, so that linear control techniques can be applied. This differs entirely from conventional (Jacobian) linearization, because feedback linearization is achieved by exact state transformation and feedback, rather than by linear approximations of the dynamics.

SISO Input-Output Feedback Linearization

A control technique where the output of the dynamic system is differentiated until the physical input appears in the derivative of the output. Consider (A.6) with $m = p = 1$. If the output of the system yields a well defined relative degree it can be transformed into a linear input-output map via a coordinate and feedback transformation. By applying the local change of coordinates

$$\begin{aligned} T : U &\to T(U) \\ x &\mapsto (\eta, \xi), \end{aligned}$$

where,

$$T = \begin{bmatrix} \varphi_1(x) \\ \vdots \\ \varphi_{n-r}(x) \\ h(x) \\ L_f h(x) \\ \vdots \\ L_f^{r-1} h(x) \end{bmatrix},$$

and,

$$\eta := \varphi(x) = \begin{bmatrix} \varphi_1(x) & \varphi_2(x) & \cdots & \varphi_{n-r}(x) \end{bmatrix}^\top,$$

and

$$\xi := \begin{bmatrix} h(x) & L_f h(x) & \cdots & L_f^{r-1} h(x) \end{bmatrix}^\top.$$

Proposition 4.1.3 in Isidori [29] shows the possibility of finding $n-r$ more functions $\varphi(x)$ such that the coordinate transformation T is diffeomorphism, at least locally in the neighborhood of x_0. The system can be described in the new coordinates reads,

$$\begin{aligned}
\dot{\eta} &= \frac{d\varphi(x(t))}{dt} \\
&= \frac{\partial \varphi(x)}{\partial x}\dot{x} \\
&= \frac{\partial \varphi(x)}{\partial x}(f(x)+g(x)u)\Big|_{x=T^{-1}(\eta,\xi)} \\
&= \frac{\partial \varphi(x)}{\partial x}f(x)\Big|_{x=T^{-1}(\eta,\xi)} + \frac{\partial \varphi(x)}{\partial x}g(x)u\Big|_{x=T^{-1}(\eta,\xi)} \\
&= p(\eta,\xi)+\sum_{i=1}^{m} q_i(\eta,\xi)u_i \\
&:= p(\eta,\xi)+q(\eta,\xi)u,
\end{aligned}$$

and

$$\begin{aligned}
\dot{\xi}_1 &= \xi_2 \\
&\vdots \\
\dot{\xi}_{r-1} &= \xi_r \\
\dot{\xi}_r &= L_f^r h(x)\big|_{x=T^{-1}(\eta,\xi)} + L_g L_f^{r-1} h(x) u\big|_{x=T^{-1}(\eta,\xi)} \\
&:= b(\eta,\xi)+a(\eta,\xi)u.
\end{aligned}$$

If $a(\eta,\xi)$ is bounded away from zero for all x in the neighborhood of x_0, then we can choose a smooth, regular, static feedback

$$u = \frac{1}{a(\eta,\xi)}(-b(\eta,\xi)+\upsilon).$$

Hence, by partial input-output feedback linearization the system can be represented in the following partial linear form.

$$
\begin{aligned}
\dot{\eta} &= p(\eta, \xi) \\
\dot{\xi}_1 &= \xi 2 \\
&\vdots \\
\dot{\xi}_{r-1} &= \xi r \\
\dot{\xi}_r &= \upsilon.
\end{aligned}
$$

MIMO Input-Output Feedback Linearization

The concept of MIMO system is analogus to to SISO sytem. If the output of the system yields a well defined vector relative degree it can be transformed into a linear input-output form via a coordinate and feedback transformation. By applying the local change of coordinates

$$
\begin{aligned}
T : U &\to T(U) \\
x &\mapsto (\eta, \xi),
\end{aligned}
$$

such that

$$T(x) = \begin{bmatrix} \varphi_1(x) \\ \varphi_2(x) \\ \vdots \\ \varphi_{n-\sum_{i=1}^{m} r_i(x)}(x) \\ h_1(x) \\ L_f h_1(x) \\ \vdots \\ L_f^{r_1-1} h_1(x) \\ h_2(x) \\ L_f h_2(x) \\ \vdots \\ L_f^{r_2-1} h_2(x) \\ \vdots \\ L_f^{r_m-1} h_m(x) \end{bmatrix},$$

where $\{r_1 \cdots r_m\}$ is the vector relative degree of the system. we define,

$$\eta := \varphi(x) = \begin{bmatrix} \varphi_1(x) & \varphi_2(x) & \cdots & \varphi_{n-\sum_{i=1}^{m} r_i(x)} \end{bmatrix}^\top,$$

and

$$\xi := \begin{bmatrix} h_1(x) & L_f h_1(x) & \cdots & L_f^{r_1-1} h_1(x) & \cdots h_m(x) & L_f h_m(x) & \cdots & L_f^{r_m-1} h_m(x) \end{bmatrix}^\top,$$

by ξ_j^i we mean j^{th} derivative of the i^{th} input. The η-dynamics in MIMO case is similar to SISO case. However, ξ-dynamics for the multiple inputs are

$$\begin{aligned} \dot{\xi}_1^i &= \xi_2^i \\ \dot{\xi}_2^i &= \xi_3^i \\ &\vdots \\ \dot{\xi}_{r_i-1}^i &= \xi_{r_i}^i \\ \dot{\xi}_{r_i} &= b_i(\eta, \xi) + \sum_{k=1}^{m} a_{ik}(\eta, \xi) u_k, \end{aligned}$$

with $i \in \{1, \cdots, m\}$ and where,

$$\begin{aligned} a_{ik}(\eta, \xi) &= L_{g_k} L_f^{r_i - 1} h_i(x)\big|_{x = T^{-1}(\eta, \xi)}, k \in \{1, \cdots, m\}, \\ b_i(\eta, \xi) &= L_f^{r_i} h_i(x)\big|_{x = T^{-1}(\eta, \xi)}, \end{aligned}$$

where g_k denotes the k^{th} column of g. From Lemma 5.1.1 and Proposition 5.1.2 in Isidori [29] shows that it is always possible to choose $n - r$ more functions $\varphi(x)$ such that T is diffeomorphism, in the neighborhood of x_0. We define,

$$\overline{\beta}(\eta, \xi) := \begin{bmatrix} -b_1(\eta, \xi) \\ -b_2(\eta, \xi) \\ \vdots \\ -b_m(\eta, \xi) \end{bmatrix},$$

and

$$\overline{\alpha}(\eta, \xi) := \begin{bmatrix} a_{11}(\eta, \xi) & a_{12}(\eta, \xi) & \cdots & a_{1m}(\eta, \xi) \\ a_{21}(\eta, \xi) & a_{22}(\eta, \xi) & \cdots & a_{2m}(\eta, \xi) \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}(\eta, \xi) & a_{m2}(\eta, \xi) & \cdots & a_{mm}(\eta, \xi) \end{bmatrix}^{-1}.$$

This suggests a control law

$$u = \overline{\alpha}(\eta, \xi)(\overline{\beta}(\eta, \xi) + \upsilon),$$

where $\upsilon \in \mathbb{R}^m$. This feedback transforms the system (A.6) into a partial linear input-output system,

$$\begin{aligned} \dot{\eta} &= p(\eta, \xi) + q(\eta, \xi)\upsilon \\ \dot{\xi}_1^i &= \xi_2^i \\ &\vdots \\ \dot{\xi}_{r_i - 1}^i &= \xi^i r_i \\ \dot{\xi}_{r_i}^i &= \upsilon_i. \end{aligned}$$

A.3 Curve Approximation

The controller design proposed in this thesis relies on parameterized as well as implicit representation of curves. The curve approximation and implicitization procedure in the thesis rely on these mathematical concepts drawn from [60], [46], [18], [16], [12].

Definition A.3.1. *Let U and V be open sets of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively. A function $f : U \longrightarrow V$ is* ***uniformly continuous*** *if*

$$(\forall \epsilon > 0)\,(\exists \delta > 0)\,(\forall x, y \in U)\,\|x - y\| < \delta \Rightarrow \|f(x) - f(y)\| < \epsilon.$$

Unlike ordinary continuity, for uniform continuity δ can not depend on $x \in U$. Uniform continuity is a stronger condition than continuity at a point. Uniform continuity implies continuity but the converse is not true.

Theorem A.3.2 ([46] Theorem 44)**.** *Every continuous function defined on a compact set is uniformly continuous.*

Definition A.3.3. *Let f_n be a sequence of real valued functions. The function f_n converges uniformly to f on I if for each $\epsilon > 0$ there exits N such that for every $n > N$ the inequality $|f_n(x) - f(x)| < \epsilon$ holds for all $x \in I$.*

Definition A.3.4. *Let $x \in \mathbb{R}$ and for each $n \in \mathbb{N}$, $k \in \{0, \ldots, n\}$, consider the polynomial*

$$p_n^k(x) := \binom{n}{k} x^k (1 - x)^{n-k}, \tag{A.10}$$

where $\binom{n}{k}$ is the binomial coefficient $n!/k!(n-k)!$. The polynomial $p_n^k(x)$ is called a basic Bernstein polynomial of degree n.

For each fixed n, there are $n+1$ basic Bernstein polynomials of degree n corresponding to $k \in \{0, \ldots, n\}$. These basic polynomials form a basis for the finite-dimensional vector space of polynomials of degree n.

Let $[a, b] \subset \mathbb{R}$ denote a compact interval of the real line with end points $a < b$. Let $C^0([a, b], \mathbb{R})$ denote the set of continuous real-valued functions with compact domain $[a, b]$. Our interest in the basic Bernstein polynomials comes from the fact that they can be used to prove the following theorem.

Theorem A.3.5 (Weierstrass Approximation Theorem)**.** *The set of polynomials is dense in $C^0([a, b], \mathbb{R})$.*

Density means that for each $f \in C^0([a,b],\mathbb{R})$ and each $\epsilon > 0$ there exists a polynomial function $p(x)$ such that for all $x \in [a,b]$,

$$|f(x) - p(x)| < \epsilon.$$

There are various proofs of Theorem A.3.5 in the literature but we rely on the constructive proof presented in 1912 by S.N. Bernstein. For the complete proof the readers are referred to [18].

Given a continuous function $f : [a,b] \longrightarrow \mathbb{R}$, there is no loss of generality to assume that the interval $[a,b]$ is $[0,1]$. Bernstein's proof of Theorem A.3.5 uses the basic polynomials (A.10) and the function $f : [0,1] \longrightarrow \mathbb{R}$ to define a sequence of polynomials

$$B_n^f(x) := \sum_{k=0}^{n} c_k p_n^k(x), \tag{A.11}$$

where

$$c_k := f\left(k/n\right).$$

The polynomial $B_n^f(x)$ is a polynomial of degree at most n and is called a Bernstein polynomial. The n^{th} Bernstein polynomial $B_n^f(x)$ converge uniformly to f as $n \to \infty$ [46].

Definition A.3.6. *Let k be a field. A polynomial $f \in k[x_1, \cdots, x_n]$ is irreducible over k if it is nonconstant and is not the product of two nonconstant polynomials in $k[x_1, \cdots, x_n]$.*

This leads to the following result.

Proposition A.3.7. *Every nonconstant polynomial $f \in k[x_1, \cdots, x_n]$ can be written as a product of polynomials which are irreducible over k.*

Proposition A.3.8. *Let $f \in k[x_1, \cdots, x_n]$ be irreducible over k and suppose that f divides the product gh, for $g, h \in k[x_1, \cdots, x_n]$. Then f divides either g or h.*

Proposition A.3.9. *Let $f, g \in k[x]$ be polynomials of degree $l > 0$ and $m > 0$, respectively. Then f and g have a common factor if and only if there are polynomials $A, B \in k[x]$ such that*

1. *A and B are not both zero,*
2. *A has degree at most $m-1$ and B has degree at most $l-1$,*
3. *$Af + Bg = 0$.*

Definition A.3.10. *Given polynomials $f, g \in k[x]$ of positive degree, write them in the form*

$$f = a_0 x^l + \cdots + a_l, \quad a_0 \neq 0,$$
$$g = b_0 x^m + \cdots + b_m, \quad b_0 \neq 0.$$

Then the Sylvester matrix of f and g with respect to x is the coefficient matrix of the system of equation given above. The Sylvester matrix is the following $(l+m) \times (l+m)$ matrix:

$$Syl(f,g,x) := \begin{bmatrix} a_0 & 0 & \cdots & 0 & b_0 & 0 & \cdots & 0 \\ a_1 & a_0 & \ddots & \vdots & b_1 & b_0 & \ddots & \vdots \\ a_2 & a_1 & \ddots & 0 & b_2 & b_1 & \ddots & 0 \\ \vdots & & \ddots & a_0 & \vdots & & \ddots & b_0 \\ & \vdots & & a_1 & \vdots & & b_1 & \\ a_{l-1} & & & & b_{m-1} & & & \\ a_l & a_{l-1} & & \vdots & b_m & b_{m-1} & & \vdots \\ 0 & a_l & \ddots & & 0 & b_m & \ddots & \\ \vdots & \ddots & \ddots & a_{l-1} & \vdots & \ddots & \ddots & b_{m-1} \\ 0 & \cdots & 0 & a_l & 0 & \cdots & 0 & b_m \end{bmatrix}. \tag{A.12}$$

The resultant of f and g with respect to x, denoted by $Res(f,g,x)$ is the determinant of the Sylvester matrix,

$$Res(f,g,x) = det(Syl(f,g,x)).$$

Proposition A.3.11. *Given $f, g \in k[x]$ of positive degree, the resultant $Res(f,g,x) \in k$ is a polynomial in the coefficients of f and g. Furthermore, f and g have a common factor in $k[x]$ if and only if $Res(f,g,x) = 0$.*

Proposition A.3.12. *Given $f, g \in k[x]$ of positive degree, there are polynomials $A, B \in k[x]$ such that*

$$Af + Bg = Res(f,g,x).$$

Furthermore, the coefficients of A and B are integer polynomials in the coefficients of f and g.

Appendix B

Matlab Codes

B.1 Transverse Feedback Linearization: code

Following is the code that computes all the η-states for the car-like robot attached with one trailer system.

```
%%%%%%%%% car-like robot with constant speed %%%
clc
clear all
close all
syms x1 x2 x3 x4 x5 v L u d1
syms a0 a1 a2 a3 k1 k2

%% System
fx=[v*cos(x3);v*sin(x3);(v/L)*tan(x4);0;(v/d1)*sin(x3-x5)];
gx=[0;0;0;1;0];
%% Path
%S=x2-cos(x1); %% Sinusoidal path
S=(x1-d1*cos(x5))^2+(x2-d1*sin(x5))^2-1;
%% First Lie Derivative
pd=[diff(S,x1) diff(S,x2) diff(S,x3) diff(S,x4) diff(S,x5)];
LfS=pd*fx;
LgS=pd*gx*u
S_dot=LfS+LgS
S_dot_simplify=simplify(S_dot);
pretty(S_dot_simplify);

%% Second Lie derivative
pd=[diff(S_dot,x1) diff(S_dot,x2) diff(S_dot,x3) diff(S_dot,x4) ...
    diff(S_dot,x5)];
```

```
Lf2S=pd*fx;
LgLfS=pd*gx*u
S_ddot=simplify(Lf2S+LgLfS)
% % %
%% Third Derivative
pd=[diff(S_ddot,x1) diff(S_ddot,x2) diff(S_ddot,x3) diff(S_ddot,x4) ...
    diff(S_ddot,x5)];
Lf3S=pd*fx
LgLf2S=pd*gx
LgLf2S =0;
%u=(-Lf3S-k1*S-k1*LfS-k2*Lf2S)/(LgLf2S)
S_d3dot=simplify(Lf3S+LgLf2S*u);

%% Forth Derivative
pd=[diff(S_d3dot,x1) diff(S_d3dot,x2) diff(S_d3dot,x3) ...
    diff(S_d3dot,x4) diff(S_d3dot,x5)];
Lf4S=pd*fx
LgLf3S=pd*gx

u=( -Lf4S -k1*S-k1*LfS-k2*Lf2S-k1*Lf3S)/(LgLf3S)
```

B.2 Dynamic Transverse Feedback Linearization: code

Following is the code that computes all the η and ξ-states for the car-like robot attached with one trailer system.

```
%%%%%%%%% car-like robot attached with one trailer system %%%
clc
clear all
close all
syms x1 x2 x3 x4 x5 x6 x7 v L u d1
syms a0 a1 a2 a3 k1 k2
X=[x1 x2 x3 x4 x5 x6 x7];

%% System
fx=[(v+x6)*cos(x3);(v+x6)*sin(x3);...
(v+x6)/L)*tan(x4);0;((v+x6)/d1)*sin(x3-x5);x7;0];
g1x=[0;0;0;1;0;0;0];
g2x=[0;0;0;0;0;0;1];
%% Path: Circular Path
S=(x1-L*cos(x5))^2+(x2-L*sin(x5))^2-1
%% First Lie Derivative
pd=[diff(S,x1) diff(S,x2) diff(S,x3) diff(S,x4) diff(S,x5) diff(S,x6) ...
    diff(S,x7)];
```

```
LfS=pd*fx;
Lg1S=pd*g1x*u
Lg2S=pd*g2x*u
S_dot=LfS+Lg1S+Lg2S
S_dot_simplify=simplify(S_dot)
pretty(S_dot_simplify);
%% Second Lie derivative
pd=[diff(S_dot,x1) diff(S_dot,x2) diff(S_dot,x3) diff(S_dot,x4) ...
    diff(S_dot,x5) diff(S_dot,x6) diff(S_dot,x7)];
Lf2S=pd*fx;
Lg1LfS=pd*g1x*u
Lg2LfS=pd*g2x*u
S_ddot=simplify(Lf2S+Lg1LfS+Lg2LfS)
% % %
%% Third Lie Derivative

pd=[diff(S_ddot,x1) diff(S_ddot,x2) diff(S_ddot,x3) diff(S_ddot,x4) ...
    diff(S_ddot,x5) diff(S_ddot,x6) diff(S_ddot,x7)];
Lf3S=pd*fx
Lg1Lf2S=pd*g1x
Lg2Lf2S=pd*g2x
S_d3dot=simplify(Lf3S);

%% Forth Derivative
pd=[diff(S_d3dot,x1) diff(S_d3dot,x2) diff(S_d3dot,x3) ...
    diff(S_d3dot,x4) diff(S_d3dot,x5) diff(S_d3dot,x6) diff(S_d3dot,x7)];
Lf4S=pd*fx
Lg1Lf3S=pd*g1x
Lg2Lf3S=pd*g2x

%% Eta states
P=atan( (x2-L*sin(x5)) / (x1-L*cos(x5)) );
%% First Lie Derivative
pd=jacobian(P,X)
LfP=pd*fx
Lg1P=pd*g1x
Lg2P=pd*g2x
P_dot=simplify(LfP+Lg1P+Lg2P);
eta2=P_dot;
%% Second Lie derivative
pd=jacobian(P_dot,X)
Lf2P=pd*fx
Lg1LfP=pd*g1x
Lg2LfP=pd*g2x
P_ddot=simplify(Lf2P+Lg1LfP+Lg2LfP);
eta3=P_ddot;
%% Third Derivative
pd=jacobian(P_ddot,X);
```

```
Lf3P=pd*fx
Lg1Lf2P=simplify(pd*g1x)
Lg2Lf2P=simplify(pd*g2x)
P_d3dot=simplify(Lf3S)

%% Fourth Derivative Derivative
pd=jacobian(P_d3dot,X);
Lf4P=pd*fx
Lg1Lf3P=simplify(pd*g1x)
Lg2Lf3P=simplify(pd*g2x)

d21=Lg1Lf2P;
d22=Lg2Lf2P;
d11=Lg1Lf3S;
d12=Lg2Lf3S;


D=[d21 d22;d11 d12]
M=inv(D);
```

Bibliography

[1] C. Samson A. Micaelli. Trajectory tracking for unicycle-type and two-steering-wheels mobile robots. Technical report, Institut National De Recherche En Informatique Et En Automatique, 1993.

[2] A. P. Aguiar, J. P. Hespanha, and P. V. Kokotović. Performance limitations in reference tracking and path following for nonlinear systems. *Automatica*, 44(3):598–610, 2008.

[3] C. Aguilar. Approximate feedback linearization and sliding mode control for the single inverted pendulum. Technical report, Queen's University, 2002.

[4] C. Altafini. Some properties of the general n-trailer. *International Journal of Control*, 74:409–424, 2001.

[5] C. Altafini. Following a path of varying curvature as an output regulation problem. *Automatic Control, IEEE Transactions on*, 47(9):1551 – 1556, sep 2002.

[6] C. Altafini. Path following with reduced off-tracking for multibody wheeled vehicles. *Control Systems Technology, IEEE Transactions on*, 11(4):598 – 605, july 2003.

[7] A. Alvarez-Aguirre, N. V. D. Wouw., T. Oguchi., K. Kojima, and H. Nijmeijer. *Remote tracking control of unicycle robots with network-induced delays*, volume 89 LNEE of *Lecture Notes in Electrical Engineering*. 2011.

[8] A. Astolfi, P. Bolzern, and A. Locatelli. Path-tracking of a tractor-trailer vehicle along rectilinear and circular paths: a lyapunov-based approach. *Robotics and Automation, IEEE Transactions on*, 20(1):154 – 160, feb. 2004.

[9] A. Banaszuk and J. Hauser. Feedback linearization of transverse dynamics for periodic orbits. *Systems and Control Letters*, 26(2):95 – 105, 1995.

[10] P. Bolzern, R. M. DeSantis, A. Locatelli, and D. Masciocchi. Path-tracking for articulated vehicles with off-axle hitching. *Control Systems Technology, IEEE Transactions on*, 6(4):515 –523, jul 1998.

[11] R. W. Brockett. Asymptotic stability and feedback stabilization. *Differential Geometric Control Theory*, 27:181–191, 1983.

[12] C. Bruun. Vigre computational algebraic geometry junior seminar. University Lecture, 2008.

[13] L.G. Bushnell, D.M. Tilbury, and S.S. Sastry. Steering three-input nonholonomic systems: the fire truck example. *Int. J. Rob. Res.*, 14(4):366–381, 1995.

[14] L. Consolini, M. Maggiore, C. Nielsen, and M. Tosques. Path following for the pvtol aircraft. *Automatica*, 46:1284–1296, August 2010.

[15] L. Consolini, A. Piazzi, and M. Tosques. Path following of car-like vehicles using dynamic inversion. *International Journal of Control*, pages 1724–1738, 2003.

[16] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Springer, Aug 13 1998.

[17] O. H. Dağci, Ü. Y. Ogras, and Ü. Özguner. Path following controller design using sliding mode control theory. In *American Control Conference, 2003. Proceedings of the 2003*, volume 1, pages 903 – 908 vol.1, june 2003.

[18] K. R. Davidson and A.P. Donsig. *Real Analysis with Real Applications*. Pearson, 2002.

[19] V. Deligiannis, G. Davrazos, S. Manesis, and T. Arampatzis. Flatness conservation in the n-trailer system equipped with a sliding kingpin mechanism. *Journal of Intelligent & Robotic Systems*, 46:151–162, 2006.

[20] M. I. El-Hawwary and M. Maggiore. Global path following for the unicycle and other results. In *American Control Conference, 2008*, pages 3500 –3505, june 2008.

[21] M. Fliess, J. Lévine, Ph. Martin, and P. Rouchon. On differentially flat nonlinear systems. *Proceedings. IFAC-Symposium NOLCOS'92 Bordeaux*, pages 408–412, 1992.

[22] M. Fliess, J. Lévine, Ph. Martin, and P. Rouchon. A lie-backlund approach to equivalence and flatness of nonlinear systems. *Automatic Control, IEEE Transactions on*, 44(5):922 –937, May 1999.

[23] V. Guillemin and A. Pollack. *Differential Topology*. Prentice-Hall, Englewood Cliffs NJ, 1974.

[24] J. Hauser, S. Sastry, and P. Kokotovic. Nonlinear control via approximate input-output linearization: the ball and beam example. In *Proceedings of the 28th IEEE Conference on Decision and Control*, 1989.

[25] A. Hladio. Path following for mechanical systems applied to robotic manipulators. Master's thesis, Department of Electrical and Computer Engineering University of Waterloo, 2010.

[26] A. Hladio, C. Nielsen, and D. Wang. Path following controller design for a class of mechanical systems. In *18th World Congress of the International Federation of Automatic Control*, Milano, Italy, August 2011. Accepted.

[27] A. Hladio, C. Nielsen, and D. Wang. Path following for mechanical systems: Experiments and examples. In *American Control Conference*, June 2011. Accepted.

[28] G. Indiveri and M. L. Corradini. Switching linear path following for bounded curvature car-like vehicles. In *5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, july 2004.

[29] A. Isidori. *Nonlinear Control Systems*. Springer-Verlag New York, Inc., Secaucus, NJ, U.S.A, 1995.

[30] J. Jakubiak, E. Lefeber, K. Tchon, and H. Nijmeijer. Two observer-based tracking algorithms for a unicycle mobile robot. *Int. Journal of Applied Math. and Comp. Science*, 12(4):513–522, 2002.

[31] H. K. Khalil. *Nonlinear systems*. Prentice Hall, 2002.

[32] L. Lapierre, R. Zapata, and P. Lepinay. Combined path-following and obstacle avoidance control of a wheeled robot. *Int. J. Rob. Res.*, 26(4):361–375, 2007.

[33] K. Lee, D. Kim, W. Chung, H. W. Chang, and P. Yoon. Car parking control using a trajectory tracking controller. In *2006 SICE-ICASE International Joint Conference*, pages 2058–2063, 2006.

[34] Q. Lu, Y. Sun, and S. Mei. *Nonlinear Control Systems and Power System Dynamics*. Kluwer Academic Publishers, Norwell, MA, USA, 2001.

[35] A. D. Luca, G. Oriolo, and C. Samso. Feedback control of a nonholonomic car-like robot. In J. Laumond, editor, *Robot Motion Planning and Control*, volume 229 of *Lecture Notes in Control and Information Sciences*, pages 171–253. Springer Berlin, 1998.

[36] M.Aicardi, G.Casalino, A. Bicchi, and A.Balestrino. Closed loop steering of unicycle like vehicles via lyapunov techniques. *Robotics Automation Magazine, IEEE*, 2(1):27 –35, mar 1995.

[37] P. Martin and P. Rouchon. Feedback linearization and driftless systems. *Mathematics of Control, Signals, and Systems*, 7:235–254, 1994.

[38] F. N. Martins, M. Sarcinelli-Filho, T. F. Bastos, and R. Carelli. Dynamic modeling and adaptive dynamic compensation for unicycle-like mobile robots. In *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pages 1 –6, june 2009.

[39] D. E. Miller and R. H. Middleton. On limitations to the achievable path tracking performance for linear multivariable plants. *IEEE Transactions on Automatic Control*, 52(11):2586–2601, 2008.

[40] R. M. Murray, M. Rathinam, and W. Sluis. Differential flatness of mechanical control systems: A catalog of prototype systems. In *Proceedings of the 1995 ASME International Congress and Exposition*, 1995.

[41] C. Nielsen. Maneuver regulation, transverse feedback linearization and zero dynamics. Master's thesis, Department of Electrical and Computer Engineering University of Toronto, 2004.

[42] C. Nielsen, C. Fulford, and M. Maggiore. Path following using transverse feedback linearization: Application to a maglev positioning system. *Automatica*, 46:585–590, March 2010.

[43] C. Nielsen and M. Maggiore. Maneuver regulation via transverse feedback linearization: Theory and examples. *Symposium on Nonlinear Control Systems (NOLCOS)*, September 2004.

[44] C. Nielsen and M. Maggiore. On local transverse feedback linearization. *SIAM J. Control Optim.*, 47(5):2227–2250, 2008.

[45] A. Pressley. *Elementry Differential Geometry*. Springer, New York, 2000.

[46] C. C. Pugh. *Real mathematical analysis*. Springer, New York, 2002.

[47] P. Rouchon, M. Fliess, J. Lvine, and P.Martin. Flatness, motion planning and trailer systems. In *In: Proceedings. Conf. on Decision and Control*, pages 2700–2705, 1993.

[48] J.E. Rowe. A new method of finding the equation of a rational planer curve form its parametric equaions. *Math Society*, pages 338–340, 1916.

[49] J.E. Rowe. The equation of rational planer curve derived form its parametic equaitons (second paper). *Math Society*, pages 304–307, 1917.

[50] C. Samson. Time-varying feedback stabilization of car-like wheeled mobile robots. *Int. J. Rob. Res.*, 12:55–64, February 1993.

[51] S. Sastry. *Nonlinear systems: analysis, stability, and control*. Springer, New York, 1999.

[52] T. W. Sederberg, D. C. Anderson, and R. N. Goldman. Implicit representation of parametric curves and surfaces.

[53] R. Skjetne, T. I. Fossen, and P. V. Kokotović. Robust output maneuvering for a class of nonlinear systems. *Automatica*, 40(3):373 – 383, 2004.

[54] D. Soetanto, L. Lapierre, and A. Pascoal. Adaptive, non-singular path-following control of dynamic wheeled robots. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, volume 2, pages 1765 – 1770 Vol.2, dec. 2003.

[55] O. J. Sørdalen. Conversion of the kinematics of a car with n trailers into a chained form. In *ICRA (1)*, pages 382–387, 1993.

[56] A. Tayebi, M. Tadjine, and A. Rachid. Stabilization of nonholonomic systems in chained form: Application to a car-like mobile robot. In *IEEE Conference on Control Applications - Proceedings*, pages 195–200, 1997.

[57] C. J. Tomlin and S. S. Sastry. Switching through singularities. In *Decision and Control, 1997., Proceedings of the 36th IEEE Conference on*, volume 1, pages 1 –6 vol.1, dec 1997.

[58] V. Utkin. *Sliding Mode Control in Electro-Mechanical System*. CRC Press., 1999.

[59] M. van Nieuwstadt, M. Rathinam, and R. M. Murray. Differential flatness and absolute equivalence of nonlinear control systems. *SIAM J. Control and Optimization.*, 36:1225–1239, July 1998.

[60] R. J. Walker. *Algebraic Curves*. Dover publications, Inc., 180 Varick Street, N.Y., USA, 1949.

[61] G. C. Walsh and L. G. Bushnell. Stabilization of multiple input chained form control systems. *Systems and Control Letters*, 25(3):227 – 234, 1995.

[62] D. Wang and G. Xu. Full-state tracking and internal dynamics of nonholonomic wheeled mobile robots. *IEEE/ASME Transactions on Mechatronics*, 8(2):203–214, 2003.

[63] H. Yalcin, M. Unel, and W. Wolovich. Implicitization of parametric curves by matrix annihilation. *Int. J. Comput. Vision*, 54:105–115, August 2003.